International Journal on Advances in Systems and Measurements

2015 vol. 8 nr. 3&4

The *International Journal on Advances in Systems and Measurements* is published by IARIA.
ISSN: 1942-261x
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Systems and Measurements
Volume 8, Number 3 & 4, 2015

Mohamed Eltoweissy, Pacific Northwest National Laboratory / Virginia Tech, USA
Paulo Felisberto, LARSyS, University of Algarve, Portugal
Miguel Franklin de Castro, Federal University of Ceará, Brazil
Mounir Gaidi, Centre de Recherches et des Technologies de l'Energie (CRTEn), Tunisie
Eva Gescheidtova, Brno University of Technology, Czech Republic
Tejas R. Gandhi, Virtua Health-Marlton, USA
Teodor Ghetiu, University of York, UK
Franca Giannini, IMATI - Consiglio Nazionale delle Ricerche - Genova, Italy
Gonçalo Gomes, Nokia Siemens Networks, Portugal
Luis Gomes, Universidade Nova Lisboa, Portugal
Antonio Luis Gomes Valente, University of Trás-os-Montes and Alto Douro, Portugal
Diego Gonzalez Aguilera, University of Salamanca - Avila, Spain
Genady Grabarnik,CUNY - New York, USA
Craig Grimes, Nanjing University of Technology, PR China
Stefanos Gritzalis, University of the Aegean, Greece
Richard Gunstone, Bournemouth University, UK
Jianlin Guo, Mitsubishi Electric Research Laboratories, USA
Mohammad Hammoudeh, Manchester Metropolitan University, UK
Petr Hanáček, Brno University of Technology, Czech Republic
Go Hasegawa, Osaka University, Japan
Henning Heuer, Fraunhofer Institut Zerstörungsfreie Prüfverfahren (FhG-IZFP-D), Germany
Paloma R. Horche, Universidad Politécnica de Madrid, Spain
Vincent Huang, Ericsson Research, Sweden
Friedrich Hülsmann, Gottfried Wilhelm Leibniz Bibliothek - Hannover, Germany
Travis Humble, Oak Ridge National Laboratory, USA
Florentin Ipate, University of Pitesti, Romania
Imad Jawhar, United Arab Emirates University, UAE
Terje Jensen, Telenor Group Industrial Development, Norway
Liudi Jiang, University of Southampton, UK
Kenneth B. Kent, University of New Brunswick, Canada
Fotis Kerasiotis, University of Patras, Greece
Andrei Khrennikov, Linnaeus University, Sweden
Alexander Klaus, Fraunhofer Institute for Experimental Software Engineering (IESE), Germany
Andrew Kusiak, The University of Iowa, USA
Vladimir Laukhin, Institució Catalana de Recerca i Estudis Avançats (ICREA) / Institut de Ciencia de Materials de Barcelona (ICMAB-CSIC), Spain
Kevin Lee, Murdoch University, Australia
Andreas Löf, University of Waikato, New Zealand
Jerzy P. Lukaszewicz, Nicholas Copernicus University - Torun, Poland
Zoubir Mammeri, IRIT - Paul Sabatier University - Toulouse, France
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Stefano Mariani, Politecnico di Milano, Italy
Paulo Martins Pedro, Chaminade University, USA / Unicamp, Brazil
Don McNickle, University of Canterbury, New Zealand
Mahmoud Meribout, The Petroleum Institute - Abu Dhabi, UAE
Luca Mesin, Politecnico di Torino, Italy

Marco Mevius, HTWG Konstanz, Germany
Marek Miskowicz, AGH University of Science and Technology, Poland
Jean-Henry Morin, University of Geneva, Switzerland
Fabrice Mourlin, Paris 12th University, France
Adrian Muscat, University of Malta, Malta
Mahmuda Naznin, Bangladesh University of Engineering and Technology, Bangladesh
George Oikonomou, University of Bristol, UK
Arnaldo S. R. Oliveira, Universidade de Aveiro-DETI / Instituto de Telecomunicações, Portugal
Aida Omerovic, SINTEF ICT, Norway
Victor Ovchinnikov, Aalto University, Finland
Telhat Özdoğan, Recep Tayyip Erdogan University, Turkey
Gurkan Ozhan, Middle East Technical University, Turkey
Constantin Paleologu, University Politehnica of Bucharest, Romania
Matteo G A Paris, Universita` degli Studi di Milano,Italy
Vittorio M.N. Passaro, Politecnico di Bari, Italy
Giuseppe Patanè, CNR-IMATI, Italy
Marek Penhaker, VSB- Technical University of Ostrava, Czech Republic
Juho Perälä, Bitfactor Oy, Finland
Florian Pinel, T.J.Watson Research Center, IBM, USA
Ana-Catalina Plesa, German Aerospace Center, Germany
Miodrag Potkonjak, University of California - Los Angeles, USA
Alessandro Pozzebon, University of Siena, Italy
Vladimir Privman, Clarkson University, USA
Konandur Rajanna, Indian Institute of Science, India
Stefan Rass, Universität Klagenfurt, Austria
Candid Reig, University of Valencia, Spain
Teresa Restivo, University of Porto, Portugal
Leon Reznik, Rochester Institute of Technology, USA
Gerasimos Rigatos, Harper-Adams University College, UK
Luis Roa Oppliger, Universidad de Concepción, Chile
Ivan Rodero, Rutgers University - Piscataway, USA
Lorenzo Rubio Arjona, Universitat Politècnica de València, Spain
Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance, Germany
Subhash Saini, NASA, USA
Mikko Sallinen, University of Oulu, Finland
Christian Schanes, Vienna University of Technology, Austria
Rainer Schönbein, Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), Germany
Guodong Shao, National Institute of Standards and Technology (NIST), USA
Dongwan Shin, New Mexico Tech, USA
Larisa Shwartz, T.J. Watson Research Center, IBM, USA
Simone Silvestri, University of Rome "La Sapienza", Italy
Diglio A. Simoni, RTI International, USA
Radosveta Sokullu, Ege University, Turkey
Junho Song, Sunnybrook Health Science Centre - Toronto, Canada
Leonel Sousa, INESC-ID/IST, TU-Lisbon, Portugal

International Journal on Advances in Systems and Measurements
Volume 8, Numbers 3 & 4, 2015

CONTENTS

DAME: On-demand Internet-scale SAML Metadata Exchange

Michael Grabatin, Wolfgang Hommel, Stefan Metzger, and Daniela Pöhn
Leibniz Supercomputing Center
Munich Network Management Team
85748 Garching n. Munich, Germany
Email: [grabatin,hommel,metzger,poehn]@lrz.de

***Abstract*—Inter-organizational IT service access based on the Security Assertion Markup Language (SAML), the predominant standard for Federated Identity Management (FIM), suffers from metadata scalability issues when Identity Providers (IDPs) and Service Providers (SPs) from different federations are involved. This article presents Dynamic Automated Metadata Exchange (DAME) for SAML-based FIM and its open source implementation, GÉANT-TrustBroker, which is currently in preparation for pilot operations within the pan-European research and education network, GÉANT. Based on the DAME metadata broker architecture and workflows, the concept of Internet-scale dynamic virtual federations is introduced and life-cycle management concepts are discussed; special emphasis is put on the risk management aspects of GÉANT-TrustBroker.**

Keywords–Federated Identity Management; SAML; Shibboleth; Inter-Federation; Trust-Management.

I. INTRODUCTION

Identity & access management (I&AM) is the umbrella term for managing users and their permissions. While I&AM can be applied to individual IT services, such as a web application, I&AM architectures typically cover the majority of all IT services within an organization. For example, higher education institutions use I&AM systems to manage the accounts of all of their students, staff, faculty, guests, and alumni along with their individual access rights to email servers, file storage, learning management systems, and other IT services. I&AM has many challenging organizational aspects, such as defining responsibilities for data quality and master systems for individual information, but its implementation technology has matured over the past 15 years. Typically, central Lightweight Directory Access Protocol (LDAP) based directory services or other database management systems aggregate all the required data and make it available to the I&AM-connected IT services.

Given the sensitivity of the personally identifiable information (PII) stored within I&AM systems, read access is only granted to trusted IT services in a selective manner. For example, IT services, which only need to authenticate users based on their usernames and passwords, will not be allowed to also read, for example, their email addresses and telephone numbers. Therefore, I&AM systems authenticate the IT services that make use of them and are often operated in firewall-protected internal networks. As a consequence, I&AM systems are not suited for inter-organizational use cases, such as multiple users from different universities and industry partners accessing a web-based collaboration platform as part of a research project.

Federated Identity Management (FIM) provides partial solutions for inter-organizational use cases. In its basic form, it assigns the role of Identity Providers (IDPs) and Service Providers (SPs) to organizations: IDPs are the home organizations of users and provide authentication as well as authorization services, whereas SPs operate IT services that can be used by multiple IDPs. Sets of at least one IDP and one SP are referred to as federations. In higher education, several dozens of national federations have been established over the past 10 years, such as InCommon in the United States, SWITCH-AAI in Switzerland, and DFN-AAI in Germany. In industry, federations are typically established for sector-specific supply chains, such as the pan-European automotive platform Odette. The Security Assertion Markup Language (SAML) is the predominant technology in both professional areas, whereas consumer-oriented Internet services often make use of more lightweight approaches such as OpenID Connect.

The large-scale real-world application of FIM is subject to two major distinct challenges: First, IDPs and SPs need each other's metadata, i. e., information about technical communication endpoints and server certificates for message signatures and encryption. Second, IDPs must provide information about their users, referred to as *user attributes*, in a data format compatible to the SP and its IT service. Existing federations solve the first problem by first centrally aggregating the metadata of each IDP and SP and then distributing the complete metadata package to each participating organization. The second problem is typically solved by defining a federation-wide user data model, commonly referred to as *federation schema*. Both solutions work well for average-size federations, but hit a dead end when users want to access IT services across federations' borders, e. g., in international research or cross-industry-sector projects: while inter-federations, such as eduGAIN, attempt to aggregate and distribute the SAML metadata of several national federations, the organizational overhead as well as the technical performance impact of huge metadata sets deters many organizations from participating. Also, given the heterogeneity of federation schemes, successful user attribute exchange is limited to their intersection, leaving many IT services with a lack of information about individual users that limits their functionality.

In [1], we presented a SAML metadata broker for dynamic federations and inter-federations. It supports the user-triggered, on-demand exchange of SAML metadata between pairs of IDPs and SPs whenever a user from a specific IDP attempts to access a particular SP service for the first time. It significantly simplifies the organizational and technical aspects of SAML setups across existing federations' borders and optimizes the technical scalability by avoiding the aggregation of metadata that is not relevant to individual organizations. It also supports inter-federation user attribute exchange by providing a repository, which allows for the sharing and re-use of conversion rules. Along with several improvements, the approach has since been refined as follows: First, the protocol has been

formally specified in the IETF Internet-Draft *Integration of Dynamic Automated Metadata Exchange into the SAML 2.0 Web Browser SSO Profile* (DAME). Second, an open source implementation based on the popular FIM software suite Shibboleth, called GÉANT-TrustBroker (GNTB), has been developed and tested within the pan-European research and education network GÉANT; it currently is being prepared for multi-national pilot operations and scheduled for integration into the GÉANT service portfolio as a part of the ongoing, EC-funded GÉANT GN4 project.

In this article, the background, design rationale, and current state of both DAME and the GÉANT-TrustBroker implementation are presented in detail. It is structured as follows: In Section II, related scientific work and practical approaches are discussed. Section III then explains the chosen broker-based approach along with its architecture and workflows. The concept of dynamic virtual federations along with their life-cycle and management procedures are then detailed in Section IV. Afterwards, the GÉANT-TrustBroker implementation is presented in Section V, followed by a discussion of its risk management aspects in Section VI. The article is concluded by a summary and outlook to ongoing work in Section VII.

II. Related Work

Though FIM is used in the recent years and many theoretical and practical solutions were designed, scalable and at the same time secure solutions are rarely found. All related work, which was investigated, concentrates on only one particular aspect and does not see the problem as a whole. First practical solutions are shown, before scientific approaches are explained.

A. Practical Approaches

Although SAML does not specify that SAML metadata of each participating entity, i. e., IDP, SP, and attribute authority, needs to be aggregated and exchanged beforehand, it is the current practice. In order to aggregate and exchange metadata, several federations have established metadata registry tools. The Swiss federation SwitchAAI was the first NREN federation to develop a so-called Resource Registry [2], where entities can register their metadata and update information. Based on all uploaded metadata, the national metadata file is aggregated, which then can be downloaded by the participants. Though the national web tool helps entities to manage their information, many manual steps are required and the local configuration needs to be updated manually.

Public Endpoint Entities Registry (PEER) by Ian Young et al. [3] is another practical solution. The implementation of PEER is called REEP and can be used by any entity, independent of the federation and the protocol used. Though PEER moves the metadata aggregation from federations one layer up to a central service, the metadata is still aggregated. Another drawback are the manual steps, e. g., to generate an attribute filter adjusted to the IDP.

Another way to distribute metadata is the submitted Internet-Draft (I-D) Metadata Query Protocol by Ian Young [4], which has a profile for SAML environments. In this approach, metadata can be retrieved by hypertext transfer protocol (HTTP) GET requests, which allow dynamic metadata distribution. Therefore, Metadata Query Protocol solves the problem of huge aggregated metadata files, while manual steps are needed to adjust the local configuration. Furthermore, Metadata Query Protocol does not suggest a workflow to exchange metadata on-demand and establish trust between two entities, i. e., SP and IDP.

B. Scientific Approaches

The scientific approach of Federated Attribute Management and Trust Negotiation (FAMTN) by Bhargav-Spantzel et al. [5] assumes that each SP can act as an IDP. Since no IDP exists, the user information need to be stored at the users. Internal users of the FAMTN system are supposed to perform negotiations by exploiting their single sign-on (SSO) ID without repeating identity verifications. External users need to declare all their attributes in the first communication, in order to receive a temporary user ID. At the second communication, the SSO ID is exploited, though it could be misused for attacks. It might appear that a provider needs less or more attributes, leading to violations of data minimization or further negotiations between providers.

Arias Cabarcos' et al. approach of IdMRep [6] shifts from pre-configured cooperation to dynamic trust establishment by a distributed reputation-based mechanism based on local dynamic trust lists (DTLs) and external reputation data. DTLs can, e. g., receive recommendations from other entities, when a cooperation was successfully ended. Hence, the cooperation runs through different phases: receiving and evaluating information, local calculation of the risk and trust values, dynamic decision based on available information, and monitoring and adjusting trust level. This mechanism does not work well for new entities. Because of the amount of data processing required for all external and internal trust information especially in inter-federations, this results in yet another bottleneck. It is vulnerable to Sybil attacks. Furthermore, the problem of different attributes, syntax, and semantics is not considered.

The approach Dynamic Identity Federation by Md.Sadek Ferdous and Ron Poet [7] also concentrates on the dynamic trust. Dynamic Identity Federation distinguishes between fully trusted, semi-trusted, and untrusted entities. Authenticated users are allowed to add SPs to their IDPs, while SPs add the IDPs to their local trust anchor list for further usage. The user establishes the trust by generating a code at his first authentication. He then informs the SP about the code and the EntityID of the IDP. After verification, the SP generates a request with two invisible fields, i. e., MetaAdd and ReturnTo. Both fields are used for the metadata exchange, while the IDP needs to evaluate the value of MetaAdd. When the user gives his consent, the IDP adds the chosen SP to the list of semi-trusted entities. Semi-trusted entities are not allowed to receive sensitive attributes. Untrusted entities are given the National Institute of Standards and Technology (NIST) level of assurance (LoA) 1. If the SP is not known by the IDP, a proxy could be used complicating the trust establishment. The trust establishment via the user generating and forwarding a code is not user friendly, while both invisible fields are not necessary. The fragmentation into trusted, semi-trusted, and untrusted entities as well as the usage of NIST LoA 1 does not reflect real world with its different LoA schemes and the trust relationships.

In sum, different aspects can be adopted, though neither approach tries to solve the problem as a whole. While the Metadata Query Protocol is a scalable approach for distributing metadata, it needs to be included in a scalable architecture

for dynamic trust establishment. The trust framework needs to reflect real world, while being flexible. IdMRep could be added as a trust layer on top of LoA. Furthermore, the solution needs to be secure for the participants.

III. SAML Metadata Broker

The project GÉANT-TrustBroker was established within GÉANT to address the challenges of SAML metadata exchange. The central trusted third party (TTP) GNTB is, as described in [1], [8], and [9], an on-demand repository for metadata and conversion rules. It extends existing discovery services, formerly known as WAYF (Where Are You From?), in order to locate the appropriate IDP. As both entities, i. e., IDP and SP are known by the TTP, metadata can be exchanged on-demand, if triggered by the user. In order to exchange and integrate the metadata automatically into the local configuration, IDPs and SPs need an extension for communicating with the TTP, as shown in Figure 1.

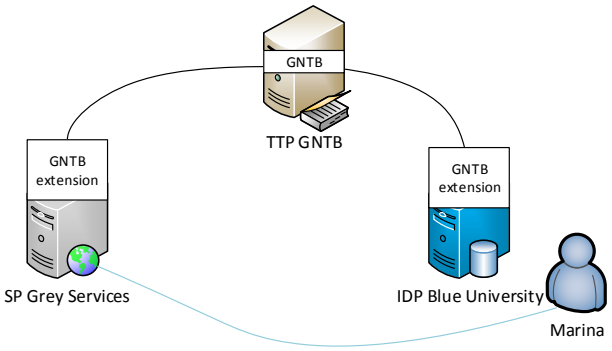


Figure 1. Basic architecture for dynamic metadata exchange with GNTB

By automating the metadata exchange, GNTB simplifies the discovery of entities and establishes the technical trust in dynamic virtual federations, while it improves the scalability of metadata release. The cooperation is not limited to an existing federation or inter-federation. Instead, the metadata can be exchanged across borders, making the federation virtual. As the metadata is not aggregated beforehand at the different providers, but exchanged on-demand, the size of the metadata files integrated at each provider is reduced. If the user trusts the SP, he can trigger the technical trust establishment at first time use of the SP, as described in the next section. The metadata is then exchanged and automatically integrated into the local configuration of the user's chosen IDP and the requested SP by extensions of the predominate software. Because the TTP keeps track of the established technical trust relationships, it can trigger the download of updated metadata information if needed. Furthermore, a conversion rule repository is provided, in order to extend and translate the amount of attributes used in collaborations. In the following section the different workflows are explained in detail. Last but not least, the architecture of this approach is visualized.

A. Workflows

In this section, three different types of workflows will be explained: management workflows, conversion rule workflow, and the core workflow. *Management workflows* on the one hand allow SP and IDP administrators to register, upload, update, and delete metadata information as well as attribute conversion rules. Uploading metadata information requires a proof-of-ownership verification step. This can be technically implemented by creation of a specific resource in the document root of the web service for that domain with a specific, random string given. Once created, the administrator can trigger the verification process and, if receiving an 200 OK status code in the response message, the metadata information will be inserted. Alternatively, certificate based verification or simple mechanisms, e. g., comparison of the entities name with the mail address' domain of the logged in user can also be implemented. This degree of automation keeps humans on the broker side out of the loop, so newly registered entities do not have to wait for manual approval of their application.

Conversion rule workflow: Since SP and IDP are usually not members of the same (inter-)federation, syntax and semantics of the user attributes, i. e., the attribute schema used, vary. Fortunately, because the metadata of a SP usually contains information about the required attributes, the IDP can determine if it can fulfill the attribute requirements directly or further attribute conversion will be required. In the latter case, the IDP can now check whether suitable rules are available at GNTB. This step can be automated by scripts. If suitable rules were found, these will be automatically downloaded and integrated into IDP's attribute resolver and filter configuration. This conversion rule workflow is not part of the core workflow, but can be triggered by it.

On the other hand, the *core workflow* (presented in Figure 2) builds up the provider pairing or virtual federation. This core workflow was specified as an Internet-Draft and submitted to the IETF as DAME.

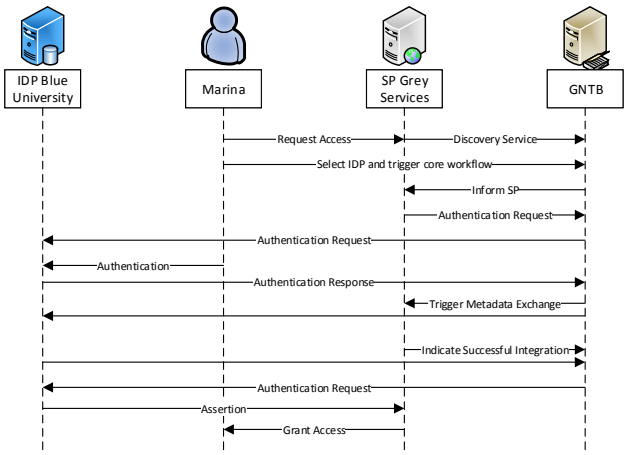


Figure 2. DAME core workflow for provider pairing

Explaining the core workflow, we assume that researcher Marina from an IDP Blue University, member of the federation Blue, requests access to a protected resource provided by SP Grey Services, which is not a member of the same federation. The often seen embedded discovery service on the SP lists all already trusted IDPs. We assume that Marina's IDP is not listed

there, so she can trigger the DAME workflow. Comparable to other typical SAML-based workflows, Marina is redirected to GNTB, technically speaking to its centralized discovery service component. Provided that both IDP Blue University and SP Grey Services are already registered and uploaded their metadata to the TTP using the management functions mentioned previously, Marina can pick the IDP she wants to use. Rather than redirecting Marina directly to the chosen IDP for authentication, GNTB passes the information about the selected IDP back to the requested SP. If Grey Services decides that users from that chosen IDP can be accepted, it sends a generated SAML authentication request to GNTB, which temporarily stores it. GNTB, now in the role of a regular SP, generates a new SAML authentication request and redirects Marina to her chosen IDP Blue University. This two-part user authentication is necessary to prevent malicious users to add arbitrary IDPs' metadata to any SP and vice versa. After successful user authentication and receiving the SAML assertion in the corresponding response message, GNTB triggers the IDP and SP afterwards to download and integrate each other's metadata. This can be done either by using Young's Metadata Query Protocol, explained in Section II, or any other appropriate mean, like a simple web service or REST API function, as described in the next section. After updating each others' configuration, GNTB forwards the temporarily stored SAML authentication request to the IDP. Unless forced user re-authentication is required by the SP, the IDP immediately responds with a SAML authentication assertion to Grey Services and Marina's browser is redirected back to the requested service and access will be granted. If Marina inadvertently has chosen her IDP, which Grey Services already trusts, a regular FIM authentication workflow without further involvement of GNTB is initiated. Analogous, if the metadata information has been exchanged and the technical trust has been established successfully, GNTB is not involved anymore.

In order to manage these workflows, a TTP was designed, which interacts with IDPs' and SPs' extensions.

B. Architecture

In this section, the architecture, internal data model of the TTP, and the data access layer are described. Besides the GNTB core service providing a centralized discovery service for IDP selection and storing metadata information on each provider entity, an DAME extension has to be installed at IDPs and SPs to enable metadata exchange and automatic integration as well as the attribute conversion rule handling.

While both metadata information and attribute conversion rules are stored in TTP's file system, a relational database is used to support the provided management functions. In contrast to the tables described in [1], the proof of concept has additional tables to realize all added functionalities (in alphabetical order), e. g:

- attributes: This table stores information on source or destination attributes, which can be used in attribute conversion rules. To identify the attributes the unique object identifiers, e. g., urn:oid:1.3.6.1.4.1.5923.1.1.1.6 (eduPersonPrincipalName) are used.
- convRules: This table contains information about an attribute conversion rule. Besides a unique identifier of the rule, its status, creation date, a short description, the owner information, the location in the file system is stored. The result of the attribute conversion is expressed as target, which links to the appropriate attribute.
- metadata: Comparable to the convRules table, this table contains information about the provider entities' metadata, e. g., the unique entityID, the location of the metadata file stored on the TTP's file system, a short comment, creation data, and owner information.
- organization: Each provider can be associated with an organization.
- providers: Stores all providers and relevant information like the entityID .
- providerWhitelist and providerBlacklist: The usage of a whitelist or blacklist enables IDP or SP administrators to explicitly allow or reject certain providers and, therefore, the metadata exchange. It is based on DNS domain names and is intertwined with the validation of new entities regarding domain ownership. As the permissions to add and change data is validated and administrators can only use this functionality for their own entity, spoofing is prevented.
- providerUserRelationship: Information about the association between provider entities and users.
- ruleDependencies: Attribute conversion rules converts some input attributes into a target attribute. This table stores information about the source attributes required for conversion.
- ruleStatus: This table contains the available rule status.
- spIdPRelationsship: Stores information about the SP to IDP relationship, i. e., information about existing virtual federations.
- users: Information about the users registered at the TTP, i. e., administrators of provider entities.

Administrators of IDPs and SPs can use basic features, such as the registration of new metadata and uploading or searching for appropriate attribute conversion rules via the web interface and to further automate some management tasks by using provided command-line-tools. The GNTB's core service, therefore, provides an application programming interface (API) consisting of a number of API functions, which were described in [1]. The API function for downloading conversion rules is publicly available, as they do not contain PII. All other functions are classified as internal use only, authentication required and additionally restricted to own account or organization. For user management, the creation of new, updating or deleting existing accounts exists. Before registration of a new provider entity and uploading its metadata, it has to be verified that this entity does not already exist to avoid duplicates. Also, for the proof-of-ownership of the registered metadata or to ensure syntactically correctness of the metadata file as well as notification of administrators, the API provides appropriate validation functions.

To support the core service, the data access layer provides function to trigger the download of the metadata information. The download can be done by the Metadata Query Protocol or any other method. The extensions, installed on the IDPs and SPs, allow the automated integration of the downloaded

metadata as well as attribute conversion rules. This results in immediate use of a service by the user.

IV. Federations in SAML Metadata Broker

The metadata is exchanged on demand between IDP and SP as described in the previous section. Therefore, as the metadata is not aggregated and then distributed as a whole any more, static federations are technically not needed. The metadata is exchanged on-demand between cooperating IDPs and SPs. When IDPs and SPs have integrated each other's metadata, dynamic virtual federations can be built. This depends on the situation, e. g., if only one IDP-SP pair cooperates, it is a bilateral federation. If more IDPs and SPs cooperate, they can dynamically build a federation. The concept of dynamic virtual federations is described in this section, followed by the design of a federation administration tool for those more fixed federations, which require opt-in. Both, dynamic virtual federations and the federation administration tool, are available with an extended version of the GNTB TTP.

A. Concept of Dynamic Virtual Federations

Dynamic virtual federations are built dynamically, depending on the needs of the users. The second dynamic aspect is the dynamic appearance of the federation. They are built dynamically, new organizations or single providers can join, while others can leave. The dynamic virtual federation can be closed, when the project or reason for the cooperation ends. This means that the size of the federation is dynamically adjusted. The degree of dynamics depends on the reason for the federation. While project and cooperation federations have a shorter length of life, national federations are less dynamic. If the federation is closed, e. g., due to an official project cooperation, the size of the federation will not change, whereas open federations have greater dynamics in relation to the size. Another aspect are service level agreements for financial services, which need to be in place beforehand. This also has impact on the dynamics of a federation. Virtual means the federation is orthogonal to existing static federations. The federation has members from different federations, which want to cooperate. Existing structures are suspended or weakened, in order to allow efficient international cooperation. Based on the characteristics dynamic and virtual, the defined term of national federations disappears. Hence, a federation is a cooperation of members, i. e., IDPs and SPs, which cooperate due to needs of users. The dynamic virtual federation can be characterized as follows:

- The structure of the cooperation can be an ad-hoc federation, a hub-and-spoke federation as well as an identity network.
- The amount of members is flexible. A bilateral federation is possible as well as a fixed number of participants and, most likely, a complex structure.
- The structure of the group depends on the needs of the participants. It can be open, open with restrictions and closed, although closed federations are opposed to the characteristic dynamic and, therefore, not likely.
- The dimension of the federation is open. It can be local, regional, national or international.
- The organizational dimension is intra-federation, though dynamic virtual inter-federations can be established by federations.
- The duration of the federation depends on the requirements and can be limited to the project length or fixed-terms.
- The sort of collaboration can be project, virtual organization, or by other reasons.
- The coordination depends on the requirements and can be implicit, explicit or mixed structure.
- The process of establishment is spontaneous, event-driven or as needs arise. Planned establishments are possible for, e. g., projects.
- The circle of trust can be anything but static.
- The degree of commitment is probably unwritten agreements, as long as contracts and service level agreements are not needed.
- The trust relationship between members is most likely direct, though it can be indirect as well.

When two or more dynamic virtual federations need to cooperate, dynamic virtual inter-federations can to be established. The establishment is likewise dynamic and virtual. If there are enough connects between the participating federations established, e. g., at least 20 percent of all possible connections, the TTP GNTB can automatically build an inter-federation. In between, the (inter-)federation can change, when entities opt-in, while others opt-out. If the (inter-)federation is not needed anymore, e. g., if a project or other sort of cooperation is terminated, the (inter-)federation can be closed. One precondition is the approval of the federations to build an inter-federation. If the federations do not want a dynamic inter-federation, they can use the federation administration tool to establish a static inter-federation.

B. Federation administration tool

In order to help managing federations and inter-federations requiring formal opt-in, a federation administration tool at the GNTB TTP needs to be implemented with the following functionalities, among others:

- establish a federation,
- define an application process,
- accept and reject possible members,
- establish and update policies,
- suspend members, and
- change permissions.

As policies, described in Extensible Markup Language (XML) files, need to be uploaded, changed, obeyed, and deleted, a policy management is needed. The policies are, similar to metadata and conversion rules, stored in a policy repository. Based on policies and other requirements, federation administrators can decide, if an IDP or SP is allowed to join a federation. By applying for membership in a federation, the entity accepts the policies. This also results in quality assurance similar to the current practice in national NREN federations. The core workflow is as follows:

Step 1: An entity, i. e., an IDP or SP, would like to join a federation. The desire is expressed by applying to a federation via the administrative web interface.

Step 2: The application is stored at the TTP as a status and the federation administrator is informed. The

federation administrator checks if all requirements are fulfilled. Depending on the requirements, this step can be automated.

Step 2a: If the entity is controlled manually, the federation administrator needs to check policies, other requirements and/or audits.

Step 2b: If the federation requires a specific certificate, it needs to be issued and sent to the entity.

Step 3: The federation accepts/denies the entity. The result is stored in the database of the TTP.

The same basic workflow is used, if a federation wants to participate in an inter-federation. The federation's members should be notified about the result. If an existing federation or inter-federation decides to use the TTP, all members can be bulk imported, though they need to accept the membership officially. In order to represent the basic federation workflow with its status in the database, federations and inter-federations first need to be able to register and assign roles to administrators. Policies and other requirements must be stored or referenced in the database as well. The status of the federation respectively inter-federation is important as it can be currently added, updated or, e. g., after a project, deleted. The same appears for policies. When a policy was updated, members need to be notified and, in the worst case, checked against it. Furthermore, the status of the relationship between entity and federation as well as federation and inter-federation is crucial. This information, as stored in the database, can be used for federation and inter-federation statistics.

In contrast to the current situation with different federation tools, these pre-defined workflows minimize the problems, described by Harris [10]. The biggest improvement is made to the metadata flow problem. The upstream and downstream of metadata varies by federations. As a single tool with predefined interfaces is used and the metadata is exchanged on demand, the problem disappears. At the same time, work load from the federation administrators is shifted to the TTP. In order to allow both types of (inter-)federations, dynamic virtual and fixed (inter-)federations, these were investigated and a federation administration tool was designed. The dynamic virtual federation can reuse the federation administration tool by fully automating the workflow when a certain percentage of technical trust was established. These additional functionalities can be seamlessly integrated into the proof of concept implementation. The proof of concept implementation of the TTP and the extensions for IDPs and SPs, required by both dynamic virtual federations and the federation administration tool, is described in the next section.

V. Proof of Concept

The primary goal of the proof of concept implementation is to show the possibility that an existing SAML implementation can be extended to support the DAME protocol without breaking SAML and the interoperability between the different parties and without complicating the authentication workflow for the user. Additionally, the coexistence of federations and the dynamic metadata exchange introduced by DAME is to be shown.

The proof of concept implementation was done by extending the Shibboleth SAML implementation. Shibboleth was chosen because it is the primary SAML implementation used across European institutions, followed by simpleSAMLphp. The documentation of installing and running Shibboleth is available from different sources, like [11] and [12]. Also, high profile extensions, like uApprove [13], demonstrate that it is possible to extend Shibboleth and have many IDP administrators install your extension. Additionally, Shibboleth provides all three components that need to be extended in order to implement DAME. The Identity Provider, Service Provider, and a Centralized Discovery Service (CDS). The first and second, IDP and SP, must be extended to support the automated, user initiated, exchange of metadata. The latter is used as a discovery service, where users select their IDPs, and is extended to provide a web-based management interface of the TTP for the participating providers. The scenarios and the evaluation of the implementation is discussed later onwards in this section.

The proof of concept network running on virtual machines was also used to demonstrate and test that different versions and installation methods of the Shibboleth base software, which can be used with the DAME extensions. All machines are Debian based Linux virtual machines. As Debian 8 (Jessie) was released recently, the upgrade from Debian 7 to 8 could be tested on some machines, others are deliberately still running the old Debian version. This reflects a real wold scenario where systems are not always immediately updated to the latest version. The software versions used for testing are Apache Tomcat 6, 7, and 8 as the Java Web Application Server for the IDP as well as Apache web server 2.2 and 2.4 for the SPs. The TTP has only been tested on a Tomcat 7 server, but, as the CDS is very similar to the IDP and there were no issues running the IDP on Tomcat 6, 7 or 8, the CDS and the TTP extension are very likely to have no issues as well.

A. TTP Discovery Service

The trusted third party consists of three modules. First, the discovery service, to allow users to pick their IDP, second, the core metadata and conversion rule exchange mechanism, and third, the management interface for IDP and SP administrators to register and manage their providers. In the proof of concept implementation all three modules have been combined in an extension to the Shibboleth centralized discovery service. The CDS was chosen because it already implements the first module, the discovery service. Extending the existing implementation also made sense as the SAML discovery protocol is not modified by the extension and reusing an existing implementation decreases the chance of creating incompatible protocol versions. The only interface between the TTP and the CDS is the CDS' access all metadata registered at the TTP. The CDS also has to be notified, if a new IDP is registered or an existing one is modified. This is achieved by generating a complete metadata file that is including all registered IDPs, whenever there are changes to the available IDPs. This file can then be included by the CDS. The second and third modules, the actual TTP, are also part of the extension to the CDS, because this way the TTP can be distributed as a single extension. The installation of extensions on the CDS is very similar to the IDP, which makes installing the extension especially easy for administrators that are used to installing or updating extensions for the IDP.

Besides the CDS and the TTP, another discovery service is needed to implement DAME efficiently. The DAME protocol should only be used the first time, if the metadata of the IDP

is not available at the SP. So, the SP or the user need a way of determining whether the metadata is already available or not. In the proof of concept, this is done via the Shibboleth embedded discovery service (EDS). The EDS is installed on the same system as the SP and a user is redirected to this EDS for IDP discovery first. The EDS can then display all IDPs that are available at the SP. If the EDS knows about the IDP, which the user would like to use, the DAME protocol is never used and a regular SAML SSO can be done. If the desired IDP is not already listed at the EDS, the user can then be forwarded to the CDS at the TTP. This forwarding is implemented by adding a button to the web page generated by the EDS that will forward the original request to the CDS at the TTP. The EDS is mostly written in Java Script and has to be configured using a separate Java Script file. The EDS does not supply a method of implementing extensions, so, to implement the necessary changes, the EDS itself needed to be modified.

In contrast, the discovery service part of the CDS does not need to be modified. The CDS can be configured to read one or multiple metadata files and then extract the required information about the name and the communication endpoint of the IDPs. After the user selects an IDP, the CDS can relay the information about the selected IDP back to the SP as it normally would. The SP is then responsible of initiating the communication with the TTP, in order to trigger the metadata exchange.

The core TTP implementation consists of a module that handles the dynamic metadata exchange according to DAME. This module is also part of the CDS extension. This allows a more efficient communication between different modules. As described in the DAME workflow in Section III-A, this module receives an authentication request from a SP and validates its signature, in order to verify that the SP is legitimately trying to contact the IDP specified in the request. Following the SAML standard, this can only be done if the authentication request by the SP is not encrypted, as the signature is placed inside the encryption, which, as the message is directed at the IDP, the TTP cannot decrypt. Encrypted authentication requests are therefore discouraged, to provide security, while transmitting the request HTTPS should be used instead. After verification, the authentication request is stored in the users session and the TTP issues its own authentication request to the IDP to verify the user actually holds a valid account at the IDP, before initiating the metadata exchange.

The metadata exchange itself is done by getting the DAME extension element from the provider's metadata. This element must be supplied and specifies the location of the communication endpoint for initiating the metadata exchange. The TTP then sends an HTTP request to this endpoint and indicates, for which EntityID metadata should be downloaded and from where. The location of the metadata, therefore, does not have to be at the TTP itself. The current implementation of the TTP only allows for this as there is no way of specifying a remote location in the management interface, but this could easily be added.

The conversion rule exchange is done similarly. The IDP determines, which attributes the SP requests from the downloaded SP metadata. If the IDP is missing some or all of them, it can then asks the TTP, if there are conversion rules available that would use the attributes the IDP can provide to build the missing attributes. The TTP replies with a XML-formatted list of rules, which could be of use. The IDP can then filter the results and pick the rule it prefers, potentially the IDP could also try to activate multiple rules in a sandbox environment until it finds the one that works best. Alternatively, a reputation system could be implemented, so that IDPs prefer conversion rules already used or issued by the federation they are in. Votes then could be cast, e.g., by federations or IDPs that successfully tested and use the specific rules; however, as reputation systems are vulnerable to misuse and conversion rules affect sensitive personal data, any implementation that runs without supervisory control constitutes a risk. We therefore plan to gather practical experiences regarding how problematic redundant conversion rules become in the real world and will then address it as necessary in future work. The conversion rule interface is done via regular HTTP calls, so that also any other tool or extension could be used to query for conversion rules.

One remaining problem related to conversion rules is that they are currently specifically designed for the Shibboleth IDP. As the IDP uses XML files for configuration, the conversion rule is an EXtensible Stylesheet Language Transformation (XSLT) file adding new XML elements to the configuration. This method is not suited for simpleSAMLphp or other IDP implementations using a different format. In the future, an abstract syntax for conversion rules needs to be designed, which can automatically generate the correct conversion rule based on the IDP software.

The last part of the CDS extension is the management interface of the TTP. For the TTP to be usable, it needs to provide some core methods for the providers and their administrators, e. g.:

- User registration: Provider administrators are able to register at the TTP, in order to manage one or more of their systems. The user management also supports multiple users being able to manage the same provider.
- Provider registration: The provider administrators can register their providers at the TTP. The registration requires a unique EntityID and can be extended by an description of the provider. Additionally, a provider can be assigned to be part of a federation.
- Provider verification: To prevent obvious misuse of the TTP, an automated method of verifying that the person registering a provider is actually allowed to do so, has been implemented. The administrator must currently place a file with a randomly generated file name on a web server at the host name of the EntityID. Other methods, like email verification, are also possible.
- Metadata management: After registering at the TTP, the provider administrators are able to upload and modify the metadata of their provider.
- Conversion rule management: Administrators of the registered providers are able to create and modify conversion rules.

To manage the user, provider, and conversion rule information, a MySQL database is used. The metadata and conversion rule files are stored on the file system and referenced in the database. The metadata files are named by calculating a SHA-1 hash of the EntityID and appending a timestamp. This way, multiple versions of a metadata file can be stored and the

resulting file name contains only ASCII characters and is of fixed length.

B. IDP Software

In order to support the DAME protocol, the Identity Provider needs to be extended. The extension implements a new communication endpoint, which can be used by the TTP to trigger the download of new metadata, and a new metadata provider component, which manages the downloaded metadata and provides it to the other IDP components, like the authentication module.

The original proof of concept implementation was done for the Shibboleth IDP version 2, as it was the current version at that time and version 3 was not used by any production IDP. Because version 3 is now released and no longer in beta status, the DAME extension has been converted to an extension for IDP version 3. This was also a chance to improve the proof of concept implementation. The conversion of an extension between versions 2 and 3 is not trivial as much of the underlying structures and interfaces have been changed. But the version 3 extension is much easier to install and maintain from an administrator's perspective.

In general, the IDP extension adds three new modules to the IDP: The communication endpoint to receive metadata synchronization requests from the TTP, a metadata provider to manage the downloaded metadata, and a method of implementing conversion.

The communication endpoint is a relative straight forward HttpServlet in the IDP version 2 extension and a Spring Webflow in the IDP version 3 extension. To prevent misuse, it first checks if the request is originating from a trusted source. This source is identified by its IP address and has to be configured by the IDP administrator. If the request is allowed, the metadata is downloaded and passed to the metadata provider as described below. After the metadata has been synchronized, the attributes requested by the SP are compared to the attributes available at the IDP and, if some are missing, conversion rules are requested from the TTP.

The IDP is designed to support multiple metadata providers. Two general examples of the metadata providers, the IDP is shipped with, are file based metadata providers, that just read a local metadata file placed on the IDP by an administrator, and HTTP metadata providers, which periodically download the metadata file from a remote location and cache it locally. A special metadata provider is the chaining metadata provider, it can be used to combine multiple other metadata providers together. The IDP extension adds another type, the DAME metadata provider.

In the IDP version 2 extension, the DAME metadata provider was just able to read the files, which were downloaded via the DAME protocol. The version 3 extension is more advanced as it is basically a chaining metadata provider and uses a file backed HTTP metadata provider for each SP. The key difference to the chaining metadata provider is that the DAME metadata provider can be modified during runtime of the IDP. This is necessary to dynamically add new metadata. If a new metadata file is synchronized using the DAME protocol, the URL of the metadata is saved in a local file. The filename is the SHA-1 hash of the EntityID and the extension ".xml.loc" designates that this is the file containing the original location of the metadata file. The file based approach has been chosen over a database to keep the number of dependencies small. Those URLs are necessary to initiate all file backed HTTP metadata resolvers if the IDP is started. The metadata itself is stored by the file backed HTTP metadata resolvers as the SHA-1 hash of the EntityID and the extension ".xml". The local copy of the metadata ensures that the metadata is always available even if the TTP or the entity hosting the metadata cannot be reached.

The conversion rule synchronization mechanism of the IDP extension creates a backup of the relevant configuration files "attribute-resolver.xml" and "attribute-filter.xml". The downloaded conversion rule XSLT file is then applied to the "attribute-resolver.xml" file. The XSLT can lookup the XML id of other attributes it depends on to reference them properly. The "attribute-filter.xml" file is extended by a XSLT file, which is distributed with the IDP extension. It is used to limit the release of the converted attribute to the SP, which requested the attribute. After modifying these files, the related IDP components need to be reloaded for the changes to become active. The extension is able to do this without restarting the whole IDP.

C. SP Software

The SP extension is written for the SP module, which can be used with the Apache web server. It is very similar to the IDP extension. Because the SP is written C/C++, while the IDP and CDS are written in Java, there cannot be a joint extension for both. The SP only exists as version 2 at the moment, so there is only one extension. Special about the SP is that it consists of two modules, which need to communicate via inter process communication. One module is included as a library into the Apache web servers processes and the other runs as a standalone daemon. This prevents the Apache web server needing to load all libraries and their dependencies, which are required to parse and process the SAML messages. For that reason, the part included in the Apache web server is called "shibd_lite", whereas the daemon that processes the messages is called "shibd".

Because the SP extension is written in C/C++, it currently needs to be build on the target SP. Unless the administrator builds the SP from source, fetching the dependencies and compiling the extension can take some time. The extension's documentation contains a description of how to build it using Debian Linux.

The extension contains a communication endpoint for initiating the metadata exchange and a metadata provider for managing the downloaded metadata. The conversion rule part does not need to be implemented for the SP, as all attribute conversion is done by the IDP.

The communication endpoint of the SP does the same checks to prevent misuse as the IDP and then downloads the metadata to a file named after the SHA-1 hash of the EntityID. The DAME metadata provider is on the same level as the metadata provider of the IDP version 2 extension. It reads all available metadata files and has methods to dynamically add new files during runtime.

D. Evaluation of the Implementation

Figure 3 shows the general setup of the environment used to demonstrate the proof of concept. It consists of multiple

virtual machines that each were assigned specific roles and federations. On the one hand, the federation setup was used to determine the amount of work, which needed to be done for setting up a federation and to compare this to setting up the TTP, and, on the other hand, to test scenarios, where some providers were available right through the federation and others could be added dynamically using DAME.

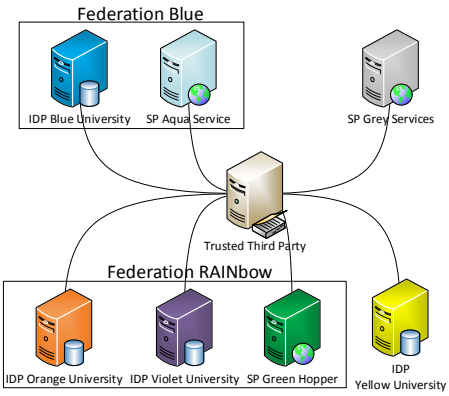


Figure 3. Overview of the proof of concept setup.

Building on the example given in Section III-A, one tested scenario included Blue University, Grey Services, and the Yellow University cooperating on the project COLORado. Marina from Blue University requests access to the SP of Grey Services, which runs a simple project collaboration tool. This tool should be used for sharing of project-related files, wiki web pages, group calendar, and has an integrated online Skype status check plugin. As both universities and the SP are not part of a common federation or inter-federation, they do not have each other's SAML metadata. Therefore, Marina chooses the federated login. Because the SP does not know her IDP, the Blue University, its embedded discovery service does not allow the selection of her IDP directly. As the organizations are set up for the GNTB, the SP's discovery service is configured to allow forwarding the discovery request to the GNTB DAME TTP. Marina chooses this option and is presented with a list of all IDPs currently available at the GNTB. After selecting her home IDP at the GNTB discovery service and subsequently authenticating there, Marina is redirected to the SP Grey Services. In the background, SP and IDP have exchanged each other's metadata and integrated it into the local configuration. A consent management tool, like uApprove, shows Marina the transmitted attributes and she needs to give her informed consent. After confirmation, she will be successfully logged-in to the collaboration tool. However, the integrated skype plugin does not yet contain Marina's Skype-ID, because the *skypeID* attribute could not be found. Marina informs her IDP administrator Azuro. Checking the SP's metadata, Azuro logs onto the GNTB web application and adds a new conversion rule that derives the *skypeID* attribute from *schacUserPresenceID*, which he knows is available at his IDP Blue University. With the conversion rule in place, Marina can use the Skype plugin as intended. We assume that at a later point in time, user Sunny from Yellow University tries to access the SP of Grey Services as well. Because both IDPs use *schacUserPrecenceID*, the attribute *skypeID* can automatically be created by re-using the attribute conversion rule from Blue University. Therefore, if Sunny chooses his IDP at the GNTB discovery service, triggering the metadata exchange. Based on the information in the SP's metadata, the correct conversion rule is downloaded and integrated into Sunny's IDP. He can directly use the Grey Services collaboration tool with the Skype plugin.

To evaluate the benefits of the GNTB extension, the test environment was setup as displayed in Figure 3. In order to measure how efficient the setup is, the manual steps for exchanging the metadata to setup the scenario are compared against the number of manual metadata exchanges needed to set up GNTB. Additionally, as an important metric it was determined how fast the metadata exchange actually could be done, as the aim was to do the exchange completely transparent to the user. The proof of concept implementation shows that the exchange can be done in under two seconds. However, it has to be noted that in this case all hosts are on the same virtual network and that real world usage would see more latency in the communication between servers. To ensure that the original authentication request from the SP is only forwarded to the IDP after the metadata has been exchanged and both providers have reload their configuration, a 10 second delay has been implemented. This should be more than enough time for the providers to finish reloading, while not being overly annoying to the user. The user also only needs to wait those extra 10 seconds if she is the first to use the specific SP–IDP combination. A more refined procedure, in which the current status is being periodically polled, will be added to minimize waiting times for users in real-world deployments, making the overall system more robust regarding latencies of any kind, including, e.g., delays due to insufficient Internet connectivity of mobile users.

To test how many manual metadata exchanges would be necessary, three different scenarios, in which GNTB could be used, are analyzed. The following description of scenarios and evaluation does not specifically include the amount of extra work that is required to install the necessary extensions at each provider. This amount of work is not specific to the DAME protocol but to its implementation. The installation and configuration of the DAME extensions could be heavily automated and is time efficient with the version 3 IDP. Table I summarizes the results.

1) Intra-federation: Within a federation GNTB could be used to exchange the metadata of the federation members. In large federations, this could improve scalability because only small subsets of identity and service providers ever need to exchange metadata and communicate with each other. The federation could deploy their own GNTB instance and reuse existing infrastructure, to get recent metadata files from their members.

To build a federation, like "Federation Blue", in the test environment, the members, i. e., Blue University, and Aqua Service, need to apply at the federation for membership and send their metadata to the federation. Both providers need to go through this procedure. To be more general, all n providers of a federation first need to register by sending their metadata to the federation. Afterwards, each provider must add the federation's metadata to its configuration, another n operations. In total $2n$ metadata exchange operations by all provider administrators. If the federation would be using GNTB, the number of operations would be less. Each provider has to register at the TTP (n

operations), but only the IDP providers would need to add the TTPs metadata to their configuration. If n_{idp} is the number of IDPs, a total of $n+n_{idp}$ operations would be needed to set up an environment with n providers. n_{idp} cannot be greater than n, the total amount of providers, thus $n + n_{idp} \leq 2n$ and the GNTB approach would reduce the overall work needed to be done to setup a federation. In the proof of concept environment shown in the figure $n = 2, n_{idp} = 1 : 2 + 1 < 2 \cdot 2 \rightarrow 3 < 4$

2) Inter-federation: Between multiple federations, the aggregated metadata file, which contains the metadata of all members, is even bigger than in federations, thus the amount of never used metadata is even higher. In this scenario, the federations could pass the metadata of their members to the GNTB instance of the inter-federation, which would be easier than all providers sending their metadata to their federation and the inter-federation. One possibility for growing federations and inter-federations is the use of a still to developed distributed GNTB, run by all participating federations.

Suppose the federations Blue and RAINbow want to join an inter-federation BlueRAINbow. The easiest way would be for the federation managers to send the federation metadata to the inter-federation GNTB instance and include the TTPs metadata in their own metadata distribution. This would be independent of whether the federations are using GNTB or the classic metadata aggregation. This would lead to $2i$ operations if i is the number of federations joining the inter-federation. From a technical view, this requires the same amount of metadata exchanges whether GNTB is used or not. Unfortunately, the TTP implementation does not support any bulk provider registration yet. This would be necessary if a federation would like to add all providers at once. With the current implementation $\sum_{x=1}^{i} n_x$ providers would need to register at the TTP and $\sum_{x=1}^{i} n_{idp_x}$ IDPs would need to integrate the TTPs metadata. This would result in $i = 2, n_1 = 2, n_2 = 3, n_{idp_1} = 1, n_{idp_2} = 2 : (2 + 3) + (1 + 2) = 8$ necessary metadata exchanges from the figure scenario.

3) No federations: Without any prior federations, each provider must register and upload its metadata to a GNTB instance itself, as described above in the COLORado example. As there is no existing infrastructure, it must be decided who runs a GNTB instance that can be used in that way. For example, larger projects or projects with much fluctuation of members could setup their own instance to manage the exchange of metadata between the members. This example uses the two federation-less providers SP Grey Services and IDP Yellow University as well as IDP Blue University to get a large enough test case.

Without a TTP and a federation, each provider would need to include everyone else's metadata and send its own metadata to everyone else, which would result in $2n(n-1)$ operations. With 3 providers, there are already $2 \cdot 3(3-1) = 12$ metadata exchanges. When a TTP is set up, this situation basically becomes the intra-federation case, which needs only $n + n_{idp} = 3 + 2 = 5$ metadata exchanges.

4) Summary of the Evaluation: Table I shows the manual metadata exchanges needed as described above. In the formula, n is the number of providers participating in building a federation, n_{idp} the number of IDPs in a federation, and i the number of federations joining a inter-federation. It is shown that in all cases, with the exception of the inter-federation case, using GNTB requires less manual metadata exchange operations and is, therefore, easier to maintain for administrators and quicker. The inter-federation case could be improved for GNTB to be equally good as the manual method by implementing the mass import of providers.

TABLE I. Comparison of manual and GNTB metadata exchange operations

	Manual	GNTB
Intra-federation	$2n$	$n + n_{idp}$
Inter-federation	$2i$	$\sum_{x=1}^{i} n_x + \sum_{x=1}^{i} n_{idp_x}$
No federations	$2n(n-1)$	$n + n_{idp}$

Because GNTB aims to make the metadata exchange more dynamic and remove the fixed structures of federations by using virtual federations, any combination of the scenarios above can be represented. A provider can be a member of multiple GNTB instances, so that it can be part of a project GNTB and of the GNTB of its federation and/or inter-federation. In order to reduce the amount of registrations, these distributed GNTB instances should cooperate. The then extended core workflow and the register of the GNTB instances still need to be developed. If the shortcut, which allows federations to add all their members in a bulk operation for use in an inter-federation scenario, is implemented, each test case is equal to or better than the currently used approaches with regards to the number of manual metadata exchanges necessary. This is also true in the case that a mix of the scenarios needs to be represented as this does not add any metadata exchange overhead. But not only the number of manual metadata exchanges is the same or smaller, the size of the metadata files is reduced as well. The inter-federation BlueRAINbow would, e. g., normally aggregate all metadata, which means at least 5 providers per metadata set. If, as an example, only IDP Blue University and IDP Orange University cooperate with SP Green Hopper the size contains with GNTB only 2 entities for the SP, while 1 for both IDPs.

VI. Risk Management

In preparation of pilot phase of the GNTB prototype and to achieve a technology readiness level TRL7, security-related questions have to be answered. Following existing good practices and international standards, e. g., ISO/IEC 27001, a risk assessment takes place. As presented in [14], we operationalize and support this continuous management process by applying our risk management template. Because GNTB allows the trust establishment between authentication and authorization infrastructure components, which are used to store, exchange, and process personally identifiable information by the user's IDP and an arbitrary SP, assuming that both are registered at the TTP, the criticality all of these have to be set to (very) high and implementing appropriate security measures is nearly unavoidable.

The first step in risk management, establishing the risk management context, requires the definition of primary and secondary assets. Primary assets are usually the core business processes and workflows of an organization. In this case, GNTB makes the immediate access to online services provided by previously unknown or untrusted SPs possible and thus could be seen as an enabler and innovator for the Research and Education community to collaborate and share data across

organizations and national or federation borders. The more technical, secondary assets support these processes and usually are categorized as the used hard- and software, the information exchanged and processed by the service's components. Operationalizing risk management focuses on the technical GNTB components: IDP and SP software extension, the TTP, the processed PII, and the exchanged metadata information and attribute conversion rules with all their dependencies to internal components like used databased, the underlying network infrastructure, libraries and operational details regarding the well-known objectives (confidentiality, integrity, and availability).

Possible events threatening GNTB's components are, e. g., flooding the TTP with metadata exchange requests. Describing such an event, using our threat scenario template, we have external actors triggering this harmful event, the threat type or category is malicious and the attacks aim at the violation of the availability in form of a complete service interruption and limitation of the services' usability. Another threat could be the download faked metadata information to compromise the IDPs or SPs and to lead them to trust each other and release sensitive user data to a malicious SP.

Assessing these example threat events, their likelihood as well as impact, especially the latter one from a privacy perspective, must be seen as high. The high risk value resulting requires further action. So, to overcome the first threat, GNTB requires a user authentication before the metadata exchange will be triggered as well as the SP can check due to sending back information about the user's IDP selection, if there was a previous access request, and, finally, an integrated rate limiter slows down the number of allowed requests to the TTP. Countermeasures to solve the second issue are, e. g., that the IDP checks if the source IP address of the metadata trigger message, as described in Section V-B, corresponds to that one configured by the IDP's administrator and prevents download metadata information from faked TTP instances; given that any transport is based on TCP/IP and successfully completed TLS handshakes, primitive attacks such as source IP address spoofing are not explicitly addressed here. Furthermore, the validation of the entity also prevents the registration of faked entities.

By listing all assets, analyzing the risks of all components and the dynamic metadata exchange itself, all possible risks were regarded. Based on the risks, possible attacker models and counter measurements, i. e., technical, organizational respectively preventive, detective, and responsive, were inspected. This lead to a protocol, which is as secure as possible, and to a secure designed GNTB TTP. Nevertheless, the local software and the TTP need to be monitored. Further risks can be mitigated by control by federation operators and the use of assurance frameworks.

VII. Conclusion

The DAME on-demand internet-scale SAML metadata exchange enables user-triggered exchange of metadata between IDPs and SP across current federations' borders. Furthermore, it enables the re-use of conversion rules, in order to further automate and accelerate the technical trust establishment. Last but not least, the scalability of the metadata exchange in federations and inter-federations is improved. The approach GÉANT-TrustBroker supports the fully automated technical setup of FIM-based authentication/authorization data exchange. Therefore, it increases the automation and scalability of former manual implementation steps by administrators. Consequently, the users can immediately use a new service.

While the DAME workflow allows the metadata exchange between IDP and SP, which are not part of a federation, but form a dynamic virtual federation, the federation management tool helps federation operators to formally establish federation, where official opt-in is required. The approach GÉANT-TrustBroker was implemented extending the SAML implementation Shibboleth and evaluated based on several scenarios. The implementation shows a scalable approach for SAML metadata exchange, where the duration of the metadata exchange is convenient for the end user. The amount of metadata exchanges is the same or smaller. At the same time the size of the metadata file is reduced. In order to have a secure service, the risk management was applied in Section VI and taken into account during the design of the GNTB. The current state of the protocol has been submitted as an Internet-Draft to the IETF to initiate a standardization process; a second implementation based on SimpleSAMLphp is currently being worked on. With its first international setup being deployed as a part of the eduGAIN service operated by the pan-European research and education network GÉANT, practical experiences with a large number of participating organizations and users will be gathered over the next few years.

Further research topics relate to the level of assurance respectively the trust between two entities. Though the technical trust is exchanged via the metadata, the quality of the entity could be assured or estimated by a level of assurance. As explained above, an abstract format for conversion rules would help to make these rules usable for different implementations. Furthermore, distributed TTPs should be investigated in order to have cooperating TTPs as it is not likely that only one TTP is operated worldwide.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement no 605243 (Multi-gigabit European Research and Education Network and Associated Services — GÉANT).

The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful comments on previous versions of this paper. The MNM-Team, directed by Prof. Dr. Dieter Kranzlmüller and Prof. Dr. Heinz-Gerd Hegering, is a group of researchers at Ludwig-Maximilians-Universität München, Technische Universität München, the University of the Federal Armed Forces, and the Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities.

References

[1] W. Hommel, S. Metzger, and D. Pöhn, "A SAML Metadata Broker for Dynamic Federations and Inter-Federations," in Proceedings of INFOCOMP 2014, The Fourth International Conference on Advanced Communications and Computation. IARIA, 2014, pp. 132–137.

[2] "SWITCH – AAI Resource Registry," 2015, URL: https://rr.aai.switch.ch/ [accessed: 2015-07-28].

[3] "PEER 0.20.0: Python Package Index," 2015, URL: https://pypi.python.org/pypi/peer [accessed: 2015-07-28].

[4] I. A. Young, "Metadata Query Protocol – draft-young-md-query-05," Work in Progress, 2015.

[5] A. Bhargav-Spantzel, A. Squicciarini, and E. Bertino, "Trust Negotiation in Identity Management," IEEE Security and Privacy, vol. 5, no. 2, Mar. 2007, pp. 55–63.

[6] P. A. Cabarcos, F. Almenárez, F. G. Mármol, and A. Marín, "To Federate or Not To Federate: A Reputation-Based Mechanism to Dynamize Cooperation in Identity Management," Wireless Personal Communications, 2013, pp. 1–18. [Online]. Available: http://dx.doi.org/10.1007/s11277-013-1338-y

[7] M. S. Ferdous and R. Poet, "Dynamic identity federation using security assertion markup language (saml)," in Policies and Research in Identity Management. Springer Berlin Heidelberg, 2013, pp. 131–146.

[8] W. Hommel, S. Metzger, and D. Pöhn, "Géant-TrustBroker: Dynamic, Scalable Management of SAML-Based Inter-federation Authentication and Authorization Infrastructures," in ICT Systems Security and Privacy Protection. Springer Berlin Heidelberg, 2014, pp. 307–320.

[9] W. Hommel, S. Metzger, and D. Pöhn, "Project GÉANT-TrustBroker – Dynamic Identity Management across Federation Borders," in Networking with the World, The 30th Trans European Research and Education Networking Conference, 19-22 May, 2014, Dublin, Ireland, Selected Papers. TERENA, 2014.

[10] N. Harris, "The Interfederation Problem," 2014, URL: https://blog.refeds.org/a/201 [accessed: 2015-07-28].

[11] "Shibboleth – Installation," 2015, URL: https://wiki.shibboleth.net/confluence/display/SHIB2/Installation [accessed: 2015-07-28].

[12] "SWITCH – Support – SWITCHaai," 2015, URL: https://www.switch.ch/aai/support [accessed: 2015-07-28].

[13] "SWITCH – uApprove – User Consent Module for Shibboleth Identity Providers," 2015, URL: https://www.switch.ch/aai/support/tools/uapprove [accessed: 2015-07-28].

[14] W. Hommel, S. Metzger, and M. Steinke, "Information Security Risk Management in Higher Education Institutions: From Processes to Operationalization," in Proceedings of the 21th congress of the European University Information Systems Organisation. EUNIS, 2015, pp. 190–201.

Assisting the Detection of Spatio-Temporal Patterns in Temporally Transformed Map Animation

Salla Multimäki and Paula Ahonen-Rainio
Dept. of Real Estate, Planning and Geoinformatics
Aalto University, School of Engineering
Finland
e-mail: salla.multimaki@aalto.fi, paula.ahonen@aalto.fi

Abstract— **Animated maps are widely used in visualizing the temporal aspect of geographical data, even though their effectiveness depends on multiple factors and is far from obvious. Especially when the temporal structure of a dataset is irregular, new methods for exploratory analysis are required. This paper presents a novel method to manipulate map animations by transforming the temporal dimension so that events are located in equal intervals into a time period. This temporal transformation applies the idea of spatial equal density transformation, which is familiar from cartography, in order to ease the cognitive load on a user caused by the temporal congestion of the dataset. A user test with a transformed animation of two different datasets indicated that the transformed animation is useful in revealing spatio-temporal patterns, which could not be detected from an original, untransformed animation. The transformed animations were found insightful and useful by the test users. These findings indicate that a temporally transformed animation would be a useful addition to a toolbox for exploratory analysis of spatio-temporal data.**

Keywords-map animation; cognitive load; temporal transformation; equal density; user testing.

I. INTRODUCTION

This paper is extending the previous study of temporal transformation [1] with more thorough review of cognitive load and a task definition of exploratory analysis.

A map animation is a common method for visualizing spatio-temporal information. The reason for this is simple: an animated map follows the congruence principle [2], which achieves the natural correspondence between the subject and its representation, by allowing spatial information to be presented on a map and, at the same time, using the temporal dimension for presenting changes over time. However, an animated map does not automatically result in effective comprehension. The comparison between animations and static visualization methods has been considered in many studies, both in computer graphics [2] [3] and in cartography [4] [5] [6]. The results from these studies show variation in the superiority of the methods, depending on the types of tasks and datasets.

Fabrikant et al. [7] pointed out that well-designed map animations are inherently different from well-designed static maps and comparison between these two methods is not even meaningful. They argued that instead the attention should be focused on studying when and why these methods do work. Lobben [8] showed that animations are better suited when users' tasks deal with time, and also found some evidence that static maps could work better with location-based tasks. Simultaneous changes at many discrete locations are difficult to perceive from an animation but as a presentation of general spatio-temporal patterns and the behaviour as a whole an animation can be most valuable [9].

Harrower and Fabrikant [9] argued that passive viewing of an animation, without any control tools, is more valuable in an early phase of an analysis, offering an overall picture of the phenomenon. It is evident that in later phases user control may increase the usability of an animation and the effectiveness of its comprehension. In many cases, user control improves users' performance [10], but at the same time, it may distort the continuity of the animation so much that the advantages of the animation are lost [9]. The use of control tools always produces a split attention problem and increases the users' cognitive load [11].

These inefficiencies of traditional animations suggest that in order to support the interpretation of map animations we should search for methods to overcome these issues. These kinds of methods would especially be required in the exploratory analysis of spatio-temporal datasets with an irregular temporal structure containing long still periods and/or dense temporal clusters.

Especially with temporally irregular datasets, the ability to control the animation becomes essential. Without any control, such an animation would be inefficient and dull, and sudden changes could easily be missed [9]. Dykes and Mountain [12] argue that animations that present time linearly and continuously are not suitable for all kinds of exploratory analysis tasks, and call for more advanced visualization methods.

The rest of the paper is organized as follows: Section II motivates the study and Section III discusses the cognitive load of animated maps. Section IV presents the equal density transformation and Section V covers the user test and interviews. The results are presented in Section VI, after which these results are discussed in detail in Section VII. Finally, conclusions are drawn in Section VIII.

II. MOTIVATION

The dynamic visual variables duration, order, and rate of change [13] make it possible to present different spatio-temporal datasets meaningfully in an animation. Typically, the duration of the scenes is kept constant throughout the

whole animation, although Harrower and Fabrikant [9] provide a reminder about the possibility of modifying the duration of the scenes dynamically during the animation. In animations that present changes over time, the order of the scenes must be chronological. The rate of change depends on the sampling rate of the real-world phenomenon in the dataset. The duration and rate of change together produce the perceived speed of the animation.

Monmonier [14] gave examples of how to apply spatial generalization operators such as displacement, smoothing, and exaggeration to the temporal dimension of dynamic statistical maps, but his motivation was to avoid incoherent flicker and twinkling dots and instead allow the perception of salient patterns in dynamic thematic maps. Kraak [15] experimented with the transformation that he calls "from time to geography", for static presentations. It stretches the famous Minard's Map according to the temporal dimension in such a way that the time periods with slow or no movement stretch the spatial representation of the trajectory, and, correspondingly, fast movement is shrunken into shorter. This example differs from so-called travel time cartograms since it does not consider any location as a reference point to which the travelling times are calculated. Andrienko et al. [16] presented a time transformation called a "trajectory wall", where the time axis of a space-time cube is modified to present the relative order of the events. Therefore, trajectories that follow the same route do not overlap with each other, but form a wall in which the order of the trajectories on the time axis is determined by their starting times.

With static maps, cartographic transformations, such as density transformation, are used to present a phenomenon from different perspectives and to reveal patterns that would otherwise stay hidden. This inspired us to study the possibility of benefiting from a similar transformation of the temporal dimension in animated maps.

In this study, our aim was to ease the interpretation of temporally irregular datasets by presenting the data from a different perspective and with a different temporal emphasis. For the study, we created transformed map animations of two different datasets with different temporal structures. As a result of the transformation, the events recorded in a dataset were evenly located in time over the whole time period, keeping their order. The basis of this transformation is somehow similar to a trajectory wall [16], since it preserves the order of the events, but it is implemented in a dynamic display. We call this transformation as "temporal equal density transformation" because it equalizes the time intervals between consecutive events. Our hypothesis was that this kind of transformation could support the analysis of spatio-temporal phenomena. We performed concept testing and interviews with users to study the potential of temporal equal density transformation in detecting spatio-temporal patterns.

III. COGNITIVE LOAD OF MAP ANIMATIONS

In this section, we look at those parts of cognitive loads that are relevant for this research. This is to demonstrate the effect of human cognitive processes when studying irregular spatio-temporal datasets.

The human brain collects information from the environment through several information channels and processes this data in working memory. The number of objects that the working memory can handle at one time is limited; estimate of this differ from 7±2 [17] to only three or four [18]. From working memory, we move processed information to long-term memory, which has practically unlimited capacity. The research on this moving process has developed cognitive load theories [19].

Theories of cognitive load are widely accepted and used, but in the context of this study it is meaningful to specify which part of the data visualization causes an increase in the cognitive load. Paas et al. [19] divide the cognitive load into three parts: the intrinsic, extraneous, and germane cognitive loads. The intrinsic cognitive load is caused by the data itself, the extraneous cognitive load means the unnecessary information caused by the visualization or user interface, and the germane cognitive load deals with the presentation of the task and even the motivation of the user to fulfil the task. If the data contains thousands of separate objects, we must, in one way or another, enable all these objects to be handled in the working memory. In cartography, this intrinsic cognitive load is usually handled by means of generalization or classification of the data. For decades, cartographic visualization has aimed to reduce the extraneous load of unnecessary information by selecting, filtering, and emphasizing the relevant parts of the information. Clarifying the task of the user and presenting the information in a feasible way are essential in order to reduce the germane cognitive load.

Alternatively, we can discuss the "load on the working memory" or simply "mental load". Mayer and Moreno [20] use the term mental load when presenting a scenario in which both information channels, verbal and visual, are overloaded. As a solution for this kind of overload Mayer and Moreno suggest segmenting the information into parts. The principle is that enough time must be given for the user to process one part of the information before the next part is presented. This processing time can be pre-defined or the user can give a signal (for example, by clicking "continue") that he/she is ready to move on.

With animated map visualizations this segmenting means that the duration of the scenes [13] must be selected to be such that the user can perceive all the important elements on the map. The more elements that are visible, the more time we need to perceive them all. However, as mentioned earlier, the temporal structure of the dataset can have such a nature that simply slowing down the animation is not reasonable. Therefore, we must search for other ways to offer the time needed for processing.

IV. EQUAL DENSITY TRANSFORMATION

The spatial and temporal dimensions of geographic data share commonalities, such as scale and its relation to the level of details of the phenomena that are represented [21]. When we present a real-world space on a map, we shrink the presentation into a smaller scale and, in most cases, explicitly inform the users about the scale either by number or a line. In

a map animation, real-world time is usually correspondingly scaled down into a shorter display time. Despite the fact that the temporal scale, just like the spatial scale, has a strong influence on the observation and understanding of the phenomena, the temporal scale is not commonly calculated and expressed numerically in an animation. Instead, the passing of time is presented as a relative location of a pointer on a time slider. This time slider often works both as a temporal legend and a control tool.

In addition, the spatial and temporal dimensions have some similar topological and metric relationships. The temporal topological relationships presented by Allen [22] show many similarities to spatial data: moments in time can be ordered, and temporal objects can be equal to, meet, overlap with, or include each other. However, because of the one-dimensional nature of time, the temporal order is unambiguous and each point object can have only two neighbours in time: the one that is the closest before and the one after. The only temporal metric relationship is the length of time (the duration of an event or an interval between two events), corresponding to distance in space.

The idea of many of the geographic transformations, such as equal density transformation [23] and fish-eye zooming [24], are adaptable to the time dimension, while some other transformations cannot strictly be applied to the time axis because of the one-dimensional nature of time.

Spatially, in equal density transformation the areas of high density of the phenomenon are made bigger and the areas of low density become smaller, so that the spatial density of the phenomenon becomes constant [23]. This transformation is presented in an example in Figure 1. The distances between points are equalized and the reference grid stretches and shrinks correspondingly.

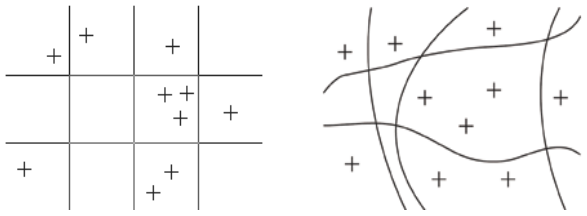


Figure 1. A small spatial example dataset (left) and the effect of spatial equalized density transformation to it (right).

In the temporal version of equal density transformation that we study, the time intervals between each two consecutive events are equalized in length over the whole time period. Equal density transformation is performed by counting the number (N) of events (E) in the dataset in the time period (P), which is the time between the time stamps of the first (t_0) and last (t_n) event of the dataset, and then calculating the new time stamps t' for each E. Each event E gets its own portion equal length of the time period and is placed in the middle of its portion.

$$t'(E_i) = t_0 + (i - 1/2) * (P/N)$$

If the dataset contains events that feature duration, their start and end times are simply handled as separate events on the timeline. In this transformation, the accurate timestamps of events are lost, but the temporal topological relationships remain constant. Should there be any events with exactly the same timestamp, their mutual order must be determined by some other attribute or one must diverge from the principle of equality and present those events at the same time.

An example of a set of temporal events in its original form and after the equal density transformation is shown in Figure 2. Events are presented in changing colour, emphasizing their order. In Figure 2, it can be seen that the degree of transformation reflects the density of the events. When the events are condensed, time around them is stretched to last longer. Consequently, time periods with sparse events are speeded up. This reduces the user's temptation to fast-forward those periods that might cause some potentially important information remaining unobserved.

The temporal equal density transformation follows the segmenting solution suggested by Mayer and Moreno [20]. It gives enough time to perceive every single event and, simultaneously, eliminates the unnecessary empty periods, which would grow even longer if the animation were slowed down traditionally. Therefore, it avoids composition of artificial groups, which can happen when segmenting the time into equal periods. Segmenting the time dimension into periods of equal duration and presenting all the events in one period at the same time is a method used when animating maps. The aim is to reduce the required computer capacity and the cognitive load on the user. However, this segmenting causes a common problem: if the dataset is artificially divided into even time periods, it is possible that temporal clusters in the dataset get unintentionally chopped. On the other hand, events that end up in the same period are automatically grouped together even though they can actually be temporally rather far from each other.

It must be underlined that like spatial equal density transformation, this temporal transformation can serve different purposes. Spatial equalization can be used to show the relative importance or weight of areas with different densities of objects, or for the closer examination of dense clusters. Similarly, temporal transformation can make all the events visible and reveals the temporal order of the data. At the same time it reduces the worthless empty periods. This feature is more important than in static spatial visualizations, where the user can simply ignore the areas with no events. If temporal equalization were used to show the importance of different periods, the visualization of the timeline, as in Figure 2, would play a critical role.

V. USER TEST AND INTERVIEWS

In this section, we first describe the datasets and the test animations, the test setting and interviews, and finally the analysis of the results.

The aim of the user test was to find out whether the transformation would support the recognition of spatio-temporal patterns in the analysis process. Therefore, we prepared a set of test animations and questions that the test users were to answer while viewing the animations. The number of times they viewed each animation and the additional comments they made during the test were recorded.

A. Test Data and Animations

The dataset used in this test contained Twitter messages, so-called tweets, from the area of Port-au-Prince, Haiti, from a four-month period after the earthquake in January 2010. Twitter was used to search for help and food or water supplies and also to find missing persons. A Twitter user can allow the exact coordinates of the tweets to be saved and shown by the service provider, and all the tweets with these coordinates were included in the test dataset.

For this test, two different datasets were prepared. The first dataset covered the four-month period after the earthquake, but to keep the size of the dataset reasonable, only every tenth tweet was selected. In this dataset, most of the tweets were strongly compressed into the first days and weeks of the time period, and after that the density of the tweets decreased remarkably. The densest period was around January 22nd. This dataset is referred to as the "Every 10th" dataset and is shown on a timeline (a) in Figure 3. The other dataset contained the very first tweets right after the earthquake. To achieve the same number of objects (193 tweets), the dataset was cut to cover about an 84-hour period. Because of the problems in electricity production in Haiti, only those tweets that were sent between 6 am and 6 pm were successfully published. This caused strong periodicity in the data. This dataset is referred to as the "First days" dataset and is shown in timeline (b) in Figure 3. Spatially, the events of the "First days" dataset are clustered more into the centre of Port-au-Prince, while the events of the "Every 10th" dataset are spread more evenly over the area of the city (Figure 4). It must be emphasized that the users never saw these datasets side by side as in Figure 4, because they examined only one dataset at the time.

Both datasets were equal density transformed by using Microsoft Excel. In the transformed datasets, the time interval between two consecutive events is the whole time period divided by the number of events. Timeline (c) in Figure 3 shows the effect of the equal density transformation; these two datasets become similar. The timestamps are not visible in this timeline, because they were artificially modified and did not correspond to the real-world time.

Four map animations were made with ArcGIS10; two presented the original "Every 10th" and "First days" datasets and two presented the equal density transformed datasets. All animations were of equal length, 60 seconds, and they each contained 193 events. The events were presented on a background map with red dots that appeared brightly and faded to a less saturated red after that. In addition to the animations, the timeline visualization was presented in the test view immediately below the test animation (Figure 4) to help the test users to comprehend the temporal patterns of the data.

The effect of the transformation to the animation of the dataset "Every 10th" can be seen in Figure 5. In the upper figure, both animations are paused after 15 seconds. In an original animation (left), majority of the events are already appeared while the events in the transformed animation (right) run slower. In the lower figure, where the animations are paused after 45 seconds, this difference is almost tied.

In the "Every 10th" dataset, the events are congested into the first days and weeks after the earthquake, the most dense period being around January 22nd. The "First days" dataset shows that there were no events between 6:00 pm and 6:00 am, and that there were relatively few events on the very first day after the earthquake.

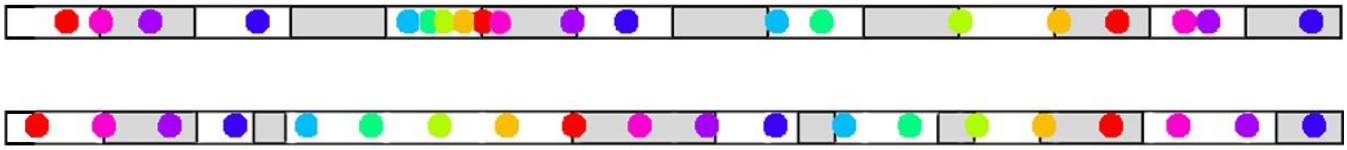


Figure 2. An example showing temporal events on a timeline (above) and the same dataset after the temporal equal density transformation (below). Grey and white areas behind the event points indicate the time units; in the upper timeline they are all of equal length, but after the transformation those time units with many events stretch and those with fewer or no events shrink.

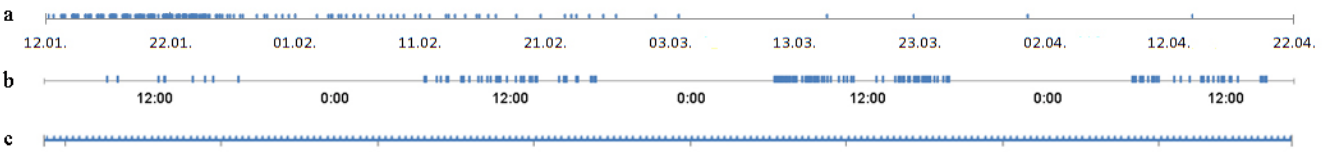


Figure 3. The top row (a) shows the "Every 10th" dataset visualized on a timeline. The middle row (b) shows the "First days" dataset visualized on a timeline. Note that the scales of the timelines are different. The bottom row (c) shows both datasets after the equal density transformation.

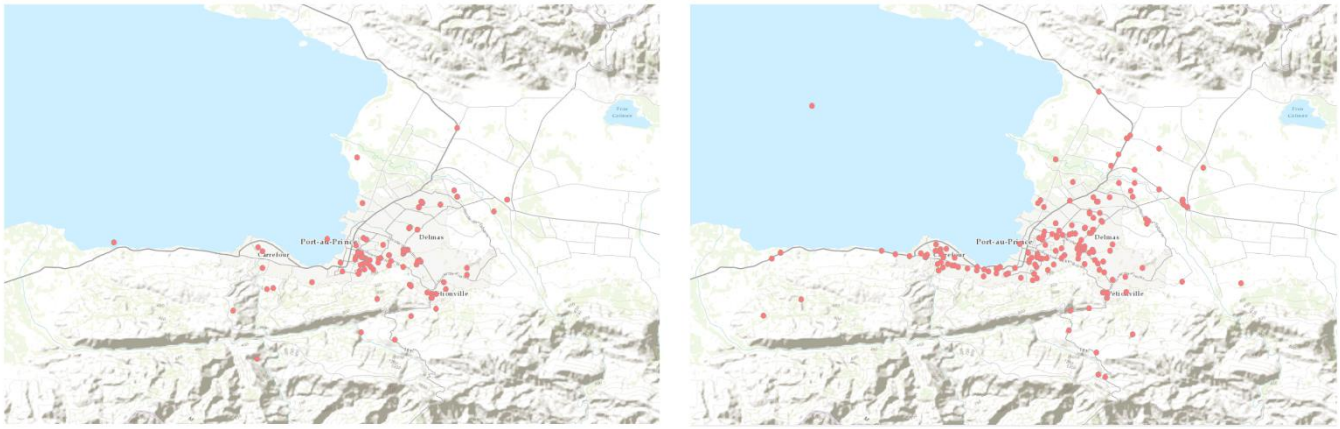


Figure 4. Spatial distribution of the events of both datasets used in the user test, "First Days" (left) and "Every 10th" (right). During the first days, there were very few events from outside the urban area of Port-Au-Prince.

B. Test Setting

Main concepts and terminology of the test, such as *pattern* and *spatio-temporal information*, were introduced to the user in the beginning of the test. Then the user was able to familiarize himself/herself with an example animation (a 10-second clip showing a zoomed part of one of the animations), layout, and arrangements of the user interface. The test contained two parts. One part presented the original and transformed animations of the "First days" dataset and the other part the corresponding animations of the "Every 10th" dataset. Each test user performed both parts, but the order of these parts varied between the users in order to avoid the influence of the learning effect. These two parts were identical in terms of their layout and arrangements.

In the first phase, the user first had the opportunity to view only the original animation as many times as he/she wanted, and after that was asked to answer Questions 1.1 and 1.2 (Table I). The questions dealt with the overall impression of the dataset. Then the user viewed the temporally transformed animation and answered the same questions. The order of the animations was fixed to this, because we wanted to simulate the explorative analysis task where the user first gets an overview of the data and then uses more complex tools, focusing on more detailed analysis.

In the second phase, the user could use both of these two animations to answer three more detailed questions (Questions 2.1-2.3 in Table I) about the behavioural patterns of the data. Because of the differences in the datasets, the questions varied slightly between the two datasets. After finishing both parts of the test, the user was interviewed. The interview was semi-structured and covered the following topics:

- Was the temporal transformation as a method easy to understand?
- What could have made the transformation easier to understand?
- Did you use the animation, timeline, or still picture of the animation to answer the questions?
- Was the transformed animation useful when answering the questions? Why?
- In what tasks was the transformation especially useful?
- Could this kind of tool be useful in your job?

Some users had already discussed these topics during the test, and in these cases not all the questions were explicitly gone through during the interview.

The test was completed by nine users. They were professional cartographers or geographers with experience of temporal datasets. Four of them were female and five male. Their ages varied between 28 and 55 years.

The users did the test on a laptop computer that was connected to a data projector. The evaluator observed the user's performance via the data projector, calculating the viewing times for each animation. The users could answer the test questions either by typing their answers into a textbox on the display or verbally to the evaluator, who wrote those answers down. The answers in the interviews were also written down by the evaluator.

TABLE I. QUESTIONS IN THE USER TEST.

Q	"First days" dataset	"Every 10th" dataset
1.1	What kind of patterns do you find from the data?	
1.2	Are there any events which seem not to fit the data or draw your attention in some other way?	
2.1	Where are the first and last events of the dataset located?	In what area are the first ten events of the dataset located?
2.2	Is there a location on the map where there are multiple sequential events? Where is it?	Are there time periods when the events are clustered into a certain area?
2.3	Is there an area on the map where the events are clustered both spatially and temporally?	Does the centroid of the events move during the animation?

C. Analysis of the Material

From the user test, the following indicators were analysed:

1. the number of times the user viewed each animation in each task, and whether he/she viewed the whole animation or was the viewing discontinued;
2. the kinds of behaviour patterns the user found from the animations and whether these findings were appropriate;
3. whether the user's impression of the phenomenon represented in the dataset differed between the original and transformed animations.

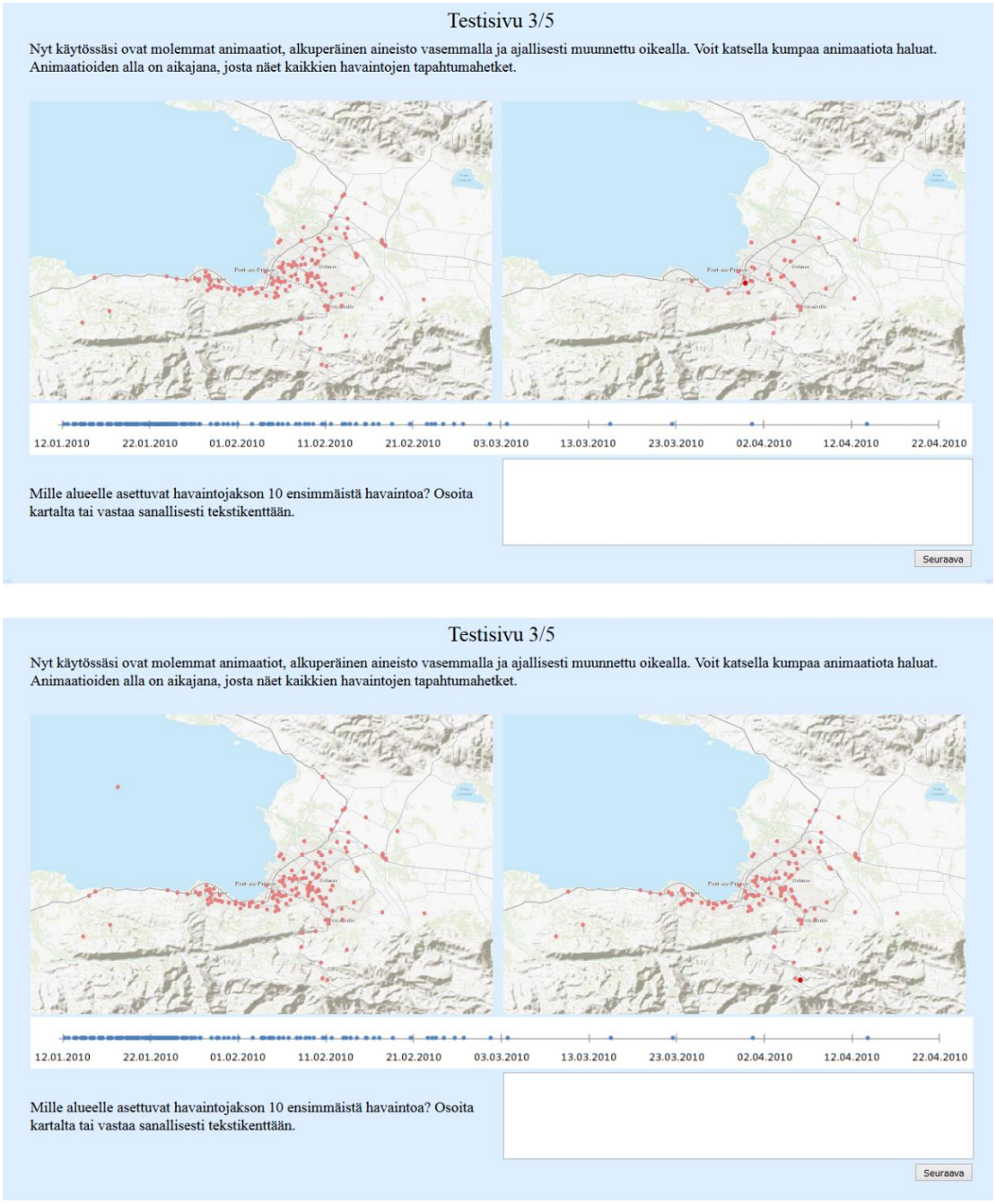


Figure 5. The test setting and the effect of the transformation. The original animation of "Every 10th" dataset is on left and the transformed animation of the same dataset is on right. Above the animations is a short instruction text. Under the animations is the timeline of the dataset showing its temporal structure. On bottom left is the task question, and on bottom right is the text box in which the user can write the answer. On upper figure, both animations are paused after 15 seconds. On lower figure, the animations are paused after 45 seconds.

From the users' comments during the test and their answers in the interview, the following factors were calculated:

1. positive and negative comments the users made about each animations;
2. faulty and inaccurate interpretations that the users made from the animations;
3. cases where the user made different interpretations from the same dataset on the basis of the two animations.

Because of the small number of test users, no statistical significance parameters were calculated from these results.

VI. RESULTS

In the test, the users chose to view the transformed animation slightly more often than the original animation. This trend was particularly clear with Questions 2.1 and 2.2, which dealt with so-called elementary lookup tasks [25]. With more complex analysis tasks, the users tended to interrupt the flow of the original animations by pausing or fast-forwarding, while the transformed animations were more often viewed in their entirety. This pattern can be seen in Table II. The first, **bold** number in each box marks the times when the animation was viewed completely, and the second number (in parentheses) marks the times when the animation was viewed partially, which means that the user paused, fast-forwarded, or interrupted it in some other way during the viewing. Every row in the table corresponds to one task in the test.

The differences between the results for the two datasets in Question 2.3 are caused by the difference in the questions. With the "First Days" dataset, the question concerned the whole time period, and, therefore, the users had no choice but to view the animation completely. On the contrary, with the "Every 10th" dataset the task was to find a spatio-temporal cluster, and the users could stop viewing the animation after finding the first one.

TABLE II. THE VIEWING TIMES OF EACH ANIMATION IN THE USER TEST

	Every 10th dataset		**First Days dataset**	
	original	***temporally transf.***	***original***	***temporally transf.***
Q 1.1 and 1.2	**16**(2)		**16**(2)	
Q 1.1 and 1.2		**15**(2)		**17**(0)
Q 2.1	**2**(10)	**3**(14)	**5**(6)	**3**(12)
Q 2.2	**4**(5)	**9**(2)	**6**(1)	**11**(0)
Q 2.3	**6**(10)	**9**(4)	**9**(1)	**8**(0)

When viewing the transformed animation of the "First days" dataset, in the first phase of the test six out of the nine users mentioned that they perceived a location at which several events appeared sequentially. This location can be seen as the latest and brightest dot in the upper-right map of Figure 5. Later, when explicitly asked (Q2.2), the remaining three users also perceived it. From the original animation, none of the users perceived this kind of behaviour at first, and after the question Q2.2, only one user mentioned that he also saw that phenomenon in the original animation, "but much more weakly than in the transformed animation". This location with sequent events proved to be a police station in the centre of Port-au-Prince. Our assumption is that the inhabitants of the city went to the police station to seek their missing relatives after the earthquake, and the police used Twitter to support the search and rescue efforts.

The users learned to favour the transformed animation with some tasks even during this kind of very short test. For the questions Q2.1 and Q2.2 the transformed animation was used approximately 40 % more often than the original, and was viewed more often without interruption. With the question Q2.3 the preference between the animations varied, depending on the dataset, but the transformed animation was viewed in its entirety more often.

In the interviews most of the users had a positive attitude towards the transformed animations. They were apparently pleased and said that the transformation was "charming" or "nicer". They also mentioned that the transformed animation was "better" and "easier to watch". Two users said that the transformed animation was "exhausting" and its "continuous info flow was tiring". However, these two users also commented that the transformation was useful in some cases. A summary of elements calculated from the interviews is shown in Table III.

TABLE III. SUMMARY FROM THE INTERVIEWS

	Original animation	**Temporally transformed animation**
More useful (pos.)	2	14
Unpleasant to view (neg.)	1	2
Misinterpretations	Not applicable	2
Different interpretations	8	

The users had varying opinions about the applicability of the transformation. For example, when the task was to find spatio-temporal clusters (Q 2.3), some of the users said that the transformed version was "essential", while others said that it did not suit those tasks. When asked about the use cases for this kind of transformation, the users mentioned several possible application areas in addition to elementary lookup tasks dealing with time. For example, traffic planning, crowd movement analysis, environmental analysis, and oil destruction activities were proposed. The users also pointed out the possibility of combining the analysis of the temporal dimension on the timeline with behavioural analysis of the transformed animation.

From the interviews it became clear that a proper temporal legend could have improved the performance; five of the nine test users mentioned this when asked about development ideas. More specifically, the idea of colouring

the events according to their timestamp was mentioned by several users. Another suggestion was to improve the linking between the timeline and the animation; a moving pointer should show the flow of time of the transformed animation on the timeline of the original data.

The test results indicate that it is essential to ensure that the user understands how the transformation influences the animation. In several cases (eight out of the 27 behaviour descriptions recorded) the users' impression about the phenomenon varied between the original and transformed animations, even though they knew that the animations presented the same dataset. Some clear misinterpretations appeared; in one case the user grouped the last events of one day and the first events of the next day into the same spatio-temporal cluster despite the fact that there was a 12-hour gap. The same user also made a false statement about the location of the last event of the dataset.

VII. DISCUSSION AND FURTHER RESEARCH

The results presented above are discussed from two different perspectives: usefulness and limitations of the temporal equal density transformation that was tested. These considerations are followed by the evaluation of the study.

A. Usefulness of the Tested Transformation

The user test shows that temporal equal density transformation of the animation revealed the location of sequential events that were not detected from the original animation. These sequential events were temporally so close to each other that, without the transformed animation, this pattern would have remained unnoticed by most users. Additionally, this pattern was found spontaneously; in the majority of cases it arrested the attention of the users without any specific search task. This kind of capacity is an important feature of the visual analysis tool when it is used for data exploration.

As the test results and interviews with the users suggest, the power of the temporal equal density transformation lies in the fact that it seems to reduce the user's need to interrupt the animation, and therefore offers a smooth overall evocation of the phenomenon. At the same time, it eases the cognitive load on the user by offering a continuous, temporally predictable change with no congested periods. It emphasizes the order of the events and equalizes them in relation to time, thus attributing equal significance to all the events.

B. Limitations of the Tested Transformation

The findings indicate that the disadvantage of the transformation is that misinterpretations of the effect of the transformation are possible, even probable. These misinterpretations could be reduced with a temporal legend in which the user can always perceive the phase of the animation. For a more sophisticated approach, we suggest two possible solutions for this problem; segmenting and colour.

The equal density transformation itself is simultaneously in line and contradiction of the principle number 2 of Mayer and Moreno [20]: "Provide pauses". It gives the user time to recognize each event separately, but lacks the longer pauses needed for processing the information in order to move it into long-term memory. This was also found from the user feedback on the test: one user stated that the animation was "exhausting". Our suggestion is to add longer pauses to mark natural time periods in the data. These pauses would work for two purposes: they give time for the user to process the information more deeply, and separate the natural time periods from each other. These time periods could be, for example, days, but the phenomenon can also have such a nature that using midnight to divide the periods can cause the splitting of temporal clusters. This solution would be suitable for bigger datasets, if the events of one period were being processed as one mental chunk with no intention to understand more than the overall behaviour of the phenomenon.

Furthermore, properly designed colours for the events could indicate the degree of the transformation, and these colours could be used for linking the timeline and the animations. The idea of colouring the events according to their timestamp also arose repeatedly in the interviews. The users suggested, for example, that the colour of the events might change smoothly day by day. This solution would help the user to detect spatio-temporal clusters from the map and it also communicates about the temporal discontinuities in the data.

However, a disadvantage of the use of colour as a temporal legend is that one cannot visualize any attribute data by means of colours at the same time. When the events are point-type and visualized with small, round objects, other ways to present attribute information are limited. Therefore, consideration should be given to whether the combination of these two variables with the use of colour is possible. Brewer [26] proposed a set of colour scheme types to be used with bivariate data, but this set does not contain the combination of qualitative attribute information and a bipolar subordinate variable (the temporal transformation can affect the time either by stretching or shrinking it, and, therefore, its visualization should be bipolar). If the degree of temporal transformation is simplified to binary data (= slower or faster than the original), then Brewer's "qualitative/binary" combination can be used. In this schema, the qualitative data is visualized with different hues and the binary data is visualized with the lightness of the colour. It must be noted that Brewer's model is designed for choropleth maps, and its applicability to point-type data is not obvious. Therefore, it is clear that more research is needed to test whether discrimination between these two variables is possible in a use case similar to the case in this study.

Another possible way to provide the information about the speed of the animation is sound. Kraak et al. [10] suggest sonic input to represent the passing of time. This could also be a useful technique with an animation with changing speed,

since human hearing is relatively sensitive to changes in rhythm and pitch.

C. Evaluation of the Study

In this study, we tested the concept of temporal equal density transformation and therefore wanted to keep the test arrangement as simple as possible. The influence of attribute information was not evaluated in this study; therefore, we suggest that the method studied here is better suited to pre-processed data where the selection of relevant events is performed beforehand and the interest of the analyst is in the spatio-temporal behaviour of the data. However, this method does not rule out the possibility of classifying the events according to their thematic content and visualizing this classification by colour, if colour is not used to present the flow of time (as suggested earlier).

Because of the simplicity of the test procedure, the user control over the animations was limited. The users did not have a chance to adjust the speed of the animation nor to filter its content. However, we offered the most common user control tools; playing, pausing, and the opportunity to jump to any moment in the animation. A wider selection of control tools might have increased the cognitive load on the user and drawn the user's attention away from the task being tested.

Finally, we want to emphasize that temporal equal density transformation, in the form that was tested here, is suitable only for relatively small datasets because it shows every event individually. Theoretical maximum number of events can be calculated by multiplying the minimum time of each eye saccade and attentional blink caused by the perception time of one event (together they take approximately 300 ms) with the maximum length of working memory without any revision (20 seconds). With this calculation we suggest that no more than 60-70 events should be presented consequently without pausing. Additionally, the method was only tested with two datasets with different temporal structures: one with decreasing intensity of events and the other with clear variation of empty and dense periods. More tests are needed to study the usefulness of the method with different spatio-temporal datasets.

VIII. CONCLUSIONS

The human ability to adopt information from an animated map is limited. If the animation runs too fast, is too long, or presents too many events simultaneously, a user can easily miss some information, and, therefore, is not able to form a full image of the phenomenon being presented. The traditional control tools of an animation, such as pausing, jumping to a specified scene, or looping, have a limited capability to improve this understanding.

This paper presented a novel method for equal density transformation of the temporal dimension of map animations by equalizing the time intervals between each two consecutive events. The user test showed that the transformation can reveal patterns that would have been left unnoticed with traditional animation. Transformed animation in parallel to an original, untransformed animation seems to be understandable for the users and useful for spatio-temporal analysis.

In exploratory analysis a rich variety of tools that complement each other is a necessity. The results from this user test and interviews indicate that temporal equal density transformation might be an appropriate technique to complement a set of such analysis tools. Our suggestion is that temporal equal density transformation should be used for those datasets that a) have an irregular temporal structure and b) are small enough to be able to be examined individually. The transformed animation could, if reasonable, be segmented so that the natural periods in the dataset are separated from each other with a longer pause. As a result of these kinds of improvements the transformed animation, complemented with an original, untransformed animation to offer an overall image of the phenomenon, would complete the toolbox of spatio-temporal exploratory analysis.

REFERENCES

[1] S. Multimäki and P. Ahonen-Rainio, "Temporally Transformed Map Animation for Visual Data Analysis," The Seventh International Conference on Advanced Geographic Information Systems, Applications, and Services (GEOProcessing 2015), C.-P. Rückemann and Y. Doytsher (eds.) IARIA Feb.2015, pp. 25-31, ISSN: 2308-393X, ISBN: 978-1-61208-383-4.

[2] B. Tversky, J. B. Morrison, and M. Betrancourt, "Animation: can it facilitate?" International Journal of Human-Computer Studies, vol 57, 2002, pp. 247-262, doi:http://dx.doi.org/10.1006/ijhc.2002.1017.

[3] D. Archambault, H. C. Purchase, and B. Pinaud, "Animation, small multiples, and the effect of mental map preservation in dynamic graphs," IEEE Transactions on Visualization and Computer Graphics, vol. 17(4), pp. 539-552, 2011, doi:http://dx.doi.org/10.1109/TVCG.2010.78.

[4] A. Koussoulakou and M.-J. Kraak, "Spatio-temporal maps and cartographic communication," The Cartographic Journal, vol. 29(2), pp. 101-108, 1992, doi: http://dx.doi.org/10.1179/caj.1992.29.2.101.

[5] T. S. Slocum, R. S. Sluter, F. C. Kessler, and S. C. Yoder, "A qualitative evaluation of MapTime, a program for exploring spatiotemporal point data," Cartographica, vol. 39(3), pp. 43-68, 2004, doi:http://dx.doi.org/10.3138/92T3-T928-8105-88X7.

[6] A. Griffin, A. MacEachren, F. Hardisty, E. Steiner, and B. Li, "A comparison of animated maps with static small-multiple maps for visually identifying space-time clusters," Annals of the American Cartographers, vol. 96(4), pp. 740-753, 2006, doi:http://dx.doi.org/10.1111/j.1467-8306.2006.00514.x.

[7] S. I. Fabrikant, S. Rebich-Hespanha, N. Andrienko, G. Andrienko, and D. R. Montello, "Novel method to measure inference affordance in static small multiple displays representing dynamic processes," The Cartographic Journal, vol. 45(3), pp. 201-215, 2008, doi:http://dx.doi.org/10.1179/000870408X311396.

[8] A. Lobben, "Influence of data properties on animated maps," Annals of the Association of American Geographers, vol. 98(3), pp. 583-603, 2008, doi:http://dx.doi.org/10.1080/00045600802046577.

[9] M. Harrower and S. Fabrikant, “The role of map animation for geographic visualization,” In: M. Dodge, M. McDerby, and M. Turner (Eds.), Geographic Visualization. John Wiley & Sons, 2008, doi:http://dx.doi.org/10.1002/9780470987643.ch4.

[10] M.-J. Kraak, R. Edsall, and A. MacEachren, “Cartographic Animation and Legends for Temporal Maps: Exploration and or Interaction,” In: Proceedings of the 18th International Cartographic Conference, Stockholm, Sweden, 1997, pp. 253-260.

[11] M. Harrower, “The cognitive limits of animated maps,” Cartographica, vol. 42(4), pp. 349-357, 2007, doi:http://dx.doi.org/10.3138/carto.42.4.349.

[12] J.A. Dykes and D.M. Mountain, "Seeking structure in records of spatio-temporal behaviour: visualization issues, efforts and applications," Computational Statistics & Data Analysis, vol. 43.4, pp. 581-603, 2003, doi:http://dx.doi.org/10.1016/S0167-9473(02)00294-3.

[13] D. DiBiase, A. M. McEachren, J. Krygier, and C. Reeves, “Animation and the role of map design in scientific visualization,” Cartography and Geographic Information Systems, vol. 19(4), pp. 201-214, 265-266, 1992.

[14] M. Monmonier, “Temporal generalization for dynamic maps,” Cartography and Geographic Information Science, vol. 23 (2), pp. 96-98, 1996, doi:http://dx.doi.org/10.1559/152304096782562118.

[15] M.-J. Kraak, “Mapping Time: Illustrated by Minard's Map of Napoleon's Russian Campaign of 1812.” ESRI Press, California, 2014.

[16] G. Andrienko, N. Andrienko, H. Schumann, and C. Tominski, “Visualization of trajectory attributes in space-time cube and trajectory wall,” in Cartography from Pole to Pole, Lecture Notes in Geoinformation and Cartography, M. Buchroitner, N. Prechtel, and D. Burghardt, Eds. Springer-Verlag Berlin Heidelberg, pp. 157-163, 2014, doi:http://dx.doi.org/10.1007/978-3-642-32618-9_11.

[17] G.A. Miller, “The magical number seven, plus or minus two: some limits on our capacity for processing information,” Psychological review, vol. 63(2), pp. 81-97, 1956, doi:http://dx.doi.org/10.1037/h0043158.

[18] R.A. Rensink, “Internal vs. External Information in Visual Perception,” Proceedings of the 2nd international symposium on Smart graphics, Jun. 2002, pp. 63-70, doi:http://dx.doi.org/10.1145/569005.569015.

[19] F. Paas, A. Renkl, and J. Sweller, "Cognitive load theory and instructional design: recent developments," Educational psychologist, vol. 38.1, pp. 1-4, 2003, doi:http://dx.doi.org/10.1207/S15326985EP3801_1.

[20] R.E. Mayer and R. Moreno, "Nine ways to reduce cognitive load in multimedia learning," Educational psychologist vol. 38.1, pp. 43-52, 2003, doi:http://dx.doi.org/10.1207/S15326985EP3801_6.

[21] G. Andrienko et al., “Space, time, and visual analytics,” International Journal of Geographical Information Science, vol. 24 (10), pp. 1577-1600, 2010, doi:http://dx.doi.org/10.1080/13658816.2010.508043.

[22] J. Allen, “Maintaining knowledge about temporal intervals,” Communications of the ACM, vol. 26(11), pp. 832-843, 1983, doi:http://dx.doi.org/10.1145/182.358434.

[23] B. FitzGerald, “Science in Geography 1: Developments in Geographical Method”. Oxford University Press, 1974.

[24] M. Sarkar and M.H. Brown, “Graphical Fisheye Views of Graphs,” Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 1992, pp. 83-91, doi:http://dx.doi.org/10.1145/142750.142763.

[25] N. Andrienko and G. Andrienko, “Exploratory Analysis of Spatial and Temporal Data – A Systematic Approach.” Springer, Berlin, Germany, 2006.

[26] C. A. Brewer, “Guidelines for Use of the Perceptual Dimensions of Color for Mapping and Visualization,” International Symposium on Electronic Imaging: Science and Technology. International Society for Optics and Photonics, 1994, pp. 54-63.

Most Probable Paths to Data Loss: An Efficient Method for Reliability Evaluation of Data Storage Systems

Ilias Iliadis and Vinodh Venkatesan
IBM Research – Zurich
Email: {ili,ven}@zurich.ibm.com

***Abstract*—The effectiveness of the redundancy schemes that have been developed to enhance the reliability of storage systems has predominantly been evaluated based on the mean time to data loss** (MTTDL) **metric. This metric has been widely used to compare schemes, to assess tradeoffs, and to estimate the effect of various parameters on system reliability. Analytical expressions for** MTTDL **are typically derived using Markov chain models. Such derivations, however, remain a challenging task owing to the high complexity of the analysis of the Markov chains involved, and therefore the system reliability is often assessed by rough approximations. To address this issue, a general methodology based on the direct-path approximation was used to obtain the** MTTDL **analytically for a class of redundancy schemes and for failure time distributions that also include real-world distributions, such as Weibull and gamma. The methodology, however, was developed for the case of a single direct path to data loss. This work establishes that this methodology can be extended and used in the case where there are multiple shortest paths to data loss to approximately derive the** MTTDL **for a broader set of redundancy schemes. The value of this simple, yet efficient methodology is demonstrated in several contexts. It is verified that the results obtained for RAID-5 and RAID-6 systems match with those obtained in previous work. As a further demonstration, we derive the exact** MTTDL **of a specific RAID-51 system and confirm that it matches with the** MTTDL **obtained from the methodology proposed. In some cases, the shortest paths are not necessarily the most probable ones. We establish that this methodology can be extended to the most probable paths to data loss to derive closed-form approximations for the** MTTDL **of RAID-6 and two-dimensional RAID-5 systems in the presence of unrecoverable errors and device failures. A thorough comparison of the reliability level achieved by the redundancy schemes considered is also conducted.**

Keywords–Shortest path; direct path; data loss; latent errors; MTTDL; rebuild; rare events; RAID; closed-form; analysis.

I. INTRODUCTION

Storage systems experience data losses due to device failures, including disk and node failures. To avoid a permanent loss of data, redundancy schemes were developed that enable the recovery of this data. However, during rebuild operations, additional device failures may occur that eventually lead to permanent data losses. There is a variety of redundancy schemes that offer different levels of reliability as they tolerate varying degrees of device failures. Each of these schemes is characterized by an overhead, which reflects the additional operations that need to be performed for maintaining data consistency, and a storage efficiency, which expresses the additional amount of data, referred to as parity, that needs to be stored in the system.

The reliability of storage systems and the effectiveness of redundancy schemes have predominantly been assessed based on the mean time to data loss (MTTDL) metric, which expresses the amount of time that is expected to elapse until the first data is irrecoverably lost [1][2][3]. During this period, failures cause data to be temporarily lost, which is subsequently recovered owing to the redundancy built into the system.

Analytical expressions for the MTTDL are typically derived using Markov chain models [4], which assume that the times to component failures are independent and exponentially distributed. A methodology for obtaining MTTDL under general non-exponential failure and rebuild time distributions, which therefore does not involve any Markov analysis, was presented in [5]. The complexity of these derivations depends on the redundancy schemes and the underlying system configurations considered. The MTTDL metric has been proven useful for assessing tradeoffs, for comparing schemes, and for estimating the effect of various parameters on system reliability [6][7][8][9]. Analytical closed-form expressions for the MTTDL provide an accurate account of the effect of various parameters on system reliability. However, deriving exact closed-form expressions remains a challenging task owing to the high complexity of the analysis of the Markov chains involved [10][11]. For this reason, the system reliability is often assessed by rough approximations. As the direct MTTDL analysis is typically hard, an alternative is performing event-driven simulations [12][13]. However, simulations do not provide insight into how the various parameters affect the system reliability. This article addresses these issues by presenting a simple, yet efficient method, referred to as *most-probable-path approximation*, to obtain the MTTDL analytically for a broad set of redundancy schemes. It achieves that by considering the most likely paths that lead to data loss, which are the shortest ones. In contrast to simulations, this method provides approximate closed-form expressions for the MTTDL, thus circumventing the inherent complexity of deriving exact expressions using Markov analysis. Note also that this method was previously applied in the context of assessing system unavailability, in particular for systems characterized by large Markov chains [14]. It turns out that this approach agrees with the principle encountered in the probability context expressed by the phrase *"rare events occur in the most likely way"*. This is also demonstrated in [15], where the reliability level of

systems composed of highly reliable components is essentially determined by the so-called "main event", which is the shortest way of failure appearance, that is, along the minimal monotone paths.

In [5][16][17][18][19], it was shown that the direct-path approximation, which considers paths without loops, yields accurate analytical reliability results. To further investigate the validity of the shortest-path-approximation method, we apply it to derive the MTTDL results for RAID-5 and RAID-6 systems and subsequently verify that they match with those obtained in previous works [2][3] for practical cases where the device failure rates are much smaller than the device rebuild rates. In all these previous works though, there is a single direct path to data loss. In contrast, our article is concerned with the case where there are multiple shortest paths to data loss. In this work, we investigate this issue and establish that the shortest-path-approximation method can be extended and also applied in the case of multiple shortest paths and can yield accurate reliability results. In particular, we derive the approximate MTTDL of a RAID-51 system using the shortest-path approximation. Subsequently, as a demonstration of the validity of the method proposed, we derive the exact MTTDL for a specific instance of a RAID-51 system and confirm that it matches with the corresponding MTTDL obtained using our method. Furthermore, we establish that the shortest-path approximation can be extended to the most probable path approximation in cases where the shortest paths may not necessarily be the most probable ones. In fact, an approximation that considers all direct paths implicitly considers the most probable ones because the direct paths are the most probable ones owing to the absence of loops.

The key contributions of this article are the following. We consider the reliability of the RAID-5, RAID-6, and RAID-51 systems that was assessed in our earlier work [1]. In this study, we extend our previous work by also considering two-dimensional RAID-5 systems. The MTTDL of a specific square two-dimensional RAID-5 system was estimated through a Markov chain model in [20], but no closed-form expression was provided owing to its complexity. In this work, using the shortest-path-approximation method, we obtain approximate closed-form expressions for the MTTDL that are general, simple, yet accurate for real-world systems. Furthermore, we perform a thorough comparison of the reliability levels, in terms of the MTTDL, achieved by these schemes. Subsequently, we consider the reliability of RAID-6 and two-dimensional RAID-5 systems in the presence of unrecoverable (latent) errors and device failures, and establish that in general the shortest paths may not be the most probable ones, A new enhanced methodology that considers the most probable paths, as opposed to the shortest paths, is subsequently introduced for efficiently assessing system reliability.

The remainder of the paper is organized as follows. Section II reviews the general framework for deriving the MTTDL of a storage system. Subsequently, the notion of the direct path to data loss is discussed in Section III, and the efficiency of the direct-path approximation is demonstrated in Section IV. Section V discusses the case of multiple shortest paths to data loss and presents the analysis of the RAID-51 and two-dimensional RAID-5 systems. Section VI presents a thorough comparison of the various redundancy schemes considered. Section VII provides a detailed analysis and comparison of the RAID-6 and two-dimensional RAID-5 systems in the presence of independent unrecoverable sector errors. The shortest-path-approximation method is enhanced to account for the most probable paths. Finally, we conclude in Section IX.

II. Derivation of MTTDL

In this section, we review the various methods that are used to obtain the MTTDL analytically.

A. Markov Analysis

Continuous-time Markov chain (CTMC) models reflecting the system operation can be constructed when the device failures and rebuild times are assumed to be independent and exponentially distributed. Under these assumptions, an appropriate CTMC model can be formulated to characterize the system behavior and capture the corresponding state transitions, including those that lead to data loss. Subsequently, using the infinitesimal generator matrix approach and determining the average time spent in the transient states of the Markov chain yields a closed-form expression for the MTTDL of the system [4]. The results obtained by using CTMC models are often approximate because in practice the times to device failure and the rebuild times are not exponentially distributed. To address this issue, a more general analytical method is required.

B. Non-Markov Analysis

Here, we briefly review the general framework for deriving the MTTDL developed in [5][16] using an analytical approach that does not involve any Markov analysis and therefore avoids the deficiencies of Markov models. The underlying models are not semi-Markov, in that the the system evolution does not depend only on the latest state, but also on the entire path that led to that state. In particular, it depends on the fractions of the data not rebuilt when entering each state. In [21], it was demonstrated that a careless evaluation of these fractions may in fact easily lead to erroneous results.

At any point in time, the system can be thought to be in one of two modes: normal mode and rebuild mode. During normal mode, all data in the system has the original amount of redundancy and there is no active rebuild in process. During rebuild mode, some data in the system has less than the original amount of redundancy and there is an active rebuild process that is trying to restore the redundancy lost. A transition from normal to rebuild mode occurs when a device fails; we refer to the device failure that causes this transition as a *first-device* failure. Following a first-device failure, a complex sequence of rebuild operations and subsequent device failures may occur, which eventually leads the system either to an irrecoverable data loss (DL), with the probability of this event denoted by P_{DL}, or back to the original normal mode by restoring all replicas lost. Typically, the rebuild times are much shorter than the times to failure. Consequently, the time required for this complex sequence of events to complete is negligible compared with the time between successive first-device failures and therefore can be ignored.

Let T_i be the ith interval of a fully operational period, that is, the time interval from the time at which the system is brought to its original state until a subsequent first-device

failure occurs. As the system becomes stationary, the length of T_i converges to T. In particular, for a system comprising N devices with a mean time to failure of a device equal to $1/\lambda$, the expected length of T is given by [5]

$$E(T) := \lim_{i \to \infty} E(T_i) = 1/(N\,\lambda)\ . \tag{1}$$

The notation used is given in Table I. Note that the methodology presented here does not involve any Markov analysis and holds for general failure time distributions, which can be exponential or non-exponential, such as the Weibull and gamma distributions.

As the probability that each first-device failure results in data loss is $P_{\rm DL}$, the expected number of first-device failures until data loss occurs is $1/P_{\rm DL}$. Thus, by neglecting the effect of the relatively short transient rebuild periods of the system, the MTTDL is essentially the product of the expected time between two first-device-failure events, $E(T)$, and the expected number of first-device-failure events, $1/P_{\rm DL}$:

$$\text{MTTDL} \approx \frac{E(T)}{P_{\rm DL}}\ . \tag{2}$$

Substituting (1) into (2) yields

$$\text{MTTDL} \approx \frac{1}{N\,\lambda\,P_{\rm DL}}\ . \tag{3}$$

III. Direct Path to Data Loss

As mentioned in Section II, during rebuild mode, some data in the system has less than the original amount of redundancy and there is an active rebuild process that aims at restoring the lost redundancy. The direct path to data loss represents the most likely scenario that leads to data loss. This path considers the smallest number of subsequent device failures that occur while the system is in rebuild mode and lead to data loss.

The direct-path-approximation method was applied in [5][16] and led to an analytical approach that does not involve any Markov analysis and therefore avoids the deficiencies of Markov models. This approach yields accurate results when the storage devices are highly reliable, that is, when the ratio of the mean rebuild time $1/\mu$ (typically on the order of tens of hours) to the mean time to failure of a device $1/\lambda$ (typically on the order of a few years) is very small:

$$\frac{1}{\mu} \ll \frac{1}{\lambda}\ , \quad \text{or} \quad \frac{\lambda}{\mu} \ll 1\ , \quad \text{or} \quad \lambda \ll \mu\ . \tag{4}$$

More specifically, this approach considers the system to be in exposure level e when the maximum number of replicas lost by any of the data (or the maximum number of codeword symbols lost in an erasure-coded system) is equal to e. Let us consider, for instance, a replication-based storage system where user data is replicated r times. In this case, the system is in exposure level e if there exists data with $r-e$ copies, but there is no data with fewer than $r-e$ copies. Device failures and rebuild processes cause the exposure level to vary over time. Consider the direct path of successive transitions from exposure level 1 to r. In [16], it was shown that $P_{\rm DL}$ can be approximated by the probability of the direct path to data loss, $P_{\rm DL,direct}$, when devices are highly reliable, that is,

$$P_{\rm DL} \approx P_{\rm DL,direct} = \prod_{e=1}^{r-1} P_{e \to e+1}, \tag{5}$$

where $P_{e \to e+1}$ denotes the transition probability from exposure level e to $e+1$. In fact, the above approximation holds for arbitrary device failure time distributions, and the relative error tends to zero as for highly reliable devices the ratio λ/μ tends to zero [5]. The MTTDL is then obtained by substituting (5) into (3). In [18], the direct-path methodology is extended to more general erasure codes, which include RAID systems.

Note that this analysis can also be applied to assess reliability, in terms of the MTTDL, for systems modeled using a CTMC. For instance, in [6], a RAID-5 system that was modeled using a CTMC was analyzed by both a Markov analysis and an approach similar to the general framework. This fact is used in Section IV to compare the MTTDL of RAID systems obtained using the direct-path approximation in the context of the general framework with the corresponding MTTDL obtained using Markov analysis of CTMCs. This approach is simpler, in that it circumvents the inherent complexity of deriving exact MTTDL expressions using Markov analysis. In Section V, we demonstrate that the direct-path-approximation method can be extended and also applied in the case of multiple shortest paths. We establish this for a system modeled using a CTMC, and conjecture that this should also hold in the case of non-Markovian systems.

Note that this method is in contrast to other methods presented in previous works that associate a probability to each device being in a failed state [22]. In particular, those works assume that these probabilities are given and therefore do not account for the rebuild processes, whereas the methods presented in this work do account for the rebuild processes through the probabilities of traversing various states until data loss occurs.

TABLE I. Notation of System Parameters

Parameter	Definition
N	Number of devices in the system
$1/\lambda$	Mean time to failure for a device
$1/\mu$	Mean time to rebuild device failures
$se^{(\text{RAID})}$	Storage efficiency of a RAID scheme
S	Sector size
C_d	Device capacity
P_s	Probability of an unrecoverable or latent sector error
n_s	Number of data sectors in a device ($n_s = C_d/S$)

IV. Comparison of Markov Analysis and Direct-Path Approximation

A common scheme used for tolerating device (disk) failures is the redundant array of independent disks (RAID) [2][3]. The RAID-5 scheme arranges devices in groups (arrays), each with one redundant device, and can tolerate one device failure per array. Similarly, the RAID-6 scheme arranges devices in arrays, each with two redundant devices, and can tolerate up to two device failures per array. Considering a RAID array comprised of N devices, the storage efficiency of a RAID-5 system is given by

$$se^{(\text{RAID-5})} \ = \ \frac{N-1}{N}\ , \tag{6}$$

and the storage efficiency of a RAID-6 system is given by

$$se^{(\text{RAID-6})} \;=\; \frac{N-2}{N}\;. \tag{7}$$

It turns out that the MTTDL of systems comprised of highly reliable devices can be approximated by using the *direct-path approximation*. We proceed to demonstrate this by presenting two specific examples, the RAID-5 and RAID-6 systems. In both cases, the RAID array is assumed to contain N devices, and the numbered states of the corresponding Markov models represent the number of failed devices. The DL state represents a data loss due to a device failure that occurs when the system is in the critical mode of operation. A RAID array is considered to be in *critical mode* when an additional device failure can no longer be tolerated. Thus, RAID-5 and RAID-6 arrays are in critical mode when there are $N-1$ devices and $N-2$ devices in operation, that is, when they operate with one device and two devices failed, respectively.

A. MTTDL *of a RAID-5 Array*

The Markov chain model for a RAID-5 array is shown in Fig. 1. When the first device fails, the array enters critical mode, which corresponds to the transition from state 0 to state 1. As initially there are N devices in operation, the mean time until the first failure is equal to $1/(N\lambda)$, and the corresponding transition rate is its inverse, that is, $N\lambda$. Subsequently, the critical mode ends owing to either a successful completion of the rebuild or another device failure. The former event is represented by the state transition from state 1 to state 0 with a rate of μ, given that the mean rebuild time is equal to $1/\mu$. The latter event leads to data loss and is represented by the state transition from state 1 to state DL with a rate of $(N-1)\lambda$ given that in critical mode there are $N-1$ devices in operation.

The exact MTTDL, denoted by $\text{MTTDL}_{\text{RAID-5}}$, is obtained from [6, Eq. (45)] by setting $P_{\text{uf}}^{(1)} = 0$:

$$\text{MTTDL}_{\text{RAID-5}} \;=\; \frac{\mu + (2N-1)\lambda}{N\,(N-1)\,\lambda^2}\;. \tag{8}$$

Note that when $\lambda \ll \mu$, the first term of the numerator in (8) can be ignored, such that the $\text{MTTDL}_{\text{RAID-5}}$ can be approximated by $\text{MTTDL}_{\text{RAID-5}}^{(\text{approx})}$ as follows:

$$\text{MTTDL}_{\text{RAID-5}}^{(\text{approx})} \;\approx\; \frac{\mu}{N(N-1)\,\lambda^2}\;. \tag{9}$$

This result was obtained in [2] by using an approach that is essentially the direct-path approximation. Next, we present its derivation for completeness. The transition from state 0 to state 1 represents the first device failure. The direct path to data loss involves a subsequent device failure before it can complete the rebuild process and return to state 0. This corresponds to the state transition from state 1 to state DL, with the corresponding probability $P_{1\to\text{DL}}$ given by

$$P_{\text{DL}} \approx P_{\text{DL,direct}} = P_{1\to\text{DL}} = \frac{(N-1)\,\lambda}{\mu + (N-1)\,\lambda} \tag{10}$$

$$\approx (N-1)\left(\frac{\lambda}{\mu}\right)\;, \tag{11}$$

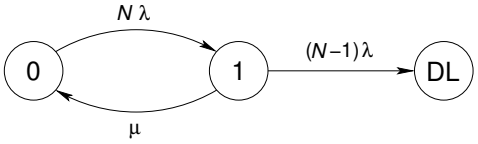


Figure 1. Reliability model for a RAID-5 array.

where the approximation is obtained by using (4) and therefore neglecting the second term of the denominators in (10). Substituting (10) into (3) yields

$$\text{MTTDL}'_{\text{RAID-5}} \;\approx\; \frac{\mu + (N-1)\,\lambda}{N(N-1)\,\lambda^2}\;. \tag{12}$$

Note that the approximation given in (9) now follows immediately from (12) by using (4) and therefore neglecting the second term of the numerator.

B. MTTDL *of a RAID-6 Array*

The Markov chain model for a RAID-6 array is shown in Fig. 2. The first device failure is represented by the transition from state 0 to state 1. As initially there are N devices in operation, the mean time until the first failure is $1/(N\lambda)$, and the corresponding transition rate is its inverse, that is, $N\lambda$. The system exits from state 1 owing to either a successful completion of the rebuild or another device failure. The former event is represented by the state transition from state 1 to state 0 with a rate of μ. The latter event is represented by the state transition from state 1 to state 2 with a rate of $(N-1)\lambda$. Subsequently, the system exits from state 2 owing to either a successful completion of the rebuild or another device failure. The former event is represented by the state transition from state 2 to state 0 with a rate of μ, given that the mean rebuild time is equal to $1/\mu$. The latter event leads to data loss and is represented by the state transition from state 2 to state DL with a rate of $(N-2)\lambda$ given that in critical mode there are $N-2$ devices in operation.

The exact MTTDL, denoted by $\text{MTTDL}_{\text{RAID-6}}$, is obtained from [6, Eq. (52)] by setting $\mu_1 = \mu_2 = \mu$ and $P_{\text{uf}}^{(\text{r})} = P_{\text{uf}}^{(2)} = 0$:

$$\text{MTTDL}_{\text{RAID-6}} \;=\; \frac{\mu^2 + 3(N-1)\lambda\mu + (3N^2-6N+2)\,\lambda^2}{N\,(N-1)\,(N-2)\,\lambda^3}\;. \tag{13}$$

Note that when $\lambda \ll \mu$, the last two terms of the numerator of (13) can be neglected and thus $\text{MTTDL}_{\text{RAID-6}}$ can be approximated by $\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})}$ as follows:

$$\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})} \;\approx\; \frac{\mu^2}{N(N-1)(N-2)\,\lambda^3}\;, \tag{14}$$

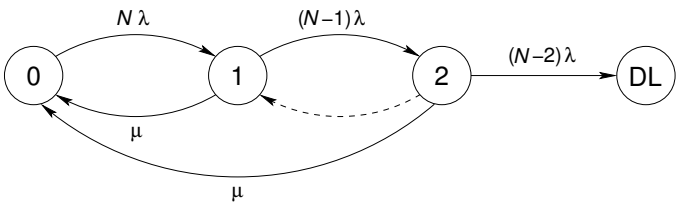


Figure 2. Reliability model for a RAID-6 array.

which is the same result as that reported in [3].

We now proceed to show how the approximate MTTDL of the system can be derived in a straightforward manner by applying the direct-path-approximation technique. The transition from state 0 to state 1 represents the first device failure. The direct path to data loss involves two subsequent device failures before it can complete the rebuild process and return to state 0. This corresponds to the state transitions from state 1 to state 2 and from state 2 to state DL, with the corresponding probabilities $P_{1\to 2}$ and $P_{2\to \mathrm{DL}}$ given by

$$P_{1\to 2} = \frac{(N-1)\,\lambda}{\mu + (N-1)\,\lambda}\ . \tag{15}$$

and

$$P_{2\to \mathrm{DL}} = \frac{(N-2)\,\lambda}{\mu + (N-2)\,\lambda}\ . \tag{16}$$

Thus, the probability of data loss, that is, the probability that from state 1 the system goes to state DL before it can reach state 0, is equal to

$$\begin{aligned} P_{\mathrm{DL}} \approx P_{\mathrm{DL,direct}} &= P_{1\to 2}\, P_{2\to \mathrm{DL}} \\ &= \frac{(N-1)\,\lambda}{\mu + (N-1)\,\lambda} \cdot \frac{(N-2)\,\lambda}{\mu + (N-2)\,\lambda} \qquad (17) \\ &\approx (N-1)(N-2)\left(\frac{\lambda}{\mu}\right)^2\ , \qquad (18) \end{aligned}$$

where the approximation is obtained by using (4) and therefore neglecting the second terms of the denominators in (17).

We verify that substituting (18) into (3) yields the approximation given in (14).

Remark 1: If the transition from state 2 to state 0 were not to state 0 but to state 1 instead, as shown in Fig. 2 by the dashed arrow, the expression for $P_{2\to \mathrm{DL}}$ given by (16) would still hold. However, in this case it would hold that $P_{\mathrm{DL}} > P_{\mathrm{DL,direct}}$ as, in addition to the direct path $1 \to 2 \to \mathrm{DL}$, there are other possible paths $1 \to 2 \to 1 \to 2 \to \cdots \to 1 \to 2 \to \mathrm{DL}$ to data loss. In [16] it was shown that, for systems with highly reliable components, the direct path dominates the effect of all other possible paths and therefore its probability, $P_{\mathrm{DL,direct}}$, approximates well the probability of all paths, P_{DL}, that is,

$$P_{\mathrm{DL}} \approx P_{\mathrm{DL,direct}} = P_{1\to 2}\, P_{2\to \mathrm{DL}} \approx \frac{(N-1)(N-2)\,\lambda^2}{\mu^2}\ . \tag{19}$$

In this case, the MTTDL is given by

$$\mathrm{MTTDL}'_{\text{RAID-6}} \ = \ \frac{(3N^2-6N+2)\,\lambda^2 + 2(N-1)\lambda\mu + \mu^2}{N\,(N-1)\,(N-2)\,\lambda^3}\ , \tag{20}$$

which, as expected, is less than that given in (13). Despite this difference, the approximation given in (14) still holds because (19) is the same as (18).

V. Multiple Shortest Paths to Data Loss

Here, we consider redundancy schemes for which there are multiple shortest paths to data loss. Following the analysis presented in [16] for the direct-path approximation, we conjecture that, for systems with highly reliable devices, the shortest paths dominate the effect of all other possible paths and therefore the sum of their corresponding probabilities, $P_{\mathrm{DL,shortest}}$, approximates well the probability of all paths, P_{DL}, that is,

$$P_{\mathrm{DL}} \approx P_{\mathrm{DL,shortest}}\ . \tag{21}$$

A. A RAID-51 Array

We proceed by considering a RAID-51 system, which is a RAID-5 array with mirroring. The contents of failed devices are recovered by their mirrors, and if this is not possible, they are recovered through the corresponding RAID-5 arrays. The configuration comprises D pairs of mirrored devices, where each pair contains two devices with identical content. Therefore, it consists of two identical RAID-5 arrays, for a total of $N = 2\,D$ devices and a storage efficiency given by

$$se^{(\text{RAID-51})} \ = \ \frac{D-1}{2D} \ = \ \frac{N-2}{2N}\ . \tag{22}$$

This configuration was considered in [11], referred to as RAID 5+1, with the corresponding Markov model shown in [11, Fig. 7(a)]. It is redrawn in Fig. 3 with the parameters λ and μ corresponding to the parameters μ and ν of the initial figure, respectively. Also, the DL states correspond to the 'Failure' states, and the state tuples (x, y, z) indicate that there are x pairs with both devices in operation, y pairs with one device in operation and one device failed, and z pairs with both devices failed. Also, some typos regarding the transition rates were corrected.

An exact evaluation of the MTTDL associated with this Markov chain model appears to be a very challenging, if not infeasible, task. Thus, in [11] a rough approximation was obtained by first deriving the failure and repair rates for a mirrored pair of devices, and then substituting these values into expression (9) for a single RAID-5 system. The MTTDL is obtained in [11, Eq. (11)] as follows:

$$\mathrm{MTTDL} \ \approx \ \frac{\mu^3}{4D(D-1)\,\lambda^4}\ . \tag{23}$$

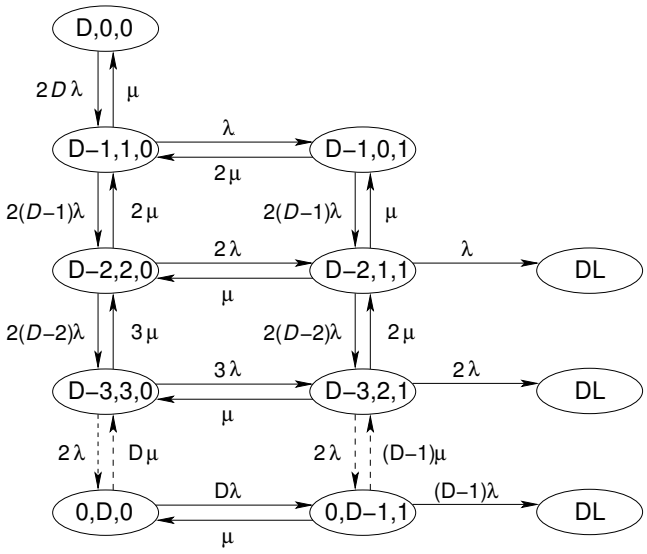


Figure 3. Reliability model for a RAID-51 array.

1) MTTDL *Evaluation Using the Shortest-Path Approximation:* The transition from state $(D,0,0)$ to state $(D-1,1,0)$ represents the first device failure. As initially there are $2\,D$ devices in operation, the mean time until the first failure is $1/(2D\lambda)$, and the corresponding transition rate is its inverse, $2D\lambda$.

The most likely path to data loss is the shortest path from state $(D-1,1,0)$ to a DL state, which in this case comprises two such paths, as shown in Fig. 4: the upper path $(D-1,1,0) \rightarrow (D-1,0,1) \rightarrow (D-2,1,1) \rightarrow$ DL and the lower path $(D-1,1,0) \rightarrow (D-2,2,0) \rightarrow (D-2,1,1) \rightarrow$ DL. Each of these paths involves three subsequent device failures.

After the first device has failed, there are $D-1$ pairs with both devices in operation, and one pair, say PR_1, with one device in operation and one device failed, which corresponds to the transition from state $(D,0,0)$ to state $(D-1,1,0)$. The rebuild of the failed device consists of recovering its data to a new spare device by copying the contents of its mirror to it, that is, of the device in operation in PR_1. Then, the next event can be either a successful completion of the rebuild or another device failure. The former event is represented by the state transition from state $(D-1,1,0)$ to state $(D,0,0)$ with a rate of μ. For the latter event, two cases are considered:

Case 1: Upper path. The second device that fails is the device in operation concerning pair PR_1, which corresponds to the transition from state $(D-1,1,0)$ to state $(D-1,0,1)$, as now both devices of pair PR_1 have failed, and all other $D-1$ pairs remain intact. The transition rate is λ, which is the failure rate of the last failed device. The contents of the devices of pair PR_1 are recovered through the corresponding RAID-5 arrays. As both devices of pair PR_1 are under rebuild, the transition rate from state $(D-1,0,1)$ back to state $(D-1,1,0)$ is 2μ. If, however, prior to the completion of any of these two rebuilds another device of the remaining $2(D-1)$ devices fails, then there will be $D-2$ pairs with both devices in operation, one pair, say PR_2, with one device in operation and one device failed, and pair PR_1 with both devices failed. This corresponds to the transition from state $(D-1,0,1)$ to state $(D-2,1,1)$, with a transition rate equal to $2(D-1)\lambda$. Note that in [11, Fig. 7(a)] this transition rate is erroneously indicated as $(2D-1)\mu$ instead of $2(D-1)\mu$.

Case 2: Lower path. The second device that fails is one of the $2(D-1)$ devices in the $D-1$ pairs, say a device concerning PR_2. This corresponds to the transition from state $(D-1,1,0)$ to state $(D-2,2,0)$, as both pairs PR_1 and PR_2 now have one device in operation and one device failed, and all other $D-2$ pairs remain intact. The corresponding transition rate is equal to $2(D-1)\lambda$. Note that in [11, Fig. 7(a)] this transition rate is erroneously indicated as $(2D-1)\mu$ instead of $2(D-1)\mu$. The contents of the failed devices are recovered from their corresponding mirrors. As both devices of the two pairs PR_1 and PR_2 are under rebuild, the transition rate from state $(D-2,2,0)$ back to state $(D-1,1,0)$ is 2μ. If, however, prior to the completion of any of these two rebuilds another device of the two remaining devices in operation in PR_1 and PR_2 fails (say, that of pair PR_1), then there will be $D-2$ pairs with both devices in operation, one pair (PR_2) with one device in operation and one device failed, and one pair (PR_1) with both devices failed. This corresponds to the transition from state $(D-2,2,0)$ to state $(D-2,1,1)$, with a transition rate 2λ.

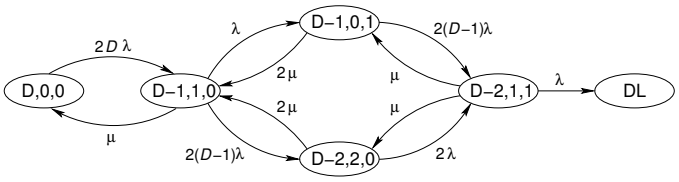


Figure 4. Shortest-path reliability model for a RAID-51 array.

At state $(D-2,1,1)$, the failed device in pair PR_2 is recovered by its mirror. However, the corresponding failed device in pair PR_1 cannot be recovered because the RAID-5 array has suffered two device failures. In contrast, the failed device in pair PR_1 can be recovered because the corresponding RAID-5 array has suffered only one device failure.

The completion of the rebuild of the failed device in pair PR_2 corresponds to the transition from state $(D-2,1,1)$ to state $(D-1,0,1)$, with a transition rate of μ. The completion of the rebuild of the failed device in pair PR_1 through the RAID capability corresponds to the transition from state $(D-2,1,1)$ to state $(D-2,2,0)$, with a transition rate of μ. Note that in [11, Fig. 7(a)] this transition rate is erroneously indicated as 2μ instead of μ. If, however, prior to the completion of any of these rebuilds, the device still in operation of pair PR_2 fails, this leads to data loss, as there will be two pairs failed, with each of the RAID-5 arrays having two devices failed. This corresponds to the transition from state $(D-2,1,1)$ to state DL, with a corresponding rate of λ.

The probabilities of the transitions discussed above are given by

$$P_{(D-1,1,0)\rightarrow(D-1,0,1)} = \frac{\lambda}{\mu + (2D-1)\,\lambda}\,, \tag{24}$$

$$P_{(D-1,0,1)\rightarrow(D-2,1,1)} = \frac{2(D-1)\,\lambda}{2\,\mu + 2(D-1)\,\lambda}\,, \tag{25}$$

$$P_{(D-1,1,0)\rightarrow(D-2,2,0)} = \frac{2(D-1)\,\lambda}{\mu + (2D-1)\,\lambda}\,, \tag{26}$$

$$P_{(D-2,2,0)\rightarrow(D-2,1,1)} = \frac{2\,\lambda}{2\,\mu + 2\,\lambda}\,, \tag{27}$$

and

$$P_{(D-2,1,1)\rightarrow\mathrm{DL}} = \frac{\lambda}{2\,\mu + \lambda}\,. \tag{28}$$

Consequently, the probability of the upper path to data loss, P_u, is given by

$$\begin{aligned} P_u &= P_{(D-1,1,0)\rightarrow(D-1,0,1)} P_{(D-1,0,1)\rightarrow(D-2,1,1)} P_{(D-2,1,1)\rightarrow\mathrm{DL}} \\ &= \frac{\lambda}{\mu + (2D-1)\,\lambda} \cdot \frac{2(D-1)\,\lambda}{2\,\mu + 2(D-1)\,\lambda} \cdot \frac{\lambda}{2\,\mu + \lambda}\,, \end{aligned} \tag{29}$$

and that of the lower path to data loss, P_l, is given by

$$\begin{aligned} P_l &= P_{(D-1,1,0)\rightarrow(D-2,2,0)} P_{(D-2,2,0)\rightarrow(D-2,1,1)} P_{(D-2,1,1)\rightarrow\mathrm{DL}} \\ &= \frac{2(D-1)\,\lambda}{\mu + (2D-1)\,\lambda} \cdot \frac{2\,\lambda}{2\,\mu + 2\,\lambda} \cdot \frac{\lambda}{2\,\mu + \lambda}\,. \end{aligned} \tag{30}$$

By considering (4), (29) and (30) yield the following approximations:

$$P_u \approx \frac{\lambda}{\mu} \cdot \frac{2(D-1)\,\lambda}{2\,\mu} \cdot \frac{\lambda}{2\,\mu} = \frac{(D-1)\,\lambda^3}{2\mu^3} \quad (31)$$

and

$$P_l \approx \frac{2(D-1)\,\lambda}{\mu} \cdot \frac{\lambda}{\mu} \cdot \frac{\lambda}{2\,\mu} = \frac{(D-1)\,\lambda^3}{\mu^3} . \quad (32)$$

The probability of the shortest paths to data loss, $P_{\text{DL,shortest}}$, is the sum of P_u and P_l, which by using (21), (31), and (32), yields

$$P_{\text{DL}} \approx P_{\text{DL,shortest}} = P_u + P_l \approx \frac{3(D-1)}{2} \left(\frac{\lambda}{\mu}\right)^3 . \quad (33)$$

Substituting (33) into (3), and considering $N = 2D$, yields the approximate MTTDL of the RAID-51 system, $\text{MTTDL}_{\text{RAID-51}}^{(\text{approx})}$, given by

$$\text{MTTDL}_{\text{RAID-51}}^{(\text{approx})} \approx \frac{\mu^3}{3D(D-1)\,\lambda^4} . \quad (34)$$

Remark 2: Note that the prediction given by (34) is higher than that obtained in [11], which is given by (23). At first glance, this seems to be counterintuitive. The approximation in [11] considers only failures of mirrored device pairs, which corresponds to the upper path to data loss. As this neglects the lower path, one would expect the prediction in [11] to be higher, not lower. The reason for this counterintuitive result is that considering additional paths may, on the one hand increase the number of paths that lead to data loss, but on the other hand it may also increase the number of paths that do not lead to data loss, therefore delaying the occurrence of data loss. For instance, when the lower path is neglected, the probability $P_{(D-2,1,1)\to\text{DL}}$ of the transition from state $(D-2,1,1)$ to state DL is equal to $\lambda/(\lambda+\mu)$, which is greater than the corresponding one given by (28) if also the lower path is considered.

2) Exact MTTDL *Evaluation for D = 3:* An exact evaluation of the reliability of a RAID-51 system through the MTTDL associated with the corresponding Markov chain model shown in Fig. 3 appears to be a very challenging, if not infeasible, task for arbitrary D. Therefore, we proceed by considering a RAID-51 system with $D = 3$. The corresponding Markov chain model is shown in Fig. 5. The exact MTTDL of this system, denoted by $\text{MTTDL}_{\text{RAID-51}}^{(D=3)}$, is obtained by using the infinitesimal generator matrix approach and determining the average time spent in the transient states of the Markov chain [4]. Because of space limitations, we only provide the final result:

$$\text{MTTDL}_{\text{RAID-51}}^{(D=3)} = \frac{2+20\frac{\lambda}{\mu}+93(\frac{\lambda}{\mu})^2+287(\frac{\lambda}{\mu})^3+677(\frac{\lambda}{\mu})^4+939(\frac{\lambda}{\mu})^5+630(\frac{\lambda}{\mu})^6}{12\,\lambda^4\,\mu^{-3}\,[3+18\frac{\lambda}{\mu}+35(\frac{\lambda}{\mu})^2+30(\frac{\lambda}{\mu})^3]} . \quad (35)$$

Note that when $\lambda \ll \mu$, $\text{MTTDL}_{\text{RAID-51}}^{(D=3)}$ can be approximated by $\text{MTTDL}_{\text{RAID-51}}^{(D=3,\text{approx})}$ as follows:

$$\text{MTTDL}_{\text{RAID-51}}^{(D=3,\text{approx})} \approx \frac{\mu^3}{18\,\lambda^4} , \quad (36)$$

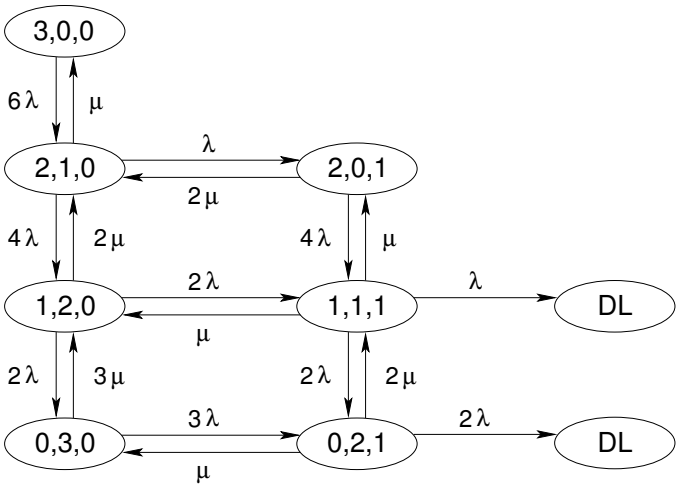


Figure 5. Reliability model for a RAID-51 array with $D = 3$.

which is the same result as that predicted by (34) for $D = 3$ and therefore confirms its validity.

B. A Two-Dimensional RAID-5 Array

We consider a two-dimensional RAID-5 array with the devices arranged in K rows and D columns for a total of $N = KD$ devices, including superparity [13]. In this configuration, denoted by 2D-RAID-5, the devices in each row and each column form a RAID-5 array with the corresponding storage efficiency given by

$$se^{(\text{2D-RAID-5})} = \frac{(K-1)\,(D-1)}{K\,D} . \quad (37)$$

The contents of failed devices are recovered either horizontally or vertically through the corresponding RAID-5 arrays. This system tolerates all triple device failures and can also tolerate more than three device failures for certain constellations, e.g., the failure of an entire row or column. However, as the devices are assumed to fail independently, the shortest path to data loss is due to the failure of four devices occurring in the constellation shown in Fig. 6 with the failed devices located in two rows and two columns. The special case of a specific square RAID-5 array (i.e., $K = D$) was considered in [20], but no closed-form expression for the MTTDL was provided owing to its complexity. Here, we obtain approximate closed-form expressions for the MTTDL that are general, simple, yet accurate for real-world systems. Fig. 6 also shows the Markov chain model corresponding to the shortest path to data loss. The state tuples (x, y, z, w) indicate that there are x rows with one device failed and $D-1$ devices in operation, y rows with two devices failed and $D-2$ devices in operation, z columns with one device failed and $N-1$ devices in operation, and w columns with two devices failed and $N-2$ devices in operation. The relevant states are shown next to the corresponding device failure constellations indicated with 'x' on the $K \times D$ plane.

We now proceed to evaluate the MTTDL using the shortest-path approximation. The transition from state $(0,0,0,0)$ to state $(1,0,1,0)$ represents the first device failure. The

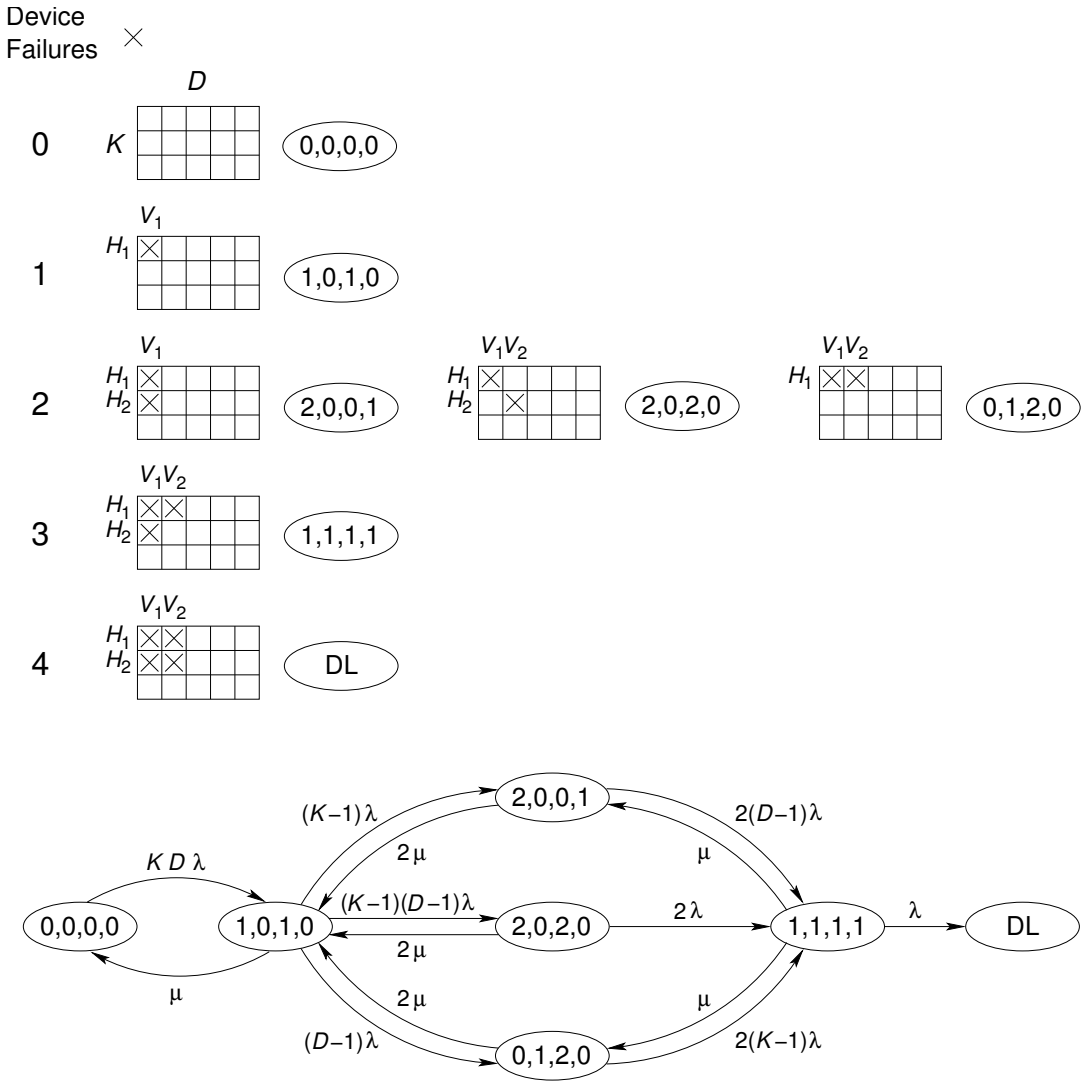


Figure 6. Shortest-path reliability model for a two-dimensional RAID-5 array.

most likely path to data loss is the shortest path from state $(1,0,1,0)$ to the DL state, which in this case comprises three such paths, as shown in Fig. 6: the upper path $(1,0,1,0) \rightarrow (2,0,0,1) \rightarrow (1,1,1,1) \rightarrow$ DL, the middle path $(1,0,1,0) \rightarrow (2,0,2,0) \rightarrow (1,1,1,1) \rightarrow$ DL, and the lower path $(1,0,1,0) \rightarrow (0,1,2,0) \rightarrow (1,1,1,1) \rightarrow$ DL. Each of these paths involves three subsequent device failures.

After the first device has failed, there are two RAID-5 arrays, one horizontal RAID-5 array, say row H_1, and one vertical RAID-5 array, say column V_1, with one device failed in each of them. Consequently, this failure corresponds to the transition from state $(0,0,0,0)$ to state $(1,0,1,0)$. As initially there are KD devices in operation, the mean time until the first failure is $1/(KD\lambda)$, and the corresponding transition rate is its inverse, $KD\lambda$. The rebuild of the failed device can be performed using either $H1$ or $V1$. Then, the next event can be either a successful completion of the rebuild or another device failure. The former event is represented by the state transition from state $(1,0,1,0)$ to state $(0,0,0,0)$, with a rate of μ. For the latter event, three cases are considered:

Case 1: Upper path. The second device that fails is one of the $K-1$ operating devices in V_1. This occurs with a transition rate of $(K-1)\lambda$ and results in another horizontal RAID-5 array, say, row H_2, with one device failed. This corresponds to the transition from state $(1,0,1,0)$ to state $(2,0,0,1)$, as there are now two rows with one device failed in each of them and one column with two devices failed. As the contents of the two failed devices in V_1 are rebuilt in parallel through the corresponding H_1 and H_2 RAID-5 arrays, the transition rate from state $(2,0,0,1)$ back to state $(1,0,1,0)$ is 2μ. If, however, prior to the completion of any of these two rebuilds, another of the remaining $2(D-1)$ devices in H_1 and H_2 (say, the one in row H_1 and column V_2) fails, then there will be
– a column, namely V_2, with one device failed;
– a row, namely H_1, with two devices failed;
– a row, namely H_2, with one device failed, and
– a column, namely V_1, with two devices failed.
Note that the $(K-2)D$ operational devices in the remaining $K-2$ horizontal RAID-5 arrays are not considered because their failure leads to states that are not in the shortest paths. The above corresponds to the transition from state $(2,0,0,1)$

to state $(1,1,1,1)$, with a transition rate equal to $2(D-1)\lambda$.

Case 2: Middle path. The second device that fails is one of the $(K-1)(D-1)$ operating devices that are not in H_1 or V_1. This occurs with a transition rate of $(K-1)(D-1)\lambda$ and results in another horizontal RAID-5 array, say, row H_2, and another vertical RAID-5 array, say, column V_2, with one device failed. This corresponds to the transition from state $(1,0,1,0)$ to state $(2,0,2,0)$ as there are now two rows and two columns with one device failed in each of them. As the contents of the two failed devices are rebuilt in parallel through the corresponding, horizontal or vertical, RAID-5 arrays, the transition rate from state $(2,0,0,1)$ back to state $(1,0,1,0)$ is 2μ. If, however, prior to the completion of any of these two rebuilds, either the (H_1, V_2) or the (H_2, V_1) device fails, where (H_1, V_2) refers to the device in row H_1 and column V_2, and (H_2, V_1) to that in row H_2 and column V_1, then there will be
– a column and a row with one device failed, and
– a column and a row with two devices failed.
Note that the remaining $KD-4$ operational devices are not considered because their failure leads to states that are not in the shortest paths. The above corresponds to the transition from state $(2,0,2,0)$ to state $(1,1,1,1)$, with a transition rate equal to 2λ.

Case 3: Lower path. The second device that fails is one of the $D-1$ operating devices in H_1. This occurs with a transition rate of $(D-1)\lambda$ and results in another vertical RAID-5 array, say column V_2, with one device failed. This corresponds to the transition from state $(1,0,1,0)$ to state $(0,1,2,0)$, as there are now one row with two devices failed and two columns with one device failed in each of them. As the contents of the two failed devices in H_1 are rebuilt in parallel through the corresponding V_1 and V_2 RAID-5 arrays, the transition rate from state $(2,0,0,1)$ back to state $(1,0,1,0)$ is 2μ. If, however, prior to the completion of any of these two rebuilds another of the remaining $2(K-1)$ devices in V_1 and V_2 (say the one in row H_2 and column V_1) fails, then there will be
– a column, namely V_2, with one device failed;
– a row, namely H_1, with two devices failed;
– a row, namely H_2, with one device failed, and
– a column, namely V_1, with two devices failed.
Note that the $(D-2)K$ operational devices in the remaining $D-2$ vertical RAID-5 arrays are not considered because their failure leads to states that are not in the shortest paths. The above corresponds to the transition from state $(0,1,2,0)$ to state $(1,1,1,1)$, with a transition rate equal to $2(K-1)\lambda$.

At state $(1,1,1,1)$, the failed device in H_2 and the failed one in V_2 are recovered through their corresponding RAID-5 arrays. However, the failed device in row H_1 and column V_1 cannot be immediately recovered because both of its corresponding RAID-5 arrays has suffered two device failures. It can only be recovered upon completion of the rebuild of either one of the two previously mentioned devices. In particular, the completion of the rebuild of the failed device in V_2 corresponds to the transition from state $(1,1,1,1)$ to state $(2,0,0,1)$, with a transition rate of μ. The completion of the rebuild of the failed device in H_2 corresponds to the transition from state $(1,1,1,1)$ to state $(0,1,2,0)$, with a transition rate of μ. If, however, prior to the completion of any of these two rebuilds, the device still in operation in row H_2 and column V_2 fails, this leads to data loss, as there will be four failed devices with each of the corresponding RAID-5 arrays having two failed devices. This corresponds to the transition from state $(1,1,1,1)$ to state DL, with a corresponding rate of λ.

The probabilities of the transitions discussed above are given by

$$P_{(1,0,1,0)\to(2,0,0,1)} = \frac{(K-1)\,\lambda}{\mu+(KD-1)\,\lambda}\,, \tag{38}$$

$$P_{(2,0,0,1)\to(1,1,1,1)} = \frac{2(D-1)\,\lambda}{2\,\mu+2(D-1)\,\lambda}\,, \tag{39}$$

$$P_{(1,0,1,0)\to(2,0,2,0)} = \frac{(K-1)(D-1)\,\lambda}{\mu+(KD-1)\,\lambda}\,, \tag{40}$$

$$P_{(2,0,2,0)\to(1,1,1,1)} = \frac{2\,\lambda}{2\,\mu+2\,\lambda}\,, \tag{41}$$

$$P_{(1,0,1,0)\to(0,1,2,0)} = \frac{(D-1)\,\lambda}{\mu+(KD-1)\,\lambda}\,, \tag{42}$$

$$P_{(0,1,2,0)\to(1,1,1,1)} = \frac{2(K-1)\,\lambda}{2\,\mu+2(K-1)\,\lambda}\,, \tag{43}$$

and

$$P_{(1,1,1,1)\to\mathrm{DL}} = \frac{\lambda}{2\,\mu+\lambda}\,. \tag{44}$$

Consequently, the probability of the upper path to data loss, P_u, is given by

$$\begin{aligned} P_u &= P_{(1,0,1,0)\to(2,0,0,1)}\,P_{(2,0,0,1)\to(1,1,1,1)}\,P_{(1,1,1,1)\to\mathrm{DL}} \\ &= \frac{(K-1)\lambda}{\mu+(KD-1)\,\lambda}\cdot\frac{2(D-1)\,\lambda}{2\,\mu+2(D-1)\,\lambda}\cdot\frac{\lambda}{2\,\mu+\lambda}\,, \end{aligned} \tag{45}$$

that of the middle path to data loss, P_m, is given by

$$\begin{aligned} P_m &= P_{(1,0,1,0)\to(2,0,2,0)}\,P_{(2,0,2,0)\to(1,1,1,1)}\,P_{(1,1,1,1)\to\mathrm{DL}} \\ &= \frac{(K-1)(D-1)\lambda}{\mu+(KD-1)\,\lambda}\cdot\frac{2\,\lambda}{2\,\mu+2\,\lambda}\cdot\frac{\lambda}{2\,\mu+\lambda}\,, \end{aligned} \tag{46}$$

and that of the lower path to data loss, P_l, is given by

$$\begin{aligned} P_l &= P_{(1,0,1,0)\to(0,1,2,0)}\,P_{(0,1,2,0)\to(1,1,1,1)}\,P_{(1,1,1,1)\to\mathrm{DL}} \\ &= \frac{(D-1)\,\lambda}{\mu+(KD-1)\,\lambda}\cdot\frac{2(K-1)\,\lambda}{2\,\mu+2(K-1)\,\lambda}\cdot\frac{\lambda}{2\,\mu+\lambda}\,. \end{aligned} \tag{47}$$

By considering (4), equations (45), (46), and (47) yield the following approximations:

$$P_u \;\approx\; \frac{(K-1)\lambda}{\mu}\cdot\frac{2(D-1)\,\lambda}{2\,\mu}\cdot\frac{\lambda}{2\,\mu} \;=\; \frac{(K-1)(D-1)\,\lambda^3}{2\mu^3}\,, \tag{48}$$

$$P_m \;\approx\; \frac{(K-1)(D-1)\lambda}{\mu}\cdot\frac{2\,\lambda}{2\,\mu}\cdot\frac{\lambda}{2\,\mu} \;=\; \frac{(K-1)(D-1)\,\lambda^3}{2\mu^3}\,, \tag{49}$$

and

$$P_l \;\approx\; \frac{(D-1)\,\lambda}{\mu}\cdot\frac{2(K-1)\,\lambda}{2\,\mu}\cdot\frac{\lambda}{2\,\mu} \;=\; \frac{(K-1)(D-1)\,\lambda^3}{2\mu^3}\,. \tag{50}$$

The probability of the shortest paths to data loss, $P_{\text{DL,shortest}}$, is the sum of P_u, P_m and P_l, which by using (21), (48), (49), and (50), yields

$$P_{\text{DL}} \approx P_{\text{DL,shortest}} = P_u + P_m + P_l \approx \frac{3(K-1)(D-1)}{2} \left(\frac{\lambda}{\mu}\right)^3 . \quad (51)$$

Substituting (51) into (3), and considering $N = KD$, yields the approximate MTTDL of the two-dimensional RAID-5 system, $\text{MTTDL}^{(\text{approx})}_{\text{2D-RAID-5}}$, given by

$$\text{MTTDL}^{(\text{approx})}_{\text{2D-RAID-5}} \approx \frac{2\,\mu^3}{3\,K\,(K-1)\,D\,(D-1)\,\lambda^4} . \quad (52)$$

VI. Reliability Comparison

Here, we assess the relative reliability of the various schemes considered. As discussed in Section III, the direct-path-approximation method yields accurate results when the storage devices are highly reliable, that is, when the ratio λ/μ of the mean rebuild time $1/\mu$ to the mean time to failure of a device $1/\lambda$ is very small. We perform a fair comparison by considering systems with the same amount of user data stored under the same storage efficiency. Note that, according to (6) and (22), the storage efficiency of a RAID-5 system cannot be less than 1/2, whereas that of a RAID-51 system is always less than 1/2. Consequently, these two systems cannot be fairly compared.

The MTTDL of a system comprising n_G RAID arrays is assessed by [8]

$$\text{MTTDL}_{\text{sys}} = \frac{\text{MTTDL}_{\text{RAID}}}{n_G} , \quad (53)$$

where $\text{MTTDL}_{\text{RAID}}$ denotes the MTTDL of a single RAID array.

A. RAID-5 vs. RAID-6

Let N_5 and N_6 be the sizes of a RAID-5 and a RAID-6 array, respectively. Assuming the same storage efficiency, we deduce from (6) and (7) that $N_6 = 2\,N_5$. Also, the user data stored in a RAID-6 array can also be stored in a system of $n_G = 2$ RAID-5 arrays. Using (9) and (53), the approximate MTTDL of the RAID-5 system, $\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-5}}$, is obtained as follows:

$$\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-5}} = \frac{\text{MTTDL}^{(\text{approx})}_{\text{RAID-5}}}{2} \quad (54)$$

$$\approx \frac{\mu}{2N_5(N_5-1)\,\lambda^2} . \quad (55)$$

Also, the approximate MTTDL of the RAID-6 system, $\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}}$, is obtained from (14) by setting $N = N_6 = 2N_5$:

$$\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}} = \text{MTTDL}^{(\text{approx})}_{\text{RAID-6}} \quad (56)$$

$$\approx \frac{\mu^2}{2N_5(2N_5-1)(2N_5-2)\,\lambda^3} . \quad (57)$$

Using (55) and (57) yields

$$\frac{\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-5}}}{\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}}} = 2\,(2N_5-1)\cdot\frac{\lambda}{\mu} . \quad (58)$$

Thus, the reliability of the RAID-5 system is less than that of the RAID-6 system by a magnitude dictated by the ratio λ/μ, which is very small.

B. RAID-6 vs. RAID-51

Assuming the same storage efficiency for the two systems, we deduce from (7) and (22) that $N_6 = (4-N_6)\,D$, which is satisfied by $N_6 = D = 3$ only. Furthermore, the user data stored in a RAID-51 array comprised of three pairs can also be stored in system of $n_G = 2$ RAID-6 arrays of size three. The approximate MTTDL of the RAID-6 array, $\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}}$, is obtained from (14) by setting $N = 3$:

$$\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}} \approx \frac{\text{MTTDL}^{(\text{approx})}_{\text{RAID-6}}}{2} \approx \frac{\mu^2}{12\,\lambda^3} . \quad (59)$$

The approximate MTTDL of the RAID-51 system, $\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-51}}$, is obtained from (34) by setting $D = 3$:

$$\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-51}} \approx \frac{\mu^3}{18\,\lambda^4} . \quad (60)$$

Using (59) and (60) yields

$$\frac{\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-6}}}{\text{MTTDL}^{(\text{approx, system})}_{\text{RAID-51}}} = \frac{3}{2}\cdot\frac{\lambda}{\mu} . \quad (61)$$

Thus, the reliability of the RAID-6 system is lower than that of the RAID-51 system by a magnitude dictated by the ratio λ/μ, which is very small.

C. RAID-6 vs. 2D-RAID-5

In general, there are several combinations of N_6, K and D that yield the same storage efficiency for the two systems. From (7) and (37), it follows that

$$\frac{N_6-2}{N_6} = \frac{(K-1)\,(D-1)}{K\,D} , \quad (62)$$

which also implies that

$$D = \frac{(K-1)\,N_6}{2\,K-N_6} . \quad (63)$$

First, we examine whether there is a square 2D-RAID-5 system that has the same storage efficiency as that of a RAID-6 system. Substituting $D = K$ into (62), after some manipulations, yields $N_6 - K = K/(2K-1)$, which is not feasible given that $0 < K/(2K-1) < 1$, for $K > 1$. Therefore, we proceed by assuming, without loss of generality, that $D \geq K+1$. It follows that $K/(K+1) \leq D/(D-1) < 1$, which using (62) implies that $(K-1)/(K+1) \leq (N_6-2)/N_6 < (K-1)/K$, which in turn yields

$$K+1 \leq N_6 \leq 2\,K-1 . \quad (64)$$

Let us consider a system comprised of a single 2D-RAID-5 array with the user data stored in $(K-1)(D-1)$ devices. This data can also be stored in a system comprised of n_G RAID-6 arrays, where

$$n_G \;=\; \frac{(K-1)\,(D-1)}{N_6-2} \;\overset{(63)}{=}\; \frac{K\,(K-1)}{2\,K-N_6}\,. \qquad (65)$$

From (64) and (65), it follows that

$$K \;\leq\; n_G \;\leq\; K\,(K-1)\,. \qquad (66)$$

The values of K, D, and N_6 that that yield the same storage efficiency for the two systems are listed in Table II. We now fix K and consider the following two extreme combinations of the other two parameters, N_6 and D, obtained using (64) and (63), respectively.

1) $N_6 = D = K+1$*:* From (65), it follows that the user data is stored in a system of $n_G = K$ RAID-6 arrays. Using (14) and (53), the approximate MTTDL of the RAID-6 system, $\text{MTTDL}_{\text{RAID-6}}^{\text{(approx, system)}}$, is obtained as follows:

$$\text{MTTDL}_{\text{RAID-6}}^{\text{(approx, system)}} \;=\; \frac{\text{MTTDL}_{\text{RAID-6}}^{\text{(approx)}}}{K} \qquad (67)$$

$$\approx\; \frac{\mu^2}{(K+1)\,K^2\,(K-1)\,\lambda^3}\,. \qquad (68)$$

TABLE II. EQUAL STORAGE EFFICIENCY VALUES ($K \leq 20$)

K	D	N_6	K	D	N_6	K	D	N_6
2	3	3	11	12	12	16	33	22
3	4	4	11	34	17	16	45	24
3	10	5	11	45	18	16	65	26
4	5	5	11	100	20	16	81	27
4	9	6	11	210	21	16	105	28
4	21	7	12	13	13	16	145	29
5	6	6	12	22	16	16	225	30
5	16	8	12	33	18	16	465	31
5	36	9	12	55	20	17	18	18
6	7	7	12	77	21	17	52	26
6	10	8	12	121	22	17	120	30
6	15	9	12	253	23	17	256	32
6	25	10	13	14	14	17	528	33
6	55	11	13	27	18	18	19	19
7	8	8	13	40	20	18	34	24
7	15	10	13	66	22	18	51	27
7	22	11	13	92	23	18	85	30
7	36	12	13	144	24	18	136	32
7	78	13	13	300	25	18	187	33
8	9	9	14	15	15	18	289	34
8	21	12	14	39	21	18	595	35
8	49	14	14	78	24	19	20	20
8	105	15	14	169	26	19	39	26
9	10	10	14	351	27	19	58	29
9	16	12	15	16	16	19	96	32
9	28	14	15	21	18	19	153	34
9	40	15	15	28	20	19	210	35
9	64	16	15	46	23	19	324	36
9	136	17	15	56	24	19	666	37
10	11	11	15	70	25	20	21	21
10	21	14	15	91	26	20	57	30
10	27	15	15	126	27	20	76	32
10	36	16	15	196	28	20	133	35
10	51	17	15	406	29	20	171	36
10	81	18	16	17	17	20	361	38
10	171	19	16	25	20	20	741	39

Also, the approximate MTTDL of the 2D-RAID-5 system, $\text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx,system)}}$, is obtained from (52) as follows:

$$\text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx,system)}} \;=\; \text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx)}} \qquad (69)$$

$$\approx\; \frac{2\,\mu^3}{3\,(K+1)\,K^2\,(K-1)\,\lambda^4}\,. \qquad (70)$$

Using (68) and (70) yields

$$\frac{\text{MTTDL}_{\text{RAID-6}}^{\text{(approx, system)}}}{\text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx,system)}}} \;=\; \frac{3}{2}\,\cdot\,\frac{\lambda}{\mu}\,. \qquad (71)$$

Thus, the reliability of the RAID-6 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the ratio λ/μ, which is very small.

2) $N_6 = 2K-1$ *and* $D = (K-1)(2K-1)$*:* From (65), it follows that the user data is stored in a system of $n_G = K(K-1)$ RAID-6 arrays. Using (14) and (53), the approximate MTTDL of the RAID-6 system is obtained as follows:

$$\text{MTTDL}_{\text{RAID-6}}^{\text{(approx, system)}} \;=\; \frac{\text{MTTDL}_{\text{RAID-6}}^{\text{(approx)}}}{K\,(K-1)} \qquad (72)$$

$$\approx\; \frac{\mu^2}{2\,K\,(K-1)^2\,(2K-1)\,(2K-3)\,\lambda^3}\,. \qquad (73)$$

Also, the approximate MTTDL of the 2D-RAID-5 system is obtained from (52) as follows:

$$\text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx,system)}} \;=\; \text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx)}} \qquad (74)$$

$$\approx\; \frac{2\,\mu^3}{3\,K^2\,(K-1)^2\,(2K-1)\,(2K-3)\,\lambda^4}\,. \qquad (75)$$

Using (73) and (75) yields

$$\frac{\text{MTTDL}_{\text{RAID-6}}^{\text{(approx, system)}}}{\text{MTTDL}_{\text{2D-RAID-5}}^{\text{(approx,system)}}} \;=\; \frac{3\,K}{4}\,\cdot\,\frac{\lambda}{\mu}\,. \qquad (76)$$

Thus, the reliability of the RAID-6 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the ratio λ/μ, which is very small.

D. RAID-5 vs. 2D-RAID-5

In general, there are several combinations of N_5, K and D that yield the same storage efficiency for the two systems. From (6) and (37), it follows that

$$\frac{N_5-1}{N_5} \;=\; \frac{(K-1)\,(D-1)}{K\,D}\,, \qquad (77)$$

or, equivalently,

$$\frac{2\,N_5-2}{2\,N_5} \;=\; \frac{(K-1)\,(D-1)}{K\,D}\,. \qquad (78)$$

From (62) and (78), it follows that the (K, D, N_5) combinations correspond to the the (K, D, N_6) ones, where N_6 is even and $N_5 = N_6/2$. From Table II, we deduce that the first two combinations are the following: $K=3$, $D=4$, $N_5=2$,

and $K = 4$, $D = 9$, $N_5 = 3$. From (62), (64), and (78), it follows that

$$\frac{K+1}{2} \;\leq\; N_5 \;\leq\; K-1\,. \tag{79}$$

We now fix K to be an odd number and consider the following two extreme combinations regarding the other two parameters, D and N.

1) $N_5 = (K+1)/2$ *and* $D = K+1$*:* In this case, the user data is stored in $(K-1)(D-1) = K(K-1)$ devices in the 2D-RAID-5 system, which implies that this data can also be stored in a system of $n_G = 2K$ RAID-5 arrays. Using (9) and (53), the approximate MTTDL of the RAID-5 system is obtained as follows:

$$\text{MTTDL}^{\text{(approx, system)}}_{\text{RAID-5}} \;=\; \frac{\text{MTTDL}^{\text{(approx)}}_{\text{RAID-5}}}{2\,K} \tag{80}$$

$$\approx\; \frac{2\,\mu}{(K+1)\,K\,(K-1)\,\lambda^2}\,. \tag{81}$$

Also, the approximate MTTDL of the 2D-RAID-5 system is obtained from (52) as follows:

$$\text{MTTDL}^{\text{(approx,system)}}_{\text{2D-RAID-5}} \;=\; \text{MTTDL}^{\text{(approx)}}_{\text{2D-RAID-5}} \tag{82}$$

$$\approx\; \frac{2\,\mu^3}{3\,(K+1)\,K^2\,(K-1)\,\lambda^4}\,. \tag{83}$$

Using (81) and (83) yields

$$\frac{\text{MTTDL}^{\text{(approx, system)}}_{\text{RAID-5}}}{\text{MTTDL}^{\text{(approx,system)}}_{\text{2D-RAID-5}}} \;=\; 3\,K\,\cdot\,\left(\frac{\lambda}{\mu}\right)^2\,. \tag{84}$$

Thus, the reliability of the RAID-5 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the square of the ratio λ/μ, which is very small.

2) $N_5 = K-1$ *and* $D = (K-1)^2$*:* In this case, the user data is stored in $(K-1)(D-1) = (K-1)[(K-1)^2 - 1] = K(K-1)(K-2)$ devices in the 2D-RAID-5 system, which implies that this data can also be stored in a system of $n_G = K(K-1)$ RAID-5 arrays. Using (9) and (53), the approximate MTTDL of the RAID-5 system is obtained as follows:

$$\text{MTTDL}^{\text{(approx, system)}}_{\text{RAID-5}} \;=\; \frac{\text{MTTDL}^{\text{(approx)}}_{\text{RAID-5}}}{K\,(K-1)} \tag{85}$$

$$\approx\; \frac{\mu}{K\,(K-1)^2\,(K-2)\,\lambda^2}\,. \tag{86}$$

Also, the approximate MTTDL of the 2D-RAID-5 system is obtained from (52) as follows:

$$\text{MTTDL}^{\text{(approx,system)}}_{\text{2D-RAID-5}} \;=\; \text{MTTDL}^{\text{(approx)}}_{\text{2D-RAID-5}} \tag{87}$$

$$\approx\; \frac{2\mu^3}{3\,K^2\,(K-1)^3\,(K-2)\,\lambda^4}\,. \tag{88}$$

Using (86) and (88) yields

$$\frac{\text{MTTDL}^{\text{(approx, system)}}_{\text{RAID-5}}}{\text{MTTDL}^{\text{(approx,system)}}_{\text{2D-RAID-5}}} \;=\; \frac{3\,K\,(K-1)}{2}\,\cdot\,\left(\frac{\lambda}{\mu}\right)^2\,. \tag{89}$$

Thus, the reliability of the RAID-5 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the square of the ratio λ/μ, which is very small.

E. Erasure Codes: Maximum Distance Separable (MDS) vs. non-MDS

An (l, m)-erasure code is a mapping from l user data symbols (or blocks) to a set of m ($> l$) symbols, called a codeword, in such a way that some subsets of the m blocks of the codeword can be used to decode the l user data blocks. Maximum distance separable (MDS) erasure codes have the property that *any* l of the m symbols can be used to decode a codeword. Examples of such codes include RAID-5 (an $(N, N+1)$-MDS code), RAID-6 (an $(N, N+2)$-MDS code), and r-way replication (an $(1, r)$-MDS code). As an MDS erasure code can decode data from any l of the m codeword symbols, a system employing such a code can sustain up to $(m-l)$ device failures. This implies that the most probable path to data loss has exactly $(m-l)$ 'hops', starting from the first-device failure and ending at data loss. As each hop has a probability proportional to λ/μ (when $\lambda/\mu \ll 1$), the resulting P_{DL} is proportional to $(\lambda/\mu)^{(m-l)}$. This can be seen from the P_{DL} equations (11) and (18) for RAID-5 and RAID-6, respectively.

In contrast, erasure codes that do not have the MDS property may not be able to sustain $(m-l)$ device failures. Examples of such non-MDS codes include RAID-51 (a $(D-1, 2D)$ non-MDS code) and 2D-RAID-5 (a $((D-1)\,(K-1), D\,K)$ non-MDS code). Both these non-MDS codes can sustain any three device failures; however, the fact that they have considerably higher redundancy may allow them to sustain certain other subsets of more than three devices, e.g., the failure of an entire row or column. Note that $(m-l)$ is equal to $D+1$ (≥ 3) and $D+K-1$ (≥ 3) for the RAID-51 and 2D-RAID-5, respectively, and therefore could be much higher than three for larger values of D and K. Despite this, these codes can sustain only up to three arbitrary device failures. This implies that the most probable path to data loss for RAID-51 and 2D-RAID-5 has exactly three hops, starting from the first-device failure and ending at data loss, and hence the resulting P_{DL} is proportional to $(\lambda/\mu)^3$. This can be seen from the P_{DL} equations (33) and (51) for RAID-51 and 2D-RAID-5, respectively.

Although it may seem that non-MDS codes are not useful because they provide a lower reliability than their MDS equivalents for the same storage efficiency, they have an advantage over MDS codes in the presence of correlated device failures that makes them valuable in practice. To see this, consider a system employing RAID-51, where each RAID-5 array is across D devices belonging to a different storage node. Such a system can sustain the failure of any node even though a node failure implies that all D devices belonging to that node are considered failed. Therefore, by carefully selecting the non-MDS code and the data placement, data can be protected from correlated failures.

VII. Most Probable Paths to Data Loss

In the preceding sections, we demonstrated that the reliability of systems comprised of highly reliable devices can be well approximated by considering the most likely paths that lead to data loss, namely, the shortest paths. These paths represent the smallest number of successive device failures that lead to data loss.

Here, we demonstrate that in general the shortest paths may not be the most likely paths that lead to data loss. We therefore extend our methodology to account for the most probable paths that lead to data loss. Clearly, the most probable paths are direct paths to data loss, i.e., paths without loops, but they may not be the shortest ones.

A. Unrecoverable or Latent Errors

When the storage devices are disks, in addition to disk failures, data loss may occur owing to errors in individual disk sectors that cannot be recovered with a reread or the sector-based error-correction code (ECC). Such media-related errors are referred to as unrecoverable or latent sector errors [6][8][11][18]. We proceed by considering the family of devices that exhibit such behavior. The occurrence of unrecoverable sector errors is particularly problematic when combined with device failures. For example, if a device fails in a RAID-5 array, the rebuild process must read all the data on the remaining devices to reconstruct the data lost. Consequently, an unrecoverable error on any of the operational devices would result in an irrecoverable loss of data. A similar problem occurs when two devices fail in a RAID-6 scheme. In this case, any unrecoverable sector errors encountered on the good devices during the rebuild process also lead to data loss.

The system reliability depends on the probability P_s of an unrecoverable error on a typical sector [6], as well as the sector size, S, and the device capacity, C_d. It in fact depends on the number of sectors in a device, n_s, which is given by

$$n_s \;=\; \frac{C_d}{S}\;. \tag{90}$$

The notation used is summarized in Table I. The parameters are divided according to whether they are independent or derived and listed in the upper and the lower part of the table, respectively.

B. A RAID-6 Array Under Latent Errors

The effect of unrecoverable sector errors in a RAID-6 system was analyzed in [6]. We proceed by briefly reviewing the Markov model developed to characterize the system behavior and capture the corresponding state transitions. The corresponding CTMC model shown in Fig. 7 is obtained from [6, Fig. 7] by setting $\mu_1 = \mu_2 = \mu$. The numbered states of the Markov model represent the number of failed devices. The DF and UF states represent a data loss due to a device failure and an unrecoverable sector failure, respectively.

When the first device fails, the array enters degraded mode, which corresponds to the transition from state 0 to state 1, with a transition rate of $N\lambda$. The rebuild of a sector of the failed device is performed based on up to $N-1$ corresponding

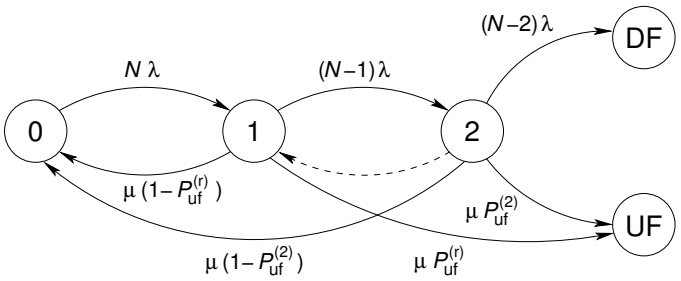


Figure 7. Reliability model for a RAID-6 array under latent errors.

sectors residing on the remaining devices. The rebuild fails when two or more of these sectors are in error. Consequently, the probability P_{recf} that a given sector of the failed device cannot be reconstructed is equal to the probability that two or more of the corresponding sectors residing in the remaining devices are in error and is given by [9, Eq. (12)]:

$$P_{\text{recf}} \;\approx\; \binom{N-1}{2} P_s^2\;. \tag{91}$$

Remark 3: Equation (91) is obtained from

$$P_{\text{recf}} \;=\; \sum_{j=2}^{N-1} \binom{N-1}{j} P_s^j (1-P_s)^{N-1-j} \;\approx\; \binom{N-1}{2} P_s^2\;, \tag{92}$$

which is derived from Equation (47) of [6] by setting $P_{\text{seg}} = P_s$. Note that (92) accounts for all combinations of sector errors that cause the rebuild of the given sector to fail. However, the **most probable combinations of sector errors** are those that involve only two sectors in error, which is **the least number of sectors in error** that cause the rebuild to fail. These combinations yield the approximation given in (91).

The probability that an unrecoverable failure occurs in degraded mode because the rebuild of the failed device cannot be completed, $P_{\text{uf}}^{(\text{r})}$, is then given by [9, Eq. (9)]:

$$P_{\text{uf}}^{(\text{r})} = 1 - \left[1 - \binom{N-1}{2} P_s^2\right]^{n_s} \;\approx\; n_s \binom{N-1}{2} P_s^2\,, \tag{93}$$

where n_s is the number of sectors in a device.

The system exits from state 1 owing to either another device failure or completion of the rebuild. The former event is represented by the state transition from state 1 to state 2 with a rate of $(N-1)\lambda$. The latter event occurs with a rate of μ and includes two possibilities: a failed rebuild (due to an unrecoverable failure) with probability $P_{\text{uf}}^{(\text{r})}$ and a successful rebuild with probability $1 - P_{\text{uf}}^{(\text{r})}$. The former event is represented by the state transition from state 1 to state UF with a rate of $\mu P_{\text{uf}}^{(\text{r})}$, and the latter one is represented by the state transition from state 1 to state 0 with a rate of $\mu(1-P_{\text{uf}}^{(\text{r})})$.

When a second device fails (state transition from state 1 to state 2), the RAID-6 array enters the critical mode as an additional device failure leads to data loss. The rebuild of the two failed devices is performed based on the remaining $N-2$ devices. The rebuild fails if any sector of these $N-2$ devices is in error. The probability of this event, $P_{\text{uf}}^{(2)}$, is given by [9,

Eq. (10)]:

$$P_{\text{uf}}^{(2)} = 1 - (1 - P_s)^{(N-2)\,n_s} \;\approx\; (N-2)\,n_s\,P_s\;. \qquad (94)$$

The system exits from state 2 owing to either another device failure or completion of the rebuild. The former event is represented by the state transition from state 2 to state DF with a rate of $(N-2)\lambda$. The latter event occurs with a rate of μ and includes two possibilities: a failed rebuild (due to an unrecoverable failure) with probability $P_{\text{uf}}^{(2)}$ and a successful rebuild with probability $1 - P_{\text{uf}}^{(2)}$. The former event is represented by the state transition from state 2 to state UF with a rate of $\mu P_{\text{uf}}^{(2)}$, and the latter one by the state transition from state 2 to state 0 with a rate of $\mu(1 - P_{\text{uf}}^{(2)})$.

We now proceed to show how the approximate MTTDL of the system can be derived in a straightforward manner by appropriate application of the direct-path-approximation technique. The transition from state 0 to state 1 represents the first device failure. The shortest path to data loss involves a subsequent transition from state 1 to UF, with a corresponding probability $P_{1\to\text{UF}}$ given by

$$P_{1\to\text{UF}} = \frac{\mu\,P_{\text{uf}}^{(\text{r})}}{\mu + (N-1)\,\lambda} \overset{(4)}{\approx} P_{\text{uf}}^{(\text{r})}\;. \qquad (95)$$

Note that there are two additional non-shortest paths, namely $1 \to 2 \to$ DF and $1 \to 2 \to$ UF, each involving two transitions, that lead to data loss. The probability $P_{1\to 2}$ of the transition from state 1 to state 2 is given by (15)

$$P_{1\to 2} = \frac{(N-1)\,\lambda}{\mu + (N-1)\,\lambda} \approx \frac{(N-1)\,\lambda}{\mu}\;. \qquad (96)$$

The probability $P_{2\to\text{DF}}$ of the transition from state 2 to state DF is given by (16)

$$P_{2\to\text{DF}} = \frac{(N-2)\,\lambda}{\mu + (N-2)\,\lambda} \approx \frac{(N-2)\,\lambda}{\mu}\;. \qquad (97)$$

Also, the probability $P_{2\to\text{UF}}$ of the transition from state 2 to state UF is given by

$$P_{2\to\text{UF}} = \frac{\mu\,P_{\text{uf}}^{(2)}}{\mu + (N-2)\,\lambda} \approx P_{\text{uf}}^{(2)}\;. \qquad (98)$$

Consequently, the probabilities of the two paths to data loss, $P_{1\to 2\to\text{DF}}$ and $P_{1\to 2\to\text{UF}}$, are given by

$$P_{1\to 2\to\text{DF}} = P_{1\to 2}\,P_{2\to\text{DF}} \approx \frac{(N-1)(N-2)\,\lambda^2}{\mu^2}\;, \qquad (99)$$

and

$$P_{1\to 2\to\text{UF}} = P_{1\to 2}\,P_{2\to\text{UF}} \approx \frac{(N-1)\,\lambda\,P_{\text{uf}}^{(2)}}{\mu}\;. \qquad (100)$$

Thus, the probability of the direct paths to data loss that cross state 2, $P_{1\to 2\to\text{DL}}$, is given by

$$\begin{aligned} P_{1\to 2\to\text{DL}} &= P_{1\to 2\to\text{DF}} + P_{1\to 2\to\text{UF}} \\ &\approx \frac{(N-1)\,\lambda}{\mu}\left[\frac{(N-2)\,\lambda}{\mu} + P_{\text{uf}}^{(2)}\right]\;. \end{aligned} \qquad (101)$$

From (95) and (99), and using (93), it follows that the ratio of the probabilities of the two paths $1 \to$ UF and $1 \to 2 \to$ DF is given by

$$\frac{P_{1\to\text{UF}}}{P_{1\to 2\to\text{DF}}} \approx \frac{1}{2}\,n_s\left(\frac{\lambda}{\mu}\right)^{-2} P_s^2\;. \qquad (102)$$

Clearly, for very small values of P_s, this ratio is also very small, which implies that the path $1 \to 2 \to$ DF is significantly more probable than the shortest path $1 \to$ UF. In contrast to the cases considered in Sections IV and V, where the shortest paths were also the most probable ones, here the shortest path is not. **Therefore, we need to enhance the notion of the shortest paths by considering the most probable ones.**

In view of this finding, we proceed by assessing the system reliability in a region of small values of P_s in which the path $1 \to 2 \to$ DF is the most probable one. From (99) and (100), and using (94), it follows that the path $1 \to 2 \to$ DF is the most probable one when P_s is in region A obtained by

$$\frac{\lambda}{\mu} \gg n_s\,P_s \quad\Leftrightarrow\quad P_s \ll \frac{1}{n_s}\cdot\frac{\lambda}{\mu} \quad \text{(region A)}\;. \qquad (103)$$

Subsequently, for $P_s > \lambda/(n_s\,\mu)$, that is, when P_s is in region B, the other path to data loss, $1 \to 2 \to$ UF, becomes the most probable one. In fact, when $P_{\text{uf}}^{(2)}$ approaches one, the P_{DL} and MTTDL no longer depend on P_s. Owing to (94), this occurs when P_s is in region C obtained by

$$P_{\text{uf}}^{(2)} \approx 1 \quad\Leftrightarrow\quad P_s \gtrapprox \frac{1}{(N-2)\,n_s} \quad \text{(region C)}\;. \qquad (104)$$

Note that in region C, the path $1 \to 2 \to$ UF is also more probable than the shortest path $1 \to$ UF. Consequently, from (95) and (100), setting $P_{\text{uf}}^{(2)} = 1$, and using (93), it follows that in region C it holds that

$$\begin{aligned} n_s\binom{N-1}{2}P_s^2 &\gtrapprox (N-1)\,\frac{\lambda}{\mu} \\ \Leftrightarrow\quad P_s &\gtrapprox \sqrt{\frac{2\,\lambda}{(N-2)\,n_s\,\mu}} \quad \text{(region C)}\;. \end{aligned} \qquad (105)$$

Subsequently, for $P_s \gtrapprox \sqrt{2\,\lambda/[(N-2)\,n_s\,\mu]}$, that is, when P_s is in region D, the shortest path to data loss $1 \to$ UF, becomes the most probable one. In fact, when $P_{\text{uf}}^{(\text{r})}$ approaches one, P_{DL} and MTTDL no longer depend on P_s. Owing to (93), this occurs when P_s is in region E obtained by

$$P_{\text{uf}}^{(\text{r})} \approx 1 \;\Leftrightarrow\; P_s \gtrapprox \sqrt{\frac{2}{(N-1)\,(N-2)\,n_s}} \quad \text{(region E)}\;. \qquad (106)$$

Combining the preceding, (101) yields

$$P_{\text{DL}} \approx \begin{cases} \frac{(N-1)(N-2)\,\lambda^2}{\mu^2}\,, & \text{A}: P_s \ll \frac{\lambda}{n_s\,\mu} \\ \frac{(N-1)(N-2)\,\lambda\,n_s}{\mu}\,P_s\,, & \text{B}: \frac{\lambda}{n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{(N-2)\,n_s} \\ \frac{(N-1)\,\lambda}{\mu}\,, & \text{C}: \frac{1}{(N-2)\,n_s} \lessapprox P_s \lessapprox \sqrt{\frac{2\,\lambda}{(N-2)\,n_s\,\mu}} \\ \frac{(N-1)(N-2)\,n_s}{2}\,P_s^2\,, & \\ & \text{D}: \sqrt{\frac{2\,\lambda}{(N-2)\,n_s\,\mu}} \lessapprox P_s \lessapprox \sqrt{\frac{2}{(N-1)(N-2)\,n_s}} \\ 1\,, & \text{E}: \sqrt{\frac{2}{(N-1)(N-2)\,n_s}} \lessapprox P_s \le 1\,. \end{cases} \tag{107}$$

Substituting (107) into (3) yields

$$\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})} \approx \begin{cases} \frac{\mu^2}{N(N-1)(N-2)\,\lambda^3}\,, & \text{A}: P_s \ll \frac{\lambda}{n_s\,\mu} \\ \frac{\mu}{N(N-1)(N-2)\,\lambda^2\,n_s}\,P_s^{-1}\,, & \text{B}: \frac{\lambda}{n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{(N-2)\,n_s} \\ \frac{\mu}{N(N-1)\,\lambda^2}\,, & \text{C}: \frac{1}{(N-2)\,n_s} \lessapprox P_s \lessapprox \sqrt{\frac{2\,\lambda}{(N-2)\,n_s\,\mu}} \\ \frac{2}{N(N-1)(N-2)\,\lambda\,n_s}\,P_s^{-2}\,, & \\ & \text{D}: \sqrt{\frac{2\,\lambda}{(N-2)\,n_s\,\mu}} \lessapprox P_s \lessapprox \sqrt{\frac{2}{(N-1)(N-2)\,n_s}} \\ \frac{1}{N\,\lambda}\,, & \text{E}: \sqrt{\frac{2}{(N-1)(N-2)\,n_s}} \lessapprox P_s \le 1\,. \end{cases} \tag{108}$$

Remark 4: The preceding expression specifies three regions, namely A, D, and E, where the MTTDL is independent of P_s. This corresponds to three plateaus, as shown in [6, Fig. 9(c)].

Remark 5: Depending on the parameter values, some of the regions may vanish. For instance, region C vanishes when $\sqrt{2\,\lambda/[(N-2)\,n_s\,\mu]} < 1/[(N-2)n_s]$, or equivalently, $2\,(N-2)\,n_s\,\lambda/\mu < 1$.

Remark 6: The most probable paths are obtained by first identifying all direct paths to data loss, i.e., paths to data loss without loops, then evaluating their probability of occurrence, and finally selecting the most probable ones. Nevertheless, the MTTDL can be obtained analytically by considering all direct paths, which are more probable than those having loops. Therefore, it suffices to simply sum the probabilities of all direct paths to data loss to obtain the P_{DL}, and in turn, the MTTDL. The paths with the highest probabilities naturally dominate the sum and therefore implicitly determine the system reliability.

The direct paths to data loss are the following: $1 \to \text{UF}$, $1 \to 2 \to \text{DF}$ and $1 \to 2 \to \text{UF}$. From (95), (99), and (100), it follows that

$$\begin{aligned} P_{\text{DL}} &\approx P_{1\to\text{UF}} + P_{1\to 2\to\text{DF}} + P_{1\to 2\to\text{UF}} \\ &\approx \min\left(1,\; P_{\text{uf}}^{(\text{r})} + \frac{(N-1)\,\lambda}{\mu}\left[\frac{(N-2)\,\lambda}{\mu} + P_{\text{uf}}^{(2)}\right]\right)\,. \end{aligned} \tag{109}$$

Note that the expression in (109) may exceed one and, as it expresses the probability of data loss, needs to be truncated to one. Assuming that the expression does not exceed one, substituting (109) into (3) yields

$$\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})} \approx \frac{\mu^2}{N\{\mu^2\,P_{\text{uf}}^{(\text{r})} + (N-1)\,\lambda\,[(N-2)\,\lambda + \mu\,P_{\text{uf}}^{(2)}]\}\,\lambda}\,, \tag{110}$$

with $P_{\text{uf}}^{(\text{r})}$ and $P_{\text{uf}}^{(2)}$ given by (93) and (94), respectively.

We verify that by setting $\mu_1 = \mu_2 = \mu$, and using (4), the exact MTTDL expression given in [6, Eq. (52)] yields $\text{MTTDL} \approx \tau_0 \approx \mu^2/(N\lambda V)$, which after some manipulations gives the same result as in (110).

Remark 7: If the transition from state 2 to state 0 were not to state 0 but to state 1 instead, as shown in Fig. 7 by the dashed arrow, the corresponding MTTDL could still be approximated by (108) and (110) because the expressions for $P_{1\to\text{UF}}$, $P_{1\to 2\to\text{DF}}$, $p_{1\to 2\to\text{UF}}$, and P_{DL} given by (95), (99), (100), and (109) respectively, would still hold.

C. A Two-Dimensional RAID-5 Array Under Latent Errors

We consider the two-dimensional RAID-5 array analyzed in Section V-B in which the devices may contain unrecoverable or latent sector errors. We consider the probability P_s of an unrecoverable sector error to be small. This in turn implies that when considering the cases that lead to unsuccessful rebuilds of sectors residing on failed devices, and according to Remark 3, it suffices to consider only the most probable ones, which are those that involve the least number of sectors in error.

We now proceed to evaluate the MTTDL using the most-probable-path approximation. The transition from state $(0,0,0,0)$ to state $A \equiv (1,0,1,0)$ represents the first device failure as shown in in Fig. 8. The rebuild of the failed device can be performed using either the corresponding horizontal RAID-5 array H_1 or the corresponding vertical RAID-5 array V_1.

The rebuild of a given sector, say SEC, of the failed device fails when there are at least three corresponding sectors on other devices in error, with the four sectors (including SEC) occurring in a constellation of two horizontal rows (horizontal RAID-5 stripes) and two vertical columns (vertical RAID-5 stripes). Note that the sector in the constellation that resides opposite to SEC can be located in any of the $(K-1)(D-1)$ devices that are not in H_1 and V_1. Consequently, the probability P_{A} that SEC cannot be reconstructed is given by

$$P_{\text{sf}} \;\approx\; 1-(1-P_s^3)^{(K-1)(D-1)} \;\approx\; (K-1)(D-1)\,P_s^3\,. \tag{111}$$

As the failed device contains n_s sectors, the probability that an unrecoverable failure occurs because its rebuild cannot be completed, P_{A}, is then given by

$$P_{\text{A}} \approx 1-(1-P_{\text{sf}})^{n_s} \overset{(111)}{\approx} 1-(1-P_s^3)^{(K-1)(D-1)n_s} \tag{112}$$

$$\approx (K-1)(D-1)\,n_s\,P_s^3\,. \tag{113}$$

When a second device fails, the 2D-RAID-5 array enters either state $B \equiv (2,0,0,1)$, state $C \equiv (2,0,2,0)$ or state $D \equiv (0,1,2,0)$, as shown in Fig. 8. When the system is in state

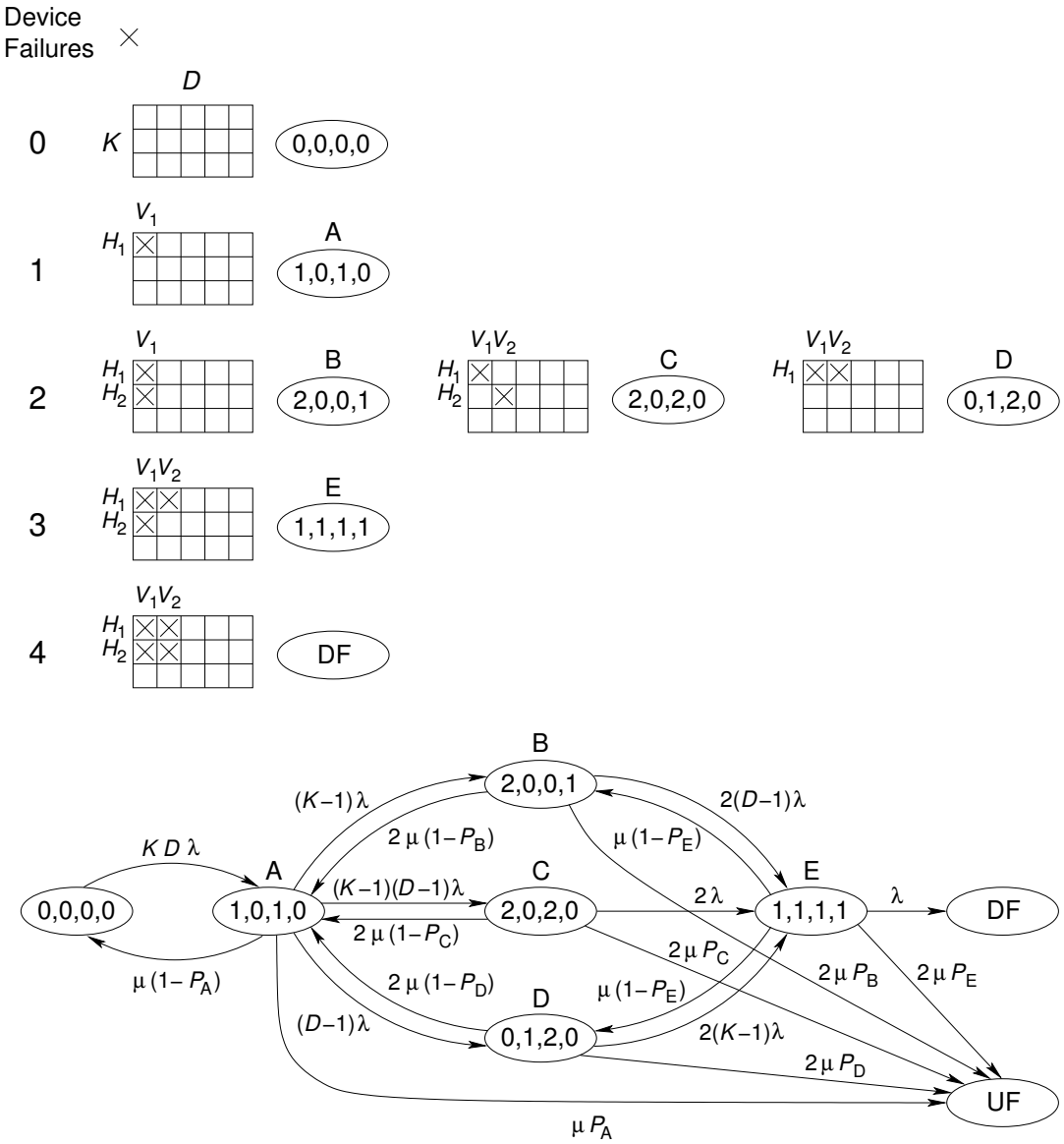


Figure 8. Reliability model for a 2D-RAID-5 array under latent errors.

B, the contents of the two failed devices in V_1 are rebuilt in parallel through the corresponding H_1 and H_2 RAID-5 arrays. The rebuild fails when there is a pair of corresponding sectors in error in some of the $D-1$ pairs of devices in the H_1 and H_2 RAID-5 arrays. As the number of such pairs is equal to $(D-1)\,n_s$, the probability of an unsuccessful rebuild, $P_{\rm B}$, is given by

$$P_{\rm B} \approx 1-(1-P_s^2)^{(D-1)n_s} \approx (D-1)\,n_s\,P_s^2\,. \qquad (114)$$

When the system is in state C, the contents of the two failed devices are rebuilt in parallel through the corresponding horizontal or vertical RAID-5 arrays. The rebuild fails when there is a pair of corresponding sectors in error in the (H_1, V_2) and (H_2, V_1) devices, where (H_1, V_2) refers to the device in row H_1 and column V_2, and (H_2, V_1) to that in row H_2 and column V_1. As the number of such pairs is equal to n_s, the probability of an unsuccessful rebuild, $P_{\rm C}$, is given by

$$P_{\rm C} \approx 1-(1-P_s^2)^{n_s} \approx n_s\,P_s^2\,. \qquad (115)$$

When the system is in state D, the contents of the two failed devices in H_1 are rebuilt in parallel through the corresponding V_1 and V_2 RAID-5 arrays. The rebuild fails when there is a pair of corresponding sectors in error in some of the $K-1$ pairs of devices in the V_1 and V_2 RAID-5 arrays. As the number of such pairs is equal to $(K-1)\,n_s$, the probability of an unsuccessful rebuild, $P_{\rm D}$, is given by

$$P_{\rm D} \approx 1-(1-P_s^2)^{(K-1)n_s} \approx (K-1)\,n_s\,P_s^2\,. \qquad (116)$$

When the system is in state $E \equiv (1,1,1,1)$, it suffices one latent sector error in the device in row H_2 and column V_2 to cause the rebuild to fail. As this device contains n_s sectors, the probability that an unrecoverable failure occurs because the rebuild of the failed devices cannot be completed, $P_{\rm E}$, is then given by

$$P_{\rm E} \approx 1-(1-P_s)^{n_s} \approx n_s\,P_s\,. \qquad (117)$$

As shown in Fig. 8, the shortest path from state $(1,0,1,0)$

to data loss involves a subsequent transition from state $(1,0,1,0)$ to UF, with a corresponding probability, $P_{\mathrm{A}\to\mathrm{UF}}$, given by

$$P_{\mathrm{A}\to\mathrm{UF}} = \frac{\mu\, P_{\mathrm{A}}}{\mu + (K\,D-1)\,\lambda} \approx P_{\mathrm{A}}\ . \tag{118}$$

Next, we consider the three additional direct, two-hop non-shortest paths, namely, A → B → UF, A → C → UF, and A → D → UF, each involving two transitions, that lead to data loss. We proceed to evaluate the probabilities of their occurrence. From Fig. 8, it follows that

$$P_{\mathrm{B}\to\mathrm{UF}} = \frac{2\,\mu\, P_{\mathrm{B}}}{2\,\mu + 2\,(D-1)\,\lambda} \approx P_{\mathrm{B}}\ , \tag{119}$$

$$P_{\mathrm{C}\to\mathrm{UF}} = \frac{2\,\mu\, P_{\mathrm{C}}}{2\,\mu + 2\,\lambda} \approx P_{\mathrm{C}}\ , \tag{120}$$

and

$$P_{\mathrm{D}\to\mathrm{UF}} = \frac{2\,\mu\, P_{\mathrm{D}}}{2\,\mu + 2\,(K-1)\,\lambda} \approx P_{\mathrm{D}}\ . \tag{121}$$

From (38) and (119), and using (114), it follows that

$$P_{\mathrm{A}\to\mathrm{B}\to\mathrm{UF}} = P_{\mathrm{A}\to\mathrm{B}}\, P_{\mathrm{B}\to\mathrm{UF}} \approx (K-1)\,\frac{\lambda}{\mu}\,P_{\mathrm{B}} \tag{122}$$

$$\approx (K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2\ . \tag{123}$$

From (40) and (120), and using (115), it follows that

$$P_{\mathrm{A}\to\mathrm{C}\to\mathrm{UF}} = P_{\mathrm{A}\to\mathrm{C}}\, P_{\mathrm{C}\to\mathrm{UF}} \approx (K-1)\,(D-1)\,\frac{\lambda}{\mu}\,P_{\mathrm{C}} \tag{124}$$

$$\approx (K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2\ . \tag{125}$$

From (42) and (121), and using (116), it follows that

$$P_{\mathrm{A}\to\mathrm{D}\to\mathrm{UF}} = P_{\mathrm{A}\to\mathrm{D}}\, P_{\mathrm{D}\to\mathrm{UF}} \approx (D-1)\,\frac{\lambda}{\mu}\,P_{\mathrm{D}} \tag{126}$$

$$\approx (K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2\ . \tag{127}$$

Next, we consider the six additional direct non-shortest paths, namely, A → B → E → DF, A → C → E → DF, A → D → E → DF, A → B → E → UF, A → C → E → UF, and A → D → E → UF, each involving three transitions, that lead to data loss.

The probabilities of occurrence of the first three paths are the corresponding probabilities of these paths in the absence of sector errors given by (48), (49), and (50), respectively, that is,

$$P_{\mathrm{A}\to\mathrm{B}\to\mathrm{E}\to\mathrm{DF}} \approx P_{\mathrm{A}\to\mathrm{C}\to\mathrm{E}\to\mathrm{DF}} \approx P_{\mathrm{A}\to\mathrm{D}\to\mathrm{E}\to\mathrm{DF}} \approx \frac{(K-1)\,(D-1)}{2}\,\left(\frac{\lambda}{\mu}\right)^3\ . \tag{128}$$

Thus,

$$P_{\mathrm{A}\to\mathrm{E}\to\mathrm{DF}} = P_{\mathrm{A}\to\mathrm{B}\to\mathrm{E}\to\mathrm{DF}} + P_{\mathrm{A}\to\mathrm{C}\to\mathrm{E}\to\mathrm{DF}} + P_{\mathrm{A}\to\mathrm{D}\to\mathrm{E}\to\mathrm{DF}} \approx \frac{3\,(K-1)\,(D-1)}{2}\,\left(\frac{\lambda}{\mu}\right)^3\ . \tag{129}$$

According to (44), it holds that

$$P_{\mathrm{E}\to\mathrm{DF}} \approx \frac{\lambda}{2\,\mu}\ . \tag{130}$$

From (129) and (130), we deduce that

$$P_{\mathrm{A}\to\mathrm{E}} = 3\,(K-1)\,(D-1)\,\left(\frac{\lambda}{\mu}\right)^2\ . \tag{131}$$

Also, it holds that

$$P_{\mathrm{E}\to\mathrm{UF}} = \frac{2\,\mu\, P_{\mathrm{E}}}{2\,\mu+\lambda} \approx P_{\mathrm{E}}\ . \tag{132}$$

Combining (131) and (132) yields

$$P_{\mathrm{A}\to\mathrm{E}\to\mathrm{UF}} = P_{\mathrm{A}\to\mathrm{E}}\, P_{\mathrm{E}\to\mathrm{UF}} \approx 3\,(K-1)\,(D-1)\,\left(\frac{\lambda}{\mu}\right)^2 P_{\mathrm{E}}\ . \tag{133}$$

Thus, the probability of the direct paths to data loss that cross state E is given by

$$P_{\mathrm{A}\to\mathrm{E}\to\mathrm{DL}} = P_{\mathrm{A}\to\mathrm{E}\to\mathrm{DF}} + P_{\mathrm{A}\to\mathrm{E}\to\mathrm{UF}} \approx 3\,(K-1)\,(D-1)\,\left(\frac{\lambda}{\mu}\right)^2\,\left(\frac{\lambda}{2\,\mu}+P_{\mathrm{E}}\right)\ . \tag{134}$$

From (118) and (129), and using (113), it follows that the ratio of the probabilities of the two paths A → UF and A → E → DF is given by

$$\frac{P_{\mathrm{A}\to\mathrm{UF}}}{P_{\mathrm{A}\to\mathrm{E}\to\mathrm{DF}}} \approx \frac{2}{3}\,n_s\,\left(\frac{\lambda}{\mu}\right)^{-3} P_s^3\ . \tag{135}$$

Clearly, for very small values of P_s, this ratio is also very small, which implies that the path A → E → DF is significantly more probable than the shortest path A → UF.

In view of this finding, we proceed by assessing the system reliability in a region of small values of P_s in which the path A → E → DF is the most probable one. Using (117), it follows that the first term of the summation in (134) dominates when P_s is in region H obtained by

$$\frac{\lambda}{2\mu} \gg n_s\, P_s \quad\Leftrightarrow\quad P_s \ll \frac{1}{2}\cdot\frac{1}{n_s}\cdot\frac{\lambda}{\mu} \quad \text{(region H)}\ . \tag{136}$$

Subsequently, for $P_s > \lambda/(2\,n_s\,\mu)$, that is, when P_s is in region I, the other path to data loss A → E → UF becomes the most probable one. In fact, when P_{E} approaches one, the P_{DL} and MTTDL no longer depend on P_s. Owing to (117), this occurs when P_s is in region J obtained by

$$P_{\mathrm{E}} \approx 1 \quad\Leftrightarrow\quad P_s \gtrapprox \frac{1}{n_s} \quad \text{(region J)}\ . \tag{137}$$

Note that in region J, the path A → E → UF is also more probable than any of the paths A → B → UF, A → C → UF, and A → D → UF. For small values of P_s, according to (123), (125), and (127), these two-hop paths are equally likely to occur. Therefore, the probability $P_{\mathrm{A}\to\mathrm{X}\to\mathrm{UF}}$ of a transition from state A to state UF through some other state X (X = B or C or D) is given by

$$P_{\mathrm{A}\to\mathrm{X}\to\mathrm{UF}} = 3\,(K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2\ . \tag{138}$$

Consequently, from (133) and (138), and setting $P_{\rm E} = 1$, it follows that in region J it holds that

$$3\,(K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2 \;\lessapprox\; 3\,(K-1)\,(D-1)\,\left(\frac{\lambda}{\mu}\right)^2$$
$$\Leftrightarrow \quad P_s \;\lessapprox\; \sqrt{\frac{\lambda}{n_s\,\mu}} \quad \text{(region J)}\,. \tag{139}$$

Subsequently, for $P_s \gtrapprox \sqrt{\lambda/(n_s\,\mu)}$, that is, when P_s is in region L, the two-hop paths to data loss, A $\rightarrow$ X $\rightarrow$ UF, become the most probable ones. In fact, as P_s increases, according to (114), (115), and (116), first $P_{\rm B}$, then $P_{\rm D}$ and $P_{\rm C}$ approach one, and therefore the $P_{\rm DL}$ and MTTDL no longer depend on P_s. This occurs when P_s is in region M, with the corresponding probability, $P_{\rm A\rightarrow X\rightarrow UF}$, obtained from (122), (124), and (126), by setting $P_{\rm B} = P_{\rm C} = P_{\rm D} = 1$, that is,

$$P_{\rm A\rightarrow X\rightarrow UF} = (K\,D-1)\,\frac{\lambda}{\mu} \quad \text{(region M)}\,. \tag{140}$$

Combining (138) and (140), we deduce that in region M it holds that

$$3\,(K-1)\,(D-1)\,n_s\,\frac{\lambda}{\mu}\,P_s^2 \;\gtrapprox\; (K\,D-1)\,\frac{\lambda}{\mu}$$
$$\Leftrightarrow \quad P_s \;\gtrapprox\; \sqrt{\frac{KD-1}{3\,(K-1)\,(D-1)\,n_s}} \quad \text{(region M)}\,. \tag{141}$$

Also, in region M, the paths A $\rightarrow$ X $\rightarrow$ UF are more probable than the shortest path A $\rightarrow$ UF. Consequently, from (118) and (140), and using (113), it follows that in region M it holds that

$$(K-1)\,(D-1)\,n_s\,P_s^3 \;\lessapprox\; (K\,D-1)\,\frac{\lambda}{\mu}$$
$$\Leftrightarrow \quad P_s \;\lessapprox\; \left[\frac{(K\,D-1)\,\lambda}{(K-1)\,(D-1)\,n_s\,\mu}\right]^{\frac{1}{3}} \quad \text{(region M)}\,. \tag{142}$$

Subsequently, for $P_s \gtrapprox \{(KD-1)\,\lambda/[(K-1)\,(D-1)\,n_s\,\mu]\}^{1/3}$, that is, when P_s is in region Q, the shortest path to data loss A $\rightarrow$ UF becomes the most probable one. In fact, when $P_{\rm A}$ approaches one, the $P_{\rm DL}$ and MTTDL no longer depend on P_s. Owing to (113), this occurs when P_s is in region R obtained by

$$P_{\rm A} \approx 1 \;\Leftrightarrow\; P_s \;\gtrapprox\; \left[\frac{1}{(K-1)\,(D-1)\,n_s}\right]^{\frac{1}{3}} \quad \text{(region R)}\,. \tag{143}$$

Combining the preceding, (101) yields

$$P_{\rm DL} \approx \begin{cases} \frac{3(K-1)(D-1)\,\lambda^3}{2\,\mu^3}\,, & \text{H}: P_s \ll \frac{\lambda}{2\,n_s\,\mu} \\ \frac{3(K-1)(D-1)\,\lambda^2\,n_s}{\mu^2}\,P_s\,, & \text{I}: \frac{\lambda}{2\,n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{n_s} \\ \frac{3(K-1)(D-1)\,\lambda^2}{\mu^2}\,, & \text{J}: \frac{1}{n_s} \lessapprox P_s \lessapprox \sqrt{\frac{\lambda}{n_s\,\mu}} \\ \frac{3(K-1)(D-1)\,\lambda\,n_s}{\mu}\,P_s^2\,, & \\ & \text{L}: \sqrt{\frac{\lambda}{n_s\,\mu}} \lessapprox P_s \lessapprox \sqrt{\frac{KD-1}{3\,(K-1)\,(D-1)\,n_s}} \\ \frac{(KD-1)\,\lambda}{\mu}\,, & \\ & \text{M}: \sqrt{\frac{KD-1}{3\,(K-1)\,(D-1)\,n_s}} \lessapprox P_s \lessapprox \left[\frac{(KD-1)\,\lambda}{(K-1)(D-1)\,n_s\,\mu}\right]^{\frac{1}{3}} \\ (K-1)(D-1)\,n_s\,P_s^3\,, & \\ & \text{Q}: \left[\frac{(KD-1)\,\lambda}{(K-1)(D-1)\,n_s\,\mu}\right]^{\frac{1}{3}} \lessapprox P_s \lessapprox \left[\frac{1}{(K-1)(D-1)\,n_s}\right]^{\frac{1}{3}} \\ 1\,, \quad \text{R}: \left[\frac{1}{(K-1)(D-1)\,n_s}\right]^{\frac{1}{3}} \lessapprox P_s \le 1\,. & \end{cases} \tag{144}$$

Substituting (144) into (3), and considering $N = KD$, yields the approximate MTTDL of the two-dimensional RAID-5 system, $\text{MTTDL}^{\text{(approx)}}_{\text{2D-RAID-5}}$, given by

$$\text{MTTDL}^{\text{(approx)}}_{\text{2D-RAID-5}} \approx \begin{cases} \frac{2\,\mu^3}{3K(K-1)D(D-1)\,\lambda^4}\,, & \text{H}: P_s \ll \frac{\lambda}{2\,n_s\,\mu} \\ \frac{\mu^2}{3K(K-1)D(D-1)\,\lambda^3\,n_s}\,P_s^{-1}\,, & \\ & \text{I}: \frac{\lambda}{2\,n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{n_s} \\ \frac{\mu^2}{3K(K-1)D(D-1)\,\lambda^3}\,, & \text{J}: \frac{1}{n_s} \lessapprox P_s \lessapprox \sqrt{\frac{\lambda}{n_s\,\mu}} \\ \frac{\mu}{3K(K-1)D(D-1)\,\lambda^2\,n_s}\,P_s^{-2}\,, & \\ & \text{L}: \sqrt{\frac{\lambda}{n_s\,\mu}} \lessapprox P_s \lessapprox \sqrt{\frac{KD-1}{3\,(K-1)\,(D-1)\,n_s}} \\ \frac{\mu}{KD(KD-1)\,\lambda^2}\,, & \\ & \text{M}: \sqrt{\frac{KD-1}{3\,(K-1)\,(D-1)\,n_s}} \lessapprox P_s \lessapprox \left[\frac{(KD-1)\,\lambda}{(K-1)(D-1)\,n_s\,\mu}\right]^{\frac{1}{3}} \\ \frac{1}{K(K-1)D(D-1)\,\lambda\,n_s}\,P_s^{-3}\,, & \\ & \text{Q}: \left[\frac{(KD-1)\,\lambda}{(K-1)(D-1)\,n_s\,\mu}\right]^{\frac{1}{3}} \lessapprox P_s \lessapprox \left[\frac{1}{(K-1)(D-1)\,n_s}\right]^{\frac{1}{3}} \\ \frac{1}{KD\lambda}\,, & \text{R}: \left[\frac{1}{(K-1)(D-1)\,n_s}\right]^{\frac{1}{3}} \lessapprox P_s \le 1\,. \end{cases} \tag{145}$$

Following Remark 6, we obtain the $P_{\rm DL}$ by summing the probabilities of all direct paths to data loss. From (118), (122), (124), (126), and (134), it follows that

$$\begin{aligned} P_{\rm DL} \approx\; & P_{\rm A\rightarrow UF} + P_{\rm A\rightarrow B\rightarrow UF} + P_{\rm A\rightarrow C\rightarrow UF} + P_{\rm A\rightarrow D\rightarrow UF} \\ & + P_{\rm A\rightarrow E\rightarrow DF} + P_{\rm A\rightarrow E\rightarrow UF} \;=\; \min\Bigg(1,\; P_{\rm A} \\ & + [(K-1)P_{\rm B} + (K-1)(D-1)\,P_{\rm C} + (D-1)P_{\rm D}]\,\frac{\lambda}{\mu} \\ & + 3\,(K-1)\,(D-1)\left(\frac{\lambda}{2\,\mu} + P_{\rm E}\right)\left(\frac{\lambda}{\mu}\right)^2\Bigg)\,. \end{aligned} \tag{146}$$

Note that the expression in (146) may exceed one and, as it expresses the probability of data loss, needs to be truncated to one. Assuming that the expression does not exceed one,

substituting (146) into (3), and considering $N = KD$, yields the approximate MTTDL of the two-dimensional RAID-5 array as follows:

$$\begin{aligned}\mathrm{MTTDL}^{(\mathrm{approx})}_{\text{2D-RAID-5}} &\approx \frac{1}{K\,D\,\lambda}\Bigg/\Bigg\{P_{\mathrm{A}} + \\ &+ [(K-1)P_{\mathrm{B}} + (K-1)(D-1)\,P_{\mathrm{C}} + (D-1)P_{\mathrm{D}}]\,\frac{\lambda}{\mu} \\ &+ 3\,(K-1)\,(D-1)\left(\frac{\lambda}{2\,\mu}+P_{\mathrm{E}}\right)\left(\frac{\lambda}{\mu}\right)^2\Bigg\}\ , \end{aligned} \tag{147}$$

where P_{A}, P_{B}, P_{C}, P_{D}, and P_{E} are given by (112), (114), (115), (116), and (117), respectively.

D. A RAID-5 Array Under Latent Errors

The MTTDL of a RAID-5 array under latent sector errors was initially derived in [6] and is included in this article for completeness. The corresponding CTMC model is obtained from [6, Fig. 6] and shown in Fig. 9. When a device fails (state transition from state 0 to state 1), the RAID-5 array enters the critical mode as an additional device failure leads to data loss. The rebuild of the failed device is performed based on the remaining $N-1$ devices. The rebuild fails if any sector of these $N-1$ devices is in error. The probability of this event, $P^{(1)}_{\mathrm{uf}}$, is given by [6, Eq. (1)]

$$P^{(1)}_{\mathrm{uf}} = 1-(1-P_s)^{(N-1)\,n_s} \;\approx\; (N-1)\,n_s\,P_s\ . \tag{148}$$

The system exits from state 1 owing to either another device failure or completion of the rebuild. The former event is represented by the state transition from state 1 to state UF with a rate of $\mu P^{(1)}_{\mathrm{uf}}$, and the latter one by the state transition from state 1 to state 0 with a rate of $\mu(1-P^{(1)}_{\mathrm{uf}})$.

We now proceed to show how the approximate MTTDL of the system can be derived in a straightforward manner by appropriately applying the direct-path-approximation technique. The transition from state 0 to state 1 represents the first device failure. The probabilities of the direct paths to data loss, $P_{1\to\mathrm{DF}}$ and $P_{1\to\mathrm{UF}}$, are given by

$$P_{1\to\mathrm{DF}} = \frac{(N-1)\,\lambda}{\mu+(N-1)\,\lambda} \approx \frac{(N-1)\,\lambda}{\mu}\ , \tag{149}$$

and

$$P_{1\to\mathrm{UF}} = \frac{\mu\,P^{(1)}_{\mathrm{uf}}}{\mu+(N-1)\,\lambda} \approx P^{(1)}_{\mathrm{uf}}\ . \tag{150}$$

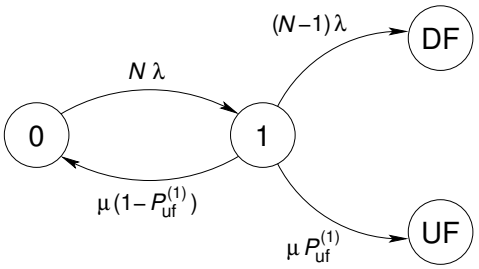


Figure 9. Reliability model for a RAID-5 array under latent errors.

Combining (149) and (150) yields

$$\begin{aligned} P_{\mathrm{DL}} \approx P_{1\to\mathrm{DL}} &= P_{1\to\mathrm{DF}} + P_{1\to\mathrm{UF}} \\ &= \frac{(N-1)\,\lambda+\mu\,P^{(1)}_{\mathrm{uf}}}{\mu+(N-1)\,\lambda} \qquad (151) \\ &\approx \min\left(1,\ (N-1)\,\left(\frac{\lambda}{\mu}\right)+P^{(1)}_{\mathrm{uf}}\right)\ . \qquad (152)\end{aligned}$$

Note that the expression in (152) may exceed one and, as it expresses the probability of data loss, needs to be truncated to one. Assuming that the expression does not exceed one, substituting (152) into (3) yields

$$\mathrm{MTTDL}^{(\mathrm{approx})}_{\text{RAID-5}} \;\approx\; \frac{\mu}{N\,\lambda\,[(N-1)\,\lambda+\mu\,P^{(1)}_{\mathrm{uf}}]}\ , \tag{153}$$

which is the result obtained by Equation (45) of [6] when the first term of the nominator is ignored.

From (149) and (150), and using (148), it follows that the path $1\to\mathrm{DF}$ is the most probable one when P_s is in region A obtained by

$$\frac{\lambda}{\mu} \gg n_s\,P_s \quad\Leftrightarrow\quad P_s \ll \frac{1}{n_s}\cdot\frac{\lambda}{\mu} \quad \text{(region A)}\ . \tag{154}$$

Note that this is the same region as that in (103) for a RAID-6 array.

Subsequently, for $P_s > \lambda/(n_s\,\mu)$, that is, when P_s is in region F, the other path to data loss, $1\to\mathrm{UF}$, becomes the most probable one. In fact, when $P^{(1)}_{\mathrm{uf}}$ approaches one, the P_{DL} and MTTDL no longer depend on P_s. Owing to (148), this occurs when P_s is in region G obtained by

$$P^{(2)}_{\mathrm{uf}} \;\approx\; 1 \quad\Leftrightarrow\quad P_s \gtrapprox \frac{1}{(N-1)\,n_s} \quad \text{(region G)}\ . \tag{155}$$

Combining the preceding, (152) yields

$$\begin{aligned} P_{\mathrm{DL}} &\approx P_{1\to\mathrm{DL}} \\ &\approx \begin{cases} \frac{(N-1)\,\lambda}{\mu}\ , & \mathrm{A}:\ P_s \ll \frac{\lambda}{n_s\,\mu} \\ (N-1)\,n_s\,P_s\ , & \mathrm{F}:\ \frac{\lambda}{n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{(N-1)\,n_s} \\ 1 & \mathrm{G}:\ \frac{1}{(N-1)\,n_s} \lessapprox P_s \le 1\ . \end{cases} \end{aligned} \tag{156}$$

Substituting (156) into (3) yields

$$\begin{aligned} &\mathrm{MTTDL}^{(\mathrm{approx})}_{\text{RAID-5}} \approx \\ &\approx \begin{cases} \frac{\mu}{N(N-1)\,\lambda^2}\ , & \mathrm{A}:\ P_s \ll \frac{\lambda}{n_s\,\mu} \\ \frac{1}{N(N-1)\,\lambda\,n_s}\,P_s^{-1}\ , & \mathrm{F}:\ \frac{\lambda}{n_s\,\mu} \lessapprox P_s \lessapprox \frac{1}{(N-1)\,n_s} \\ \frac{1}{N\,\lambda} & \mathrm{G}:\ \frac{1}{(N-1)\,n_s} \lessapprox P_s \le 1\ . \end{cases} \end{aligned} \tag{157}$$

E. RAID-5 vs. RAID-6 Under Latent Errors

In region A, the ratio of the corresponding reliabilities is given by (58). In regions G and C, the corresponding MTTDLs are obtained from (157) and (108) as follows:

$$\mathrm{MTTDL}^{(\mathrm{approx})}_{\text{RAID-5}} \;\approx\; \frac{1}{N_5\,\lambda} \quad \text{(region G)}\ , \tag{158}$$

and

$$\mathrm{MTTDL}^{(\mathrm{approx})}_{\text{RAID-6}} \;\approx\; \frac{\mu}{N_6\,(N_6-1)\,\lambda^2} \quad \text{(region C)}\ . \tag{159}$$

Moreover, according to (54) and (56), it holds that $\text{MTTDL}_{\text{RAID-5}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-5}}^{(\text{approx})}/2$ and $\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-6}}^{(\text{approx})}$, respectively. Consequently, from (158) and (159), and given that $N_6 = 2\,N_5$, it follows that in region C∩G it holds that

$$\frac{\text{MTTDL}_{\text{RAID-5}}^{(\text{approx, system})}}{\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})}} \;=\; (N_6 - 1) \;\cdot\; \frac{\lambda}{\mu}\,. \qquad (160)$$

Thus, the reliability of the RAID-5 system in region C∩G is less than that of the RAID-6 system by a magnitude dictated by the ratio λ/μ, which is very small. As this holds in both regions A and C∩G, we deduce that this also holds in region B∩F. Consequently, for all realistic values of P_s, the reliability of the RAID-5 system is lower than that of the RAID-6 system by a magnitude dictated by the ratio λ/μ.

F. RAID-6 vs. 2D-RAID-5 Under Latent Errors

In region H∩A, the ratios of the corresponding reliabilities are given by (71) and (76) for cases 1 and 2, respectively.

1) $N_6 = D = K+1$*:* In regions C and J, the corresponding MTTDLs are obtained from (108) and (145) as follows:

$$\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})} \;\approx\; \frac{\mu}{(K+1)\,K\,\lambda^2} \quad \text{(region C)}\,, \qquad (161)$$

and

$$\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})} \;\approx\; \frac{\mu^2}{3\,(K+1)\,K^2\,(K-1)\,\lambda^3} \quad \text{(region J)}\,. \qquad (162)$$

Also, according to (67) and (69), it holds that $\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-6}}^{(\text{approx})}/K$ and $\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})} = \text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})}$, respectively. Consequently, from (161) and (162), it follows that in region J∩C it holds that

$$\frac{\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})}}{\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})}} \;=\; 3\,(K-1) \;\cdot\; \frac{\lambda}{\mu}\,. \qquad (163)$$

2) $N_6 = 2K-1$ *and* $D = (K-1)(2K-1)$*:* In regions C and J, the corresponding MTTDLs are obtained from (108) and (145) as follows:

$$\text{MTTDL}_{\text{RAID-6}}^{(\text{approx})} \;\approx\; \frac{\mu}{2\,(K-1)\,(2K-1)\,\lambda^2} \quad \text{(region C)}\,, \qquad (164)$$

and

$$\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})} \;\approx\; \frac{\mu^2}{3\,K^2\,(K-1)^2\,(2K-1)\,(2K-3)\,\lambda^3} \quad \text{(region J)}\,. \qquad (165)$$

Also, according to (72) and (74), it holds that $\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-6}}^{(\text{approx})}/[K(K-1)]$ and $\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})} = \text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})}$, respectively. Consequently, from (164) and (165), it follows that in region J∩C it holds that

$$\frac{\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})}}{\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})}} \;=\; \frac{3\,K\,(2K-3)}{2} \;\cdot\; \frac{\lambda}{\mu}\,. \qquad (166)$$

Thus, in both cases, the reliability of the RAID-6 system in region J∩C is lower than that of the 2D-RAID-5 system by a magnitude dictated by the ratio λ/μ, which is very small. As this holds in both regions H∩A and J∩C, we deduce that this also holds in region I∩B. Consequently, for all realistic values of P_s, the reliability of the RAID-6 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the ratio λ/μ.

G. RAID-5 vs. 2D-RAID-5 Under Latent Errors

In region H∩A, the ratios of the corresponding reliabilities are given by (84) and (89) for cases 1 and 2, respectively.

1) $N_5 = (K+1)/2$ *and* $D = K+1$*:* In region G, the corresponding MTTDL is obtained from (157) as follows:

$$\text{MTTDL}_{\text{RAID-5}}^{(\text{approx})} \;\approx\; \frac{2}{(K+1)\,\lambda} \quad \text{(region G)}\,, \qquad (167)$$

In region J, the corresponding MTTDL is given by (162). Also, according to (80) and (82), it holds that $\text{MTTDL}_{\text{RAID-5}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-5}}^{(\text{approx})}/(2K)$ and $\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})} = \text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})}$, respectively. Consequently, from (167) and (162), it follows that in region J∩G it holds that

$$\frac{\text{MTTDL}_{\text{RAID-5}}^{(\text{approx, system})}}{\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})}} \;=\; 3\,K\,(K-1) \;\cdot\; \left(\frac{\lambda}{\mu}\right)^2\,. \qquad (168)$$

2) $N_5 = K-1$ *and* $D = (K-1)^2$*:* In regions G and J, the corresponding MTTDLs are obtained from (157) and (145) as follows:

$$\text{MTTDL}_{\text{RAID-5}}^{(\text{approx})} \;\approx\; \frac{1}{(K-1)\,\lambda} \quad \text{(region G)}\,, \qquad (169)$$

and

$$\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})} \;\approx\; \frac{\mu^2}{3\,K^2\,(K-1)^3\,(K-2)\,\lambda^3} \quad \text{(region J)}\,. \qquad (170)$$

Also, according to (85) and (87), it holds that $\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})} = \text{MTTDL}_{\text{RAID-6}}^{(\text{approx})}/[K(K-1)]$ and $\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})} = \text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx})}$, respectively. Consequently, from (169) and (170), it follows that in region J∩G it holds that

$$\frac{\text{MTTDL}_{\text{RAID-6}}^{(\text{approx, system})}}{\text{MTTDL}_{\text{2D-RAID-5}}^{(\text{approx,system})}} \;=\; 3\,K\,(K-1)\,(K-2) \;\cdot\; \left(\frac{\lambda}{\mu}\right)^2\,. \qquad (171)$$

Thus, in both cases, the reliability of the RAID-5 system in region J∩G is lower than that of the 2D-RAID-5 system by a magnitude dictated by the square of the ratio λ/μ, which is very small. As this holds in both regions H∩A and J∩G, we deduce that this also holds in region I∩F. Consequently, for all realistic values of P_s, the reliability of the RAID-5 system is lower than that of the 2D-RAID-5 system by a magnitude dictated by the square of the ratio λ/μ.

VIII. Numerical Results

We consider a system comprised of devices with $C_d = 1$ TB and $S = 512$ bytes. Note that in all cases $P_{\rm DL}$ depends on λ and μ only through their ratio λ/μ. Consequently, the quantity $\lambda\,$MTTDL, which owing to (3) and (53), is given by

$$\lambda\,\text{MTTDL} \approx \frac{1}{n_G\, N\, P_{\rm DL}}\ . \tag{172}$$

also depends on λ and μ only through their ratio λ/μ. In the remainder, we set $\lambda/\mu = 0.001$. The $\lambda\,$MTTDLs of RAID-6, 2D-RAID-5, and RAID-5 systems are evaluated analytically using (110), (147), and (153), respectively.

The combined effects of device and unrecoverable failures in a RAID-5 and a RAID-6 array ($n_G = 1$) of size $N = 8$ can be seen in Fig. 10 as a function of the unrecoverable sector error probability: it shows the most probable paths that lead to data loss along with the resulting $\lambda\,$MTTDL measure. The backward arrows have been included because they affect the probability of occurrence of these paths. The dotted backward arrows indicate transitions that are no longer possible.

The $\lambda\,$MTTDL for the RAID-5 array, indicated by the dashed line, exhibits two plateaus that, according to (157), correspond to the regions A and G. The first plateau, in region A, corresponds to the case where there are no unrecoverable errors and therefore data loss occurs owing to two successive device failures. The second plateau, in region G, corresponds to the first device failure after a mean time of $N\,\lambda$, which in turn leads to data loss during rebuild due to unrecoverable errors.

The $\lambda\,$MTTDL for the RAID-6 array, indicated by the solid line, exhibits three plateaus that, according to (108), correspond to the regions A, C, and G. The first plateau, in region A, corresponds to the case where there are no unrecoverable sector errors, and therefore data loss occurs owing to three successive device failures. In this case, the most probable path is not the shortest path, $1 \to$ UF, but the path $1 \to 2 \to$ DF, indicated by the solid red line in Fig. 11. This line is horizontal because, according to (99), the probability of occurrence of this path does not depend on P_s. In region B, the most probable path is the path $1 \to 2 \to$ UF, indicated by the dashed blue line in Fig. 11. Also, according to (100) and (104), in region C, the probability of occurrence of this path becomes independent of P_s, which results in the second plateau. This corresponds to a second device failure, which in turn leads to data loss during rebuild due to unrecoverable errors. Subsequently, in region D, the most probable path is the shortest path, $1 \to$ UF, indicated by the dotted green line in Fig. 11. Also, according to (95) and (106), in region E, the probability of occurrence of this path becomes one, independent of P_s, which results in the third plateau. This corresponds to the first device failure, which in turn leads to data loss during rebuild due to unrecoverable errors.

Note that the plateaus G and E correspond to the same MTTDL value of $1/(N\,\lambda)$. Similarly, the plateaus A and C correspond to the same MTTDL value of $\mu/[N(N-1)\,\lambda^2]$. From (103), (104), (154), and (155), it follows that region B is about the same as region F. Furthermore, from (108) and (157), it follows that in regions A, B, and C, the MTTDL of the RAID-5 array is lower than that of the RAID-6 array

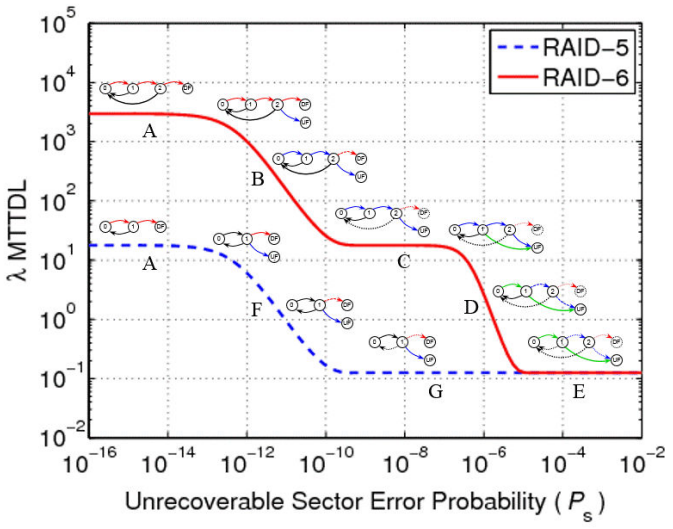


Figure 10. $\lambda\,$MTTDL for a RAID-5 and a RAID-6 array under latent errors ($\lambda/\mu = 0.001$, $N_5 = N_6 = 8$, and $C_d = 1$ TB).

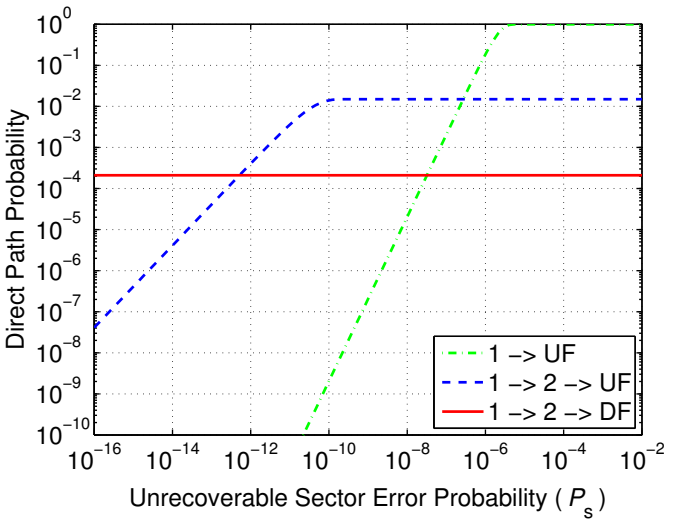


Figure 11. Direct-path probabilities for a RAID-6 array under latent errors ($\lambda/\mu = 0.001$, $N = 8$, and $C_d = 1$ TB).

by a factor of $(N-2)\lambda/\mu$, $(N-2)\lambda/\mu$, and $(N-1)\lambda/\mu$, respectively.

Next, we consider a 2D-RAID-5 array with $K = 9$ and $D = 64$, and therefore a corresponding storage efficiency of 0.875. We also consider a system comprised of $n_G = 72$ RAID-5 arrays of size $N = 8$ and a system comprised of $n_G = 36$ RAID-6 arrays of size $N = 16$, such that these systems store the same amount of user data as the 2D-RAID-5 array under the same storage efficiency. The combined effects of device and unrecoverable failures on the $\lambda\,$MTTDL measure are shown in Fig. 12 as a function of the unrecoverable sector error probability. The various regions and plateaus are also depicted. The probabilities of occurrence of all direct paths to data loss for the 2D-RAID-5 array are shown in Fig. 13. We observe that the shortest path to data loss, A $\to$ UF, indicated by the dotted green line, becomes the most probable one only if

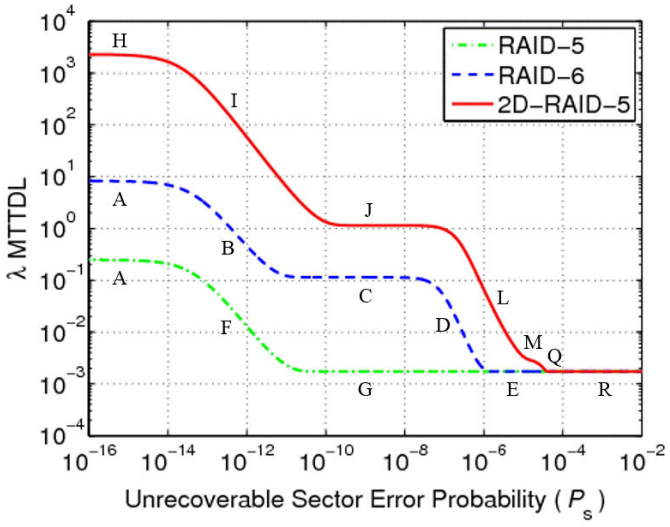


Figure 12. λ MTTDL for a RAID-5, RAID-6, and 2D-RAID-5 system under latent errors ($\lambda/\mu = 0.001$, $N_5 = 8$, $N_6 = 16$, $K = 9$, $D = 64$, and $C_d = 10$ TB).

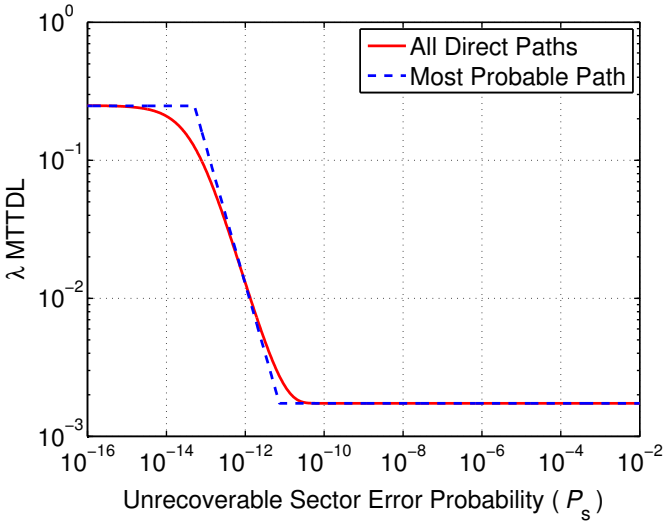


Figure 14. λ MTTDL for a RAID-5 system under latent errors ($\lambda/\mu = 0.001$, $N = 8$, $n_G = 72$, and $C_d = 10$ TB).

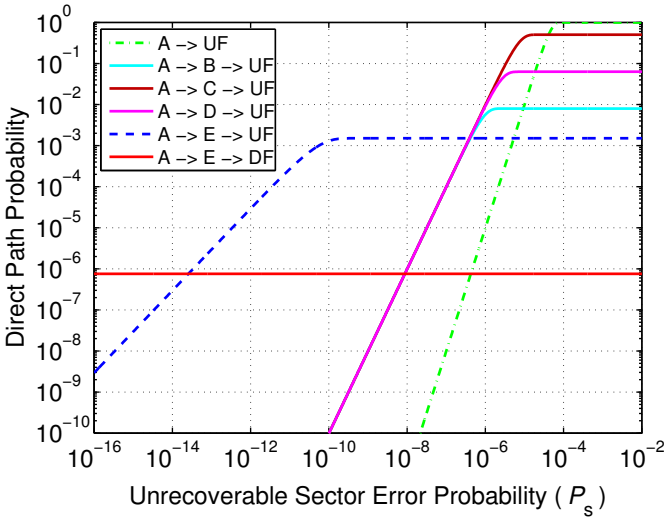


Figure 13. Direct-path probabilities for a 2D-RAID-5 array under latent errors ($\lambda/\mu = 0.001$, $K = 9$, $D = 64$, and $C_d = 10$ TB).

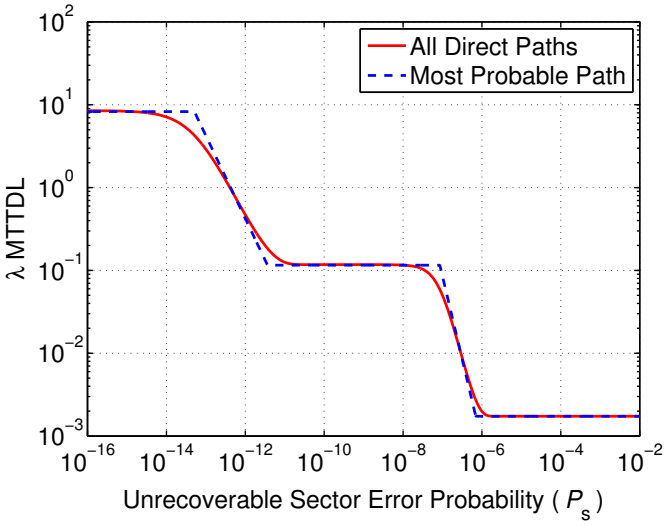


Figure 15. λ MTTDL for a RAID-6 system under latent errors ($\lambda/\mu = 0.001$, $N = 16$, $n_G = 36$, and $C_d = 10$ TB).

$P_s > 10^{-4}$. We also observe that for $P_s < 10^{-7}$, the reliability of the 2D-RAID-5 system is higher than that of the RAID-6 system, which in turn is higher than that of the RAID-5 system.

In Section VII, the MTTDL was derived in two ways: by considering the most probable path to data loss and by considering all direct paths to data loss (see Remark 6). The corresponding results for the RAID-5 system are obtained by (157) and (153) and shown in Fig. 14. The corresponding results for the RAID-6 system are obtained by (108) and (110) and shown in Fig. 15. Finally, the corresponding results for the 2D-RAID-5 array are obtained by (145) and (147) and shown in Fig. 16.

IX. Conclusions

We considered the mean time to data loss (MTTDL) metric, which assesses the reliability level of storage systems. This work presented a simple, yet efficient methodology to approximately assess it analytically for systems with highly reliable devices and a broad set of redundancy schemes. We extended the direct-path approximation to a more general method that considers the most probable paths, which are often the shortest paths, that lead to data loss. We subsequently applied this method to obtain a closed-form expression for the MTTDL of a RAID-51 system. We also considered a specific instance of a RAID-51 system, then derived the corresponding exact MTTDL, and subsequently confirmed that it matches that obtained from the shortest-path-approximation method. Closed-form approximations were also obtained for the MTTDL of

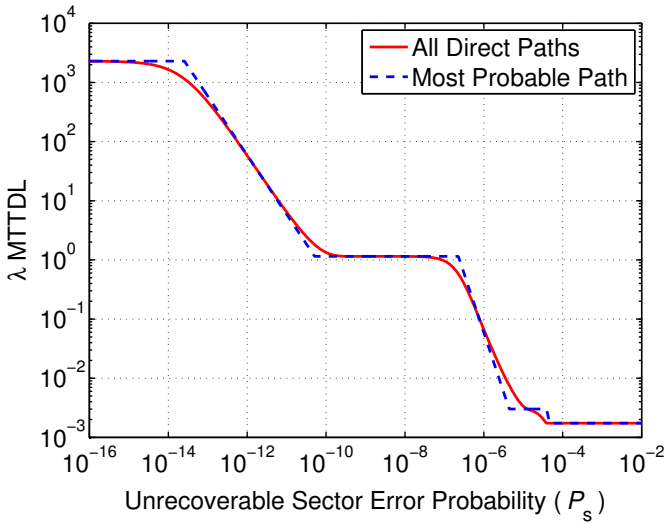


Figure 16. λ MTTDL for a 2D-RAID-5 array under latent errors ($\lambda/\mu = 0.001$, $K = 9$, $D = 64$, and $C_d = 10$ TB).

RAID-6 and two-dimensional RAID-5 systems in the presence of unrecoverable errors and device failures. Subsequently, a thorough comparison of the reliability levels achieved by the redundancy schemes considered was conducted. As the direct-path approximation accurately predicts the reliability of non-Markovian systems with a single shortest path, we conjecture that the shortest-path-approximation method would also accurately predict the reliability of non-Markovian systems with multiple shortest paths.

Application of the shortest-path-approximation methodology developed to derive the MTTDL for systems using other redundancy schemes, such as erasure codes, is a subject of future work.

This methodology can also be applied to system models that additionally consider node, rack, and data-center failures. In such models, there may be short paths to data loss that are not very likely to occur (e.g., disaster events), and direct paths to data loss that are highly probable, but not necessarily short.

REFERENCES

[1] I. Iliadis and V. Venkatesan, "An efficient method for reliability evaluation of data storage systems," in Proceedings of the 8th International Conference on Communication Theory, Reliability, and Quality of Service (CTRQ) (Barcelona, Spain), Apr. 2015, pp. 6–12.

[2] D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," in Proceedings of the ACM SIGMOD International Conference on Management of Data (Chicago, IL), Jun. 1988, pp. 109–116.

[3] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "RAID: High-performance, reliable secondary storage," ACM Comput. Surv., vol. 26, no. 2, Jun. 1994, pp. 145–185.

[4] K. S. Trivedi, Probabilistic and Statistics with Reliability, Queueing and Computer Science Applications, 2nd ed. New York: Wiley, 2002.

[5] V. Venkatesan and I. Iliadis, "A general reliability model for data storage systems," in Proceedings of the 9th International Conference on Quantitative Evaluation of Systems (QEST), Sep. 2012, pp. 209–219.

[6] A. Dholakia, E. Eleftheriou, X.-Y. Hu, I. Iliadis, J. Menon, and K. Rao, "A new intra-disk redundancy scheme for high-reliability RAID storage systems in the presence of unrecoverable errors," ACM Trans. Storage, vol. 4, no. 1, May 2008, pp. 1–42.

[7] A. Thomasian and M. Blaum, "Higher reliability redundant disk arrays: Organization, operation, and coding," ACM Trans. Storage, vol. 5, no. 3, Nov. 2009, pp. 1–59.

[8] I. Iliadis, R. Haas, X.-Y. Hu, and E. Eleftheriou, "Disk scrubbing versus intradisk redundancy for RAID storage systems," ACM Trans. Storage, vol. 7, no. 2, Jul. 2011, pp. 1–42.

[9] I. Iliadis and V. Venkatesan, "Rebuttal to 'Beyond MTTDL: A closed-form RAID-6 reliability equation'," ACM Trans. Storage, vol. 11, no. 2, Mar. 2015, pp. 1–10.

[10] K. Rao, J. L. Hafner, and R. A. Golding, "Reliability for networked storage nodes," IEEE Trans. Dependable Secure Comput., vol. 8, no. 3, May 2011, pp. 404–418.

[11] Q. Xin, E. L. Miller, T. J. E. Schwarz, D. D. E. Long, S. A. Brandt, and W. Litwin, "Reliability mechanisms for very large storage systems," in Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST) (San Diego, CA), Apr. 2003, pp. 146–156.

[12] Q. Xin, T. J. E. Schwarz, and E. L. Miller, "Disk infant mortality in large storage systems," in Proceedings of the 13th Annual IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS) (Atlanta, GA), Sep. 2005, pp. 125–134.

[13] A. Wildani, T. J. E. Schwarz, E. L. Miller, and D. D. E. Long, "Protecting against rare event failures in archival systems," in Proceedings of the 17th Annual IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS) (London, UK), Sep. 2009, pp. 1–11.

[14] M. Bouissou and Y. Lefebvre, "A path-based algorithm to evaluate asymptotic unavailability for large markov models," in Proceedings of the 48th Annual Reliability and Maintainability Symposium, Jan. 2002, pp. 32–39.

[15] I. B. Gertsbakh, "Asymptotic methods in reliability theory: A review," Adv. App. Probability, vol. 16, no. 1, Mar. 1984, pp. 147–175.

[16] V. Venkatesan, I. Iliadis, C. Fragouli, and R. Urbanke, "Reliability of clustered vs. declustered replica placement in data storage systems," in Proceedings of the 19th Annual IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Jul. 2011, pp. 307–317.

[17] V. Venkatesan, I. Iliadis, and R. Haas, "Reliability of data storage systems under network rebuild bandwidth constraints," in Proceedings of the 20th Annual IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Aug. 2012, pp. 189–197.

[18] V. Venkatesan and I. Iliadis, "Effect of codeword placement on the reliability of erasure coded data storage systems," in Proceedings of the 10th International Conference on Quantitative Evaluation of Systems (QEST), Sep. 2013, pp. 241–257.

[19] ——, "Effect of latent errors on the reliability of data storage systems," in Proceedings of the 21th Annual IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Aug. 2013, pp. 293–297.

[20] J.-F. Pâris, T. J. E. Schwarz, A. Amer, and D. D. E. Long, "Highly reliable two-dimensional RAID arrays for archival storage," in Proceedings of the 31st IEEE International Performance Computing and Communications Conference (IPCCC) (Austin, TX), Dec. 2012, pp. 324–331.

[21] I. Iliadis and V. Venkatesan, "Expected annual fraction of data loss as a metric for data storage reliability," in Proceedings of the 22nd Annual IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS) (Paris, France), Sep. 2014, pp. 375–384.

[22] A. Thomasian, "Shortcut method for reliability comparisons in RAID," J. Syst. Software, vol. 79, no. 11, Nov. 2006, pp. 1599–1605.

Accuracy Evaluation of Second-Order Shape Prediction on Tracking Non-Rigid Objects

Kenji Nishida
and Takumi Kobayashi

National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, JAPAN
Email: kenji.nishida@aist.go.jp takumi.kobayashi@aist.go.jp

Jun Fujiki

Department of Applied Mathematics,
Fukuoka University
Fukuoka, JAPAN
Email: fujiki@fukuoka-u.ac.jp

***Abstract*—For our previously proposed shape prediction based tracking algorithm for non-rigid objects, the shape prediction accuracy is critical for the tracking performance. Therefore, we have presented a preliminary evaluation of second-order shape prediction algorithm for tracking non-rigid objects. In the proposed algorithm, the object shape was predicted from the movement of feature points, which were approximated by a second-order Taylor expansion. Approximate first-order movements, the so-called optical flows, were simultaneously exploited by chamfer matching of edgelets. Shape prediction accuracy was evaluated by chamfer matching between the predicted object shape and the actual object shape. While only one video sequence was preliminary evaluated, three more video sequences were evaluated. The new sequences are captured by fixed camera, while our previous sequence was captured by Hand-held camera. In this paper, the effect of second-order shape prediction is quantitatively analyzed by more video sequences. The method exhibits superior shape prediction performance compared to a simple linear prediction method.**

Keywords–Tracking non-rigid objects; Chamfer distance; Shape prediction; Optical flow.

I. INTRODUCTION

Visual object tracking is one of the most popular techniques in the field of computer vision. We have proposed a novel algorithm for tracking non-rigid (deformable) objects based on the second order shape prediction and presented a preliminary evaluation of its performance against the linear (first-order) prediction measured by the similarity between the predicted and actual shapes of the tracked object [1].

Recently, tracking algorithms for non-rigid (deformable) objects have been used in many application fields [2], [3]. In sports scenes, especially those of team sports such as football, there are many objects in similar appearance, which increase the difficulty of tracking. Therefore, we consider both the movement and shape (form) of these objects to be discriminative for tracking.

For the shape of the non-rigid object to change in every video frame, next object shape have to be predicted from preceeding video frames to identify the object. A number of human pose estimation algorithm has been proposed, such as 3D pose estimation of an articulated body using template matching [4] and matching algorithm of pictorial structures [5]. However, they are not predicting the pose in the next video frames.

The movement of the parts must be detected to predict the object shape, and the smallest part must be a feature point. When the object shape is represented by the collection of feature points, the deformation of the object is predicted by exploiting the movement of the feature points. Sim and Sundaraj proposed a motion tracking algorithm using optical flow [6], and this can be considered as the first-order approximation of the movement. For our tracking algorithm, we adopted a shape prediction algorithm based on the second-order approximation of the feature points' movement [7].

In this paper, the evaluation was applied to three more video sequences, which were captured by a fixed camera; Tai chi chuan demonstration, a skier's backshot, and a skier's frontshot. They were compared with the previously evaluated sequence of a skier, which was captured by a hand-held camera. Thereby, the effect of object movement and camera ego-motion were examined from these results.

The remainder of this paper is organized as follows. In Section II, we summarize the previous tracking algorithms. In Section III, we describe our shape prediction algorithm and the tracking procedure that uses the chamfer distance as a similarity measure. The experimental results are presented in Section IV. Finally, we present our conclusions and ideas for future work in Section V.

II. PREVIOUS TRACKING ALGORITHMS

The primary function of an object tracking is to find a moving object in an image. Therefore, detecting differences between consecutive video frames adopted in the first approach, such as a background subtraction algorithm which was employed by Koller [8]. However, the static background might be required, and obviously object detection was difficult when the movement of the objects was small,

A group of feature-based tracking algorithms [9], [10], [11] is proposed as the second approach. Salient features such as corner features are individually extracted and tracked are grouped as belonging to the corresponding object. It can be robust to illumination change. However, the precision of the object location and dimension is affected by the difficulties that arise in feature grouping. The mean-shift algorithm [12], [13] is also included in the feature-based algorithms. In mean-shift algorithm, the local features (such as color histograms) of pixels belonging to the object are followed. The mean-shift approach allows robust and high-speed object tracking, if a local feature that successfully discriminates the object from

the background exists. However, it is difficult to discriminate objects that are close to each other and are similar in color, or to adopt this method for gray-scale images.

Avidan redefined the tracking problem as that of classifying (or discriminating between) the objects and the background [14]. This third approach can be categorized as a detect-and-track approach. In this approach, features are extracted from both the objects and the background; then, a classifier is trained to classify (discriminate) the object from the background. Grabner trained a classifier to discriminate an image patch within an object in the correct position and those with objects in the incorrect position [15], and thereby, the position of the object could be estimated more precisely. While this approach allows stable and robust object tracking, a large number of computations are necessary. The approach of Collins and Mahadevan is regarded as an approach of this type, but they selected discriminative features instead of training classifiers [16], [17]. Grabner introduced on-line boosting to update feature weights to attain compatibility between the adaptation and stability for the appearance change (illumination change, deformation, etc.) of tracking classifiers [18]. Woodley employed discriminative feature selection using a local generative model to cope with appearance change while maintaining the proximity to a static appearance model [19]. The tracking algorithms are also applied to the non-rigid (deforming) objects. Godec proposed *Hough-based tracking* algorithm for non-rigid objects, which employed Hough voting to determine the object's position in the next frame [3].

In detect-and-track approaches, the estimated object position in the next video frame is determined based on the similarity of the features to the object in the current video frame, and a change in appearance, especially deformation, may affect the similarity between the object in the current and the next frame, and thereby, the accuracy of the tracking. Therefore, the tracking accuracy can be improved by predicting the deformation of the object to improve the similarity of the object in the next video frame to that in the current video frame. Sundaramoorthi proposed a new geometric metric for the space of closed curves, and applied it to the tracking of deforming objects [2]. In this algorithm, the deforming shapes of the objects are predicted from the movement of the feature points using first order approximation. However, the first-order approximation is not sufficient to estimate the reciprocating movement, which often human legs and arms do.

III. Shape-based Predict-and-Track Algorithm

In this section, we describe an algorithm for tracking by shape prediction [7]. The algorithm consists of two components, shape prediction and tracking by shape similarity.

A. Notation

The following notation is used throughout this paper.

- X denotes the center of the object,
- $O(X)$ denotes the object image centered at position X,
- $E(X)$ denotes the binary edge image for the object at position X,
- $\hat{O}$ and $\hat{E}$ denote the predicted image and edge image of the object, respectively,
- x denotes the positions of the feature points for object X,
- x' denotes the differential of x, i.e., $x' = \frac{dx}{dt}$,
- x'' denotes $\frac{d^2x}{dt^2}$,
- $\tilde{x}$ denotes the subset of feature points in the object that constitute the outline edge, $\tilde{x} \in E(X)$,
- $\hat{x}$ denotes the predicted position at the next frame for $\tilde{x}$,
- $l(x)$ denotes the edgelet for position x.

B. Shape Prediction

The object shape is represented by the collection of feature points x, and the deformation of the object is predicted by exploiting the movement of the feature points.

Let x_t be the 2-D position of the feature points that constitute the object image O at time t. The position of the points at $t+1$ can be estimated using a Taylor expansion. Up to the second-order, this is

$$x_{t+1} = x_t + x'_t + \frac{1}{2}x''_t, \tag{1}$$

where x' is the so-called optical flow, which is practically computed as the difference in the pixel position:

$$x'_t = x_t - x_{t-1}. \tag{2}$$

Similarly, x'' denotes the second-order differential of x, which is calculated as

$$\begin{aligned} x''_t &= x'_t - x'_{t-1} \\ &= x_t - x_{t-1} - (x_{t-1} - x_{t-2}) \\ &= x_t - 2x_{t-1} + x_{t-2}. \end{aligned} \tag{3}$$

Therefore, the appearance of the object at $t+1$ can be predicted based on the feature point movements computed from three consecutive video frames. Suppose that the shape of the object is determined by the outline edge image Es. The algorithm for detecting the feature point movements is described in Section II-D.

C. Estimation of Object Translation

The movement of the feature points comprises both the object translation (global movement of the center of the object) and the movement relative to the center of the object, which is described by

$$x'_t = X'_t + r'_t, \tag{4}$$

where X denotes the position of the object's center, and r denotes the position of the pixels relative to X. Figure 1 shows the movement of feature point x', the movement of the object's center X', and the relative movement r'.

The relative movement r' is derived from the object deformation, and thus makes a significant contribution to the prediction of the object's shape. Because relative movement obeys the physical constraints of the body parts of the object, its second-order prediction is effective. In contrast, the second-order movement contributes less to the object translation X, because such global movement is considered to be smooth ($X' \approx 0$). Therefore, the purpose of our tracking algorithm is to determine the next object position X_{t+1} based on the similarity between the predicted and actual object shapes, which is computed globally.

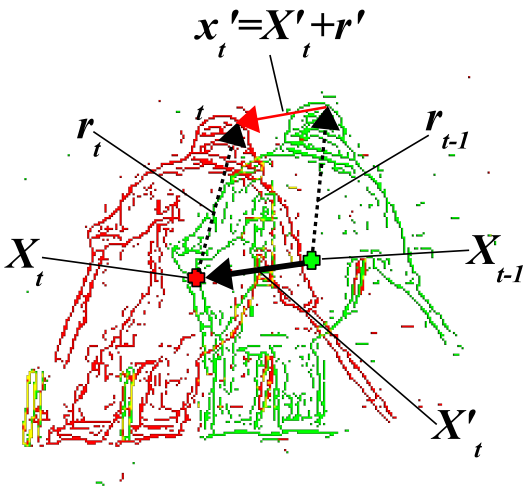


Figure 1. Edge image and object movement.
Green: Edge image for $t-1$, Red: Edge image for t

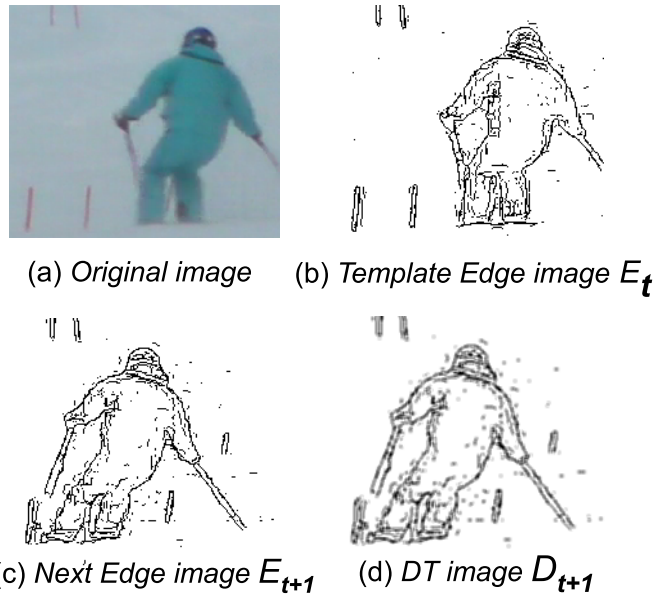


Figure 2. Chamfer system.

The similarity between the predicted edge image $\hat{E}_{t+1}$ and actual edge image E_{t+1} is measured using the chamfer system [20]. This system measures the similarity of two edge images using a distance transform (DT) methodology [21].

Let us consider the problem of measuring the similarity between template edge image E_t (Figure 2(b)) and a successive edge image E_{t+1} (Figure 2(c)). We apply the DT to obtain an image D_{t+1} (Figure 2(d)), in which each pixel value d_{t+1} denotes the distance to the nearest feature pixel in E_{t+1}. The chamfer distance $D_{chamfer}$ is defined as

$$D_{chamfer}(E_t, E_{t+1}) = \frac{1}{|E_t|} \sum_{e \in E_t} d_{t+1}(e), \quad (5)$$

where $|E_t|$ denotes the number of feature points in E_t and e denotes a feature point of E_t.

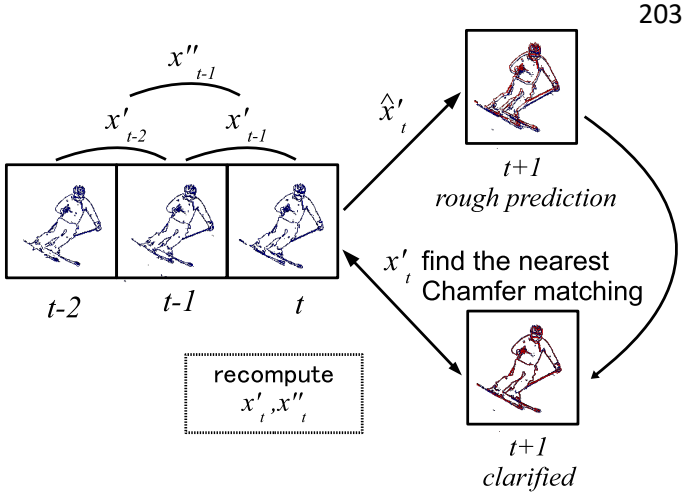


Figure 3. Tracking procedure.

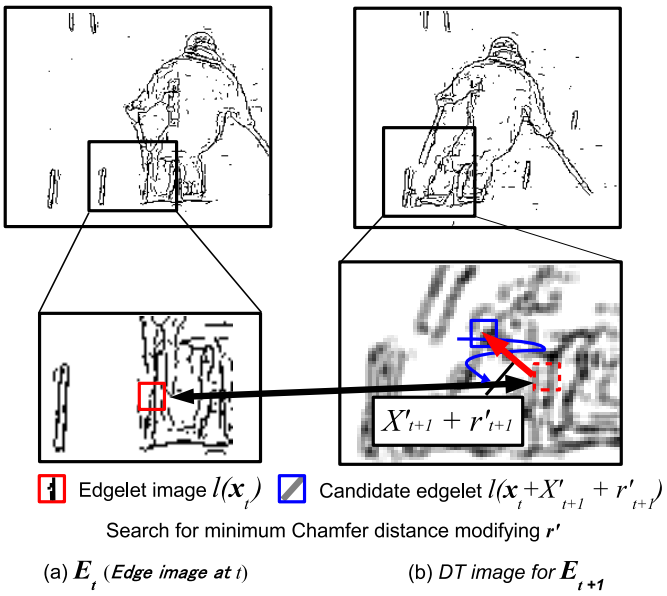


Figure 4. Edgelet tracker.

The translation of the object can be estimated by finding the position of the predicted edge image $\hat{E}_{t+1}$ that minimizes $D_{chamfer}$ between $\hat{E}_{t+1}$ and the actual edge image E_{t+1}:

$$X_{t+1} = \arg\min_{E_{t+1}} D_{chamfer}(\hat{E}_{t+1}, E_{t+1}). \quad (6)$$

Figure 3 illustrates the tracking procedure. First, the optical flow x'_t and its approximate derivative x''_t are computed from preceding video frames at $t-2$, $t-1$, and t. The object shape at $t+1$, denoted by $\hat{E}_{t+1}$, is then predicted using x' and x''. The object position is determined by locating $\hat{E}_{t+1}$ at the position of minimum chamfer distance to the actual shape at $t+1$, E_{t+1}. Finally, the optical flow for the next video frame x'_{t+1} is recomputed using actual edge images E_t and E_{t+1}.

D. Detection of Feature Point Movements

After the object translation X'_{t+1} has been determined, the movement of the feature points x'_{t+1} is detected from the actual object images $O(X_t)$ and $O(X_{t+1})$.

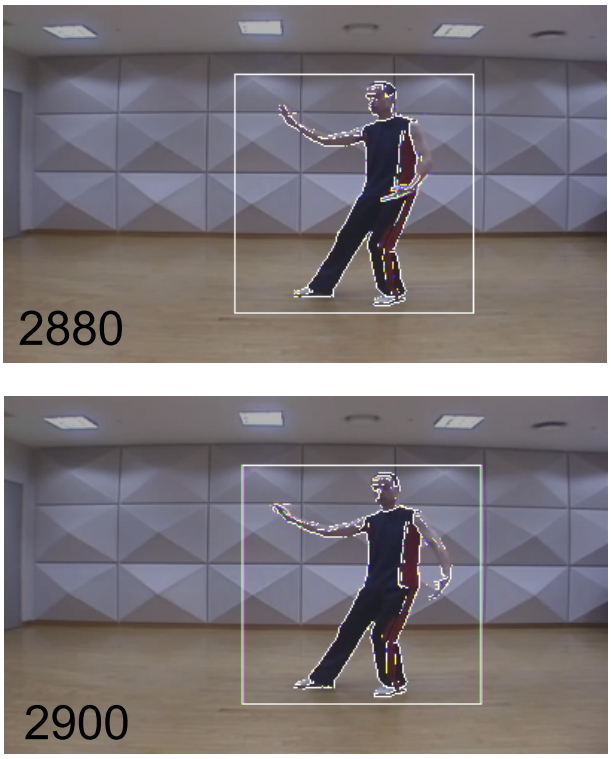


Figure 5. Tracking and shape prediction for Tai chi chuan.
Blue: Ground Truth; Green: Linear Prediction; Red: Second-order Prediction.

The feature point movements x'_{t+1} are directly computed based on the actual edge image at $t+1$ by tracking small parts of the edge (edgelets). We also employed the chamfer system to detect the movement of the edgelets. A template edgelet image $l(\tilde{x}_t)$ extracted from E_t is compared against the candidate edgelet $l(\tilde{x}_t + \hat{x}'_{t+1})$ in the next edge image E_{t+1}. By minimizing the chamfer distance between the two, we obtain the feature point movement (Figure 4):

$$\hat{x}'_{t+1} = \arg\min_{\hat{x}'_{t+1}} D_{chamfer}(l(\tilde{x}_t), l(\tilde{x}_t + \hat{x}'_{t+1})). \quad (7)$$

As the detected movements $\hat{x}'_{t+1}$ may contain noise, we apply a smoothing process by averaging the relative movements in the neighboring region:

$$x'_{t+1} = \frac{1}{N} \sum_{\hat{x}'_{t+1} \in \delta_{t+1}} \hat{x}'_{t+1}, \quad (8)$$

where N denotes the number of detected movements $\hat{x}'_{t+1}$ in the neighborhood δ of $\tilde{x}_t$.

IV. Evaluation of Shape Prediction Performance

The algorithm described above was applied to three video sequences (Tai chi chuan demonstration, a skier backshot, and a skier frontshot) captured by fixed camera, and a sequence of a skier captured by a hand-held camera. The effect of object translation and camera ego-motion on the shape prediction performance was examined.

Figure 6. Shape prediction accuracy for Tai chi chuen.

The proposed second-order shape prediction model was compared with linear one, which is formulated by

$$x_{t+1} = x_t + x'_t. \quad (9)$$

The prediction performance was evaluated by the chamfer distance between the actual image E_{t+1} and the predicted image $\hat{E}_{t+1}$ using (5).

A. Video sequences captured by fixed camera

1) Tai chi chuan demonstration: Figure 7 shows the video frames 3006–3009, when the linear prediction attained better precision. The both prediction algorithm had large error in the shape of right knee in the frame 3006, because of the quick motion by the player. The error in the estimation of feature point movements at the frame 3006 (circled with red) caused the error in the estimation of the acceleration of the feature points at the frame 3007. Thereby, the errors have larger effect on the second-order prediction.

Tai chi chuan is one of the chinese martial arts and the feature is in the slow movement, therefore, the movement and the acceleration of the feature points can easily be detected. Figure 5 shows the tracking result for Tai chi chuan demonstration. The blue pixels represents the predicted object shape (ground truth), the green ones represent the translated predicted shape to determine the object position using (6), and the red ones represents the reconstructed object shape, as calculated by (1). The result image is synthesized from these three shapes, therefore, the white pixels indicate agreement of the both result to the ground truth, the magenta pixels indicate the agreement of the second-order prediction to the ground truth, the cyan pixels indicate the agreement of the linear prediction to the ground truth, and the yellow pixels indicate the agreement of the second-order prediction to the linear prediction but contrary to the ground truth.

Figure 6 shows the chamfer distance to the ground truth, calculated over frames 2950–3050. The result shows that the

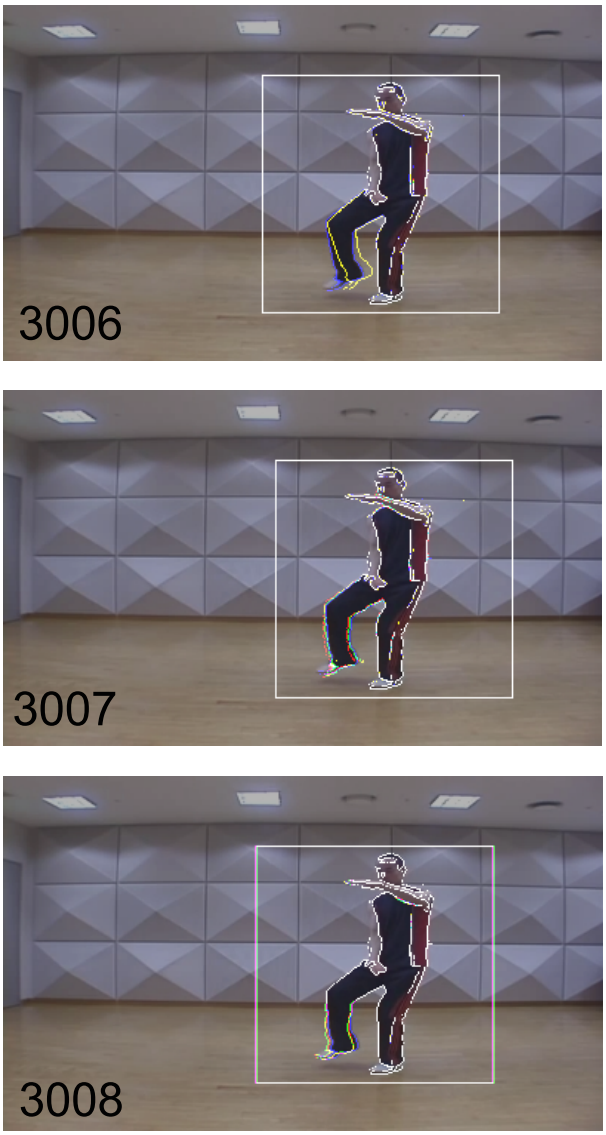


Figure 7. Erroneous video frames for 2nd-order shape prediction of Tai chi chuan.

second-order shape prediction attained better accuracy than the linear one in most of the video frames except 3001 and 3008.

2) Skier 1: backshot: In the video sequence of Tai chi chuan, the object translation is small compared to the object deformation. However, in the sequence of skier captured by a fixed camera, the object translation is much larger than the object deformation. Therefore, the translation and the acceleration of the object might affect the shape prediction accuracy.

Figure 8 shows the tracking result and Figure 9 shows the shape accuracy for the video sequence skier 1. The result in Figure 8, the estimation error (the minimum chamfer distance) tends be high when the object changed its moving direction, such as the video frames around 400, 435, and 475. It also shows that the second-order prediction attained better shape prediction accuracy than linear prediction in most of the video frames, though the second-order prediction produced larger error against the linear method at video frames 478, 479, and

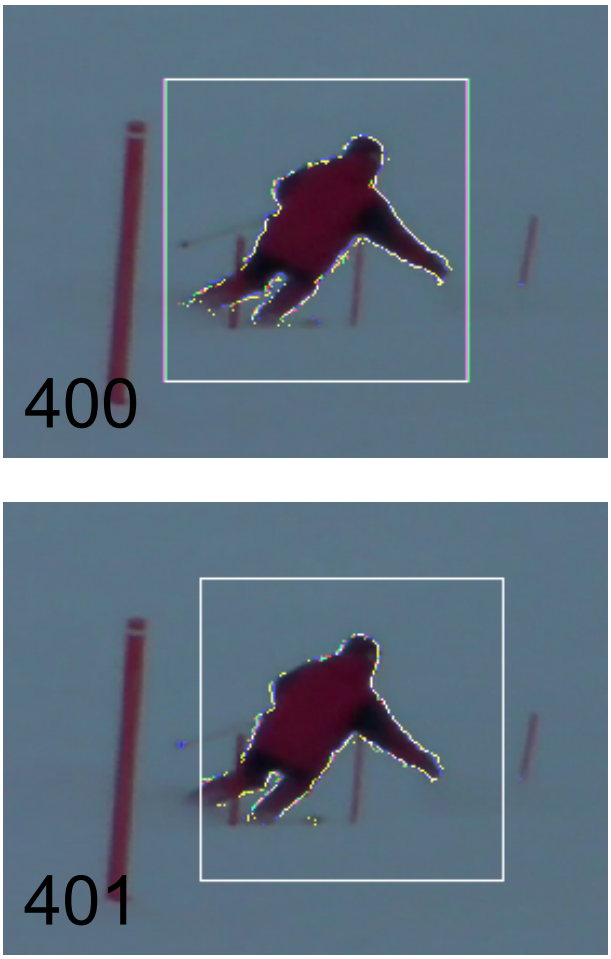


Figure 8. Tracking result for Skier 1 (Fixed camera).
Blue: Ground Truth; Green: Linear Prediction; Red: Second-order Prediction.

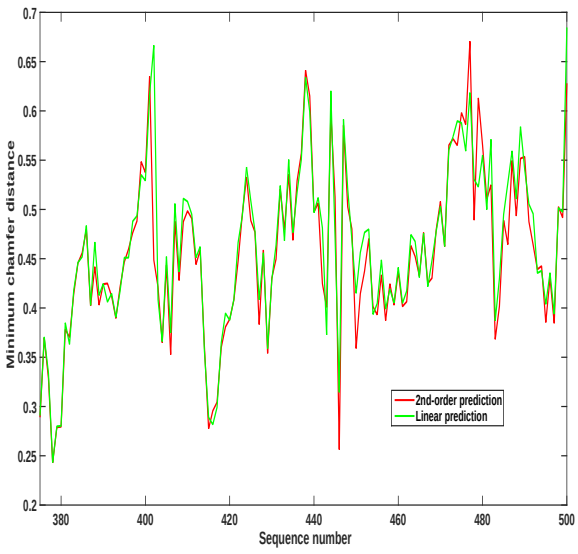
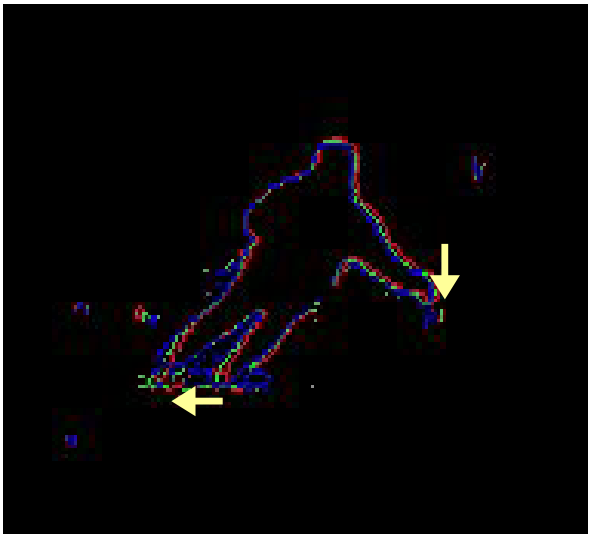


Figure 9. Shape accuracy for Skier 1.

(a) Prediction result for Frame 480



(b) Object movement during frame 478-480

Figure 10. Erroneous video frames for 2nd-order shape prediction of Skier 1 backshot.
(a) Red: 2nd-order prediction; Green: linear prediction; Blue: ground truth.
(b) Red: frame 478; Green: frame 479; Blue: frame 480; Yellow arrow: local movement

480. During video frames 478–480, the object translation was very small and the local movements were also small (Figure 10), therefore the estimation error in local movement (optical flow) must have affected the accuracy of the second-order prediction.

3) Skier 2: frontshot: Figure 11 shows the tracking result and Figure 12 prediction accuracy for skier 2 frontshot. The shape accuracy (Figure 12) shows the same tendency as Figure 8. The estimation error tends be high when the object changed its moving direction, such as the video frames around 240 and 280. In this video sequence, only at the three frames 240, 262 and 288, our second-order method could not outperform the linear method. We considered that the un-eliminated background might affect the prediction accuracy (Figure 13).

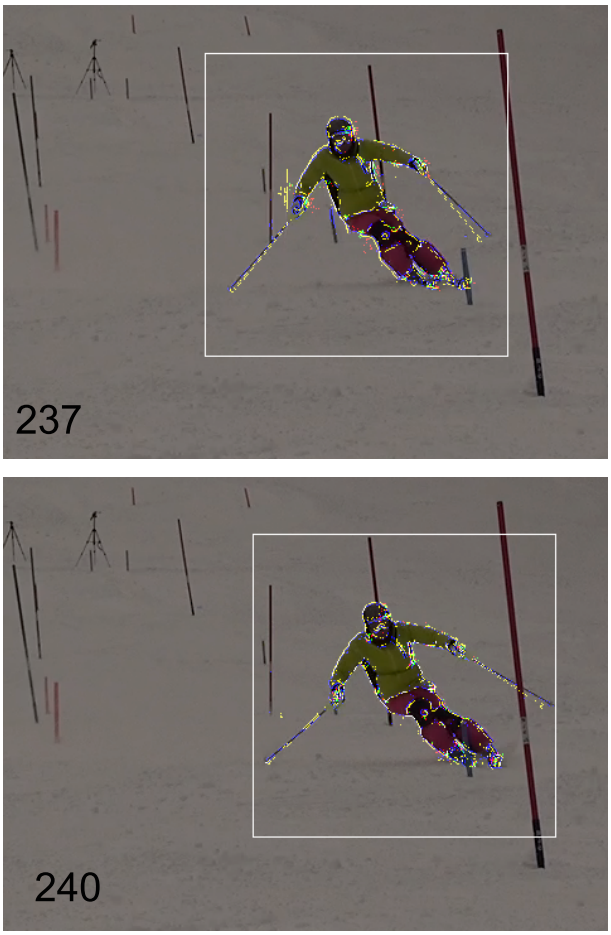


Figure 11. Tracking result for Skier 2 (Fixed camera).
Blue: Ground Truth; Green: Linear Prediction; Red: Second-order Prediction.

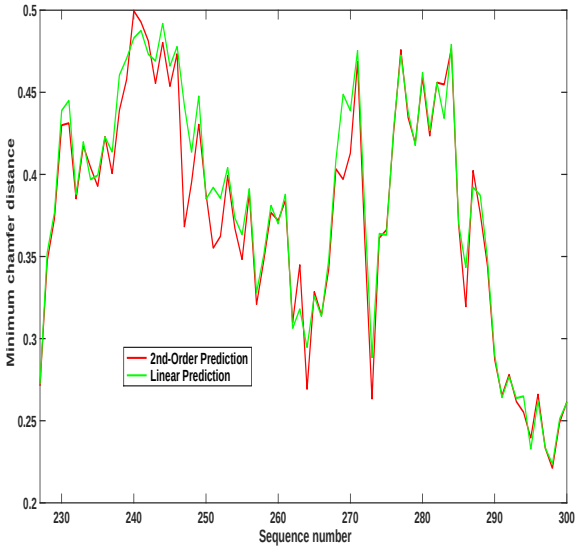


Figure 12. Shape accuracy for Skier 2 by Fixed camera.

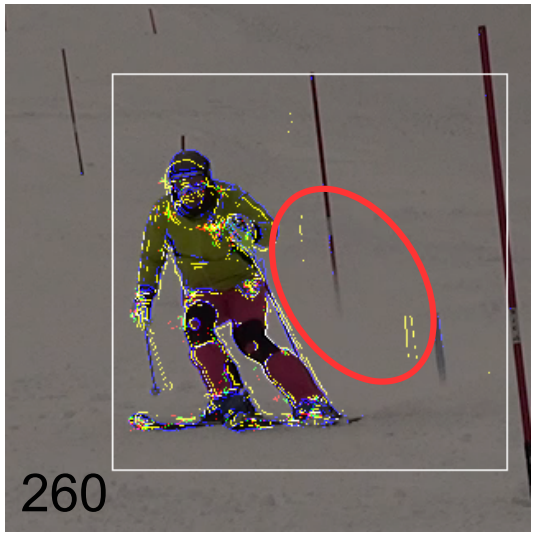
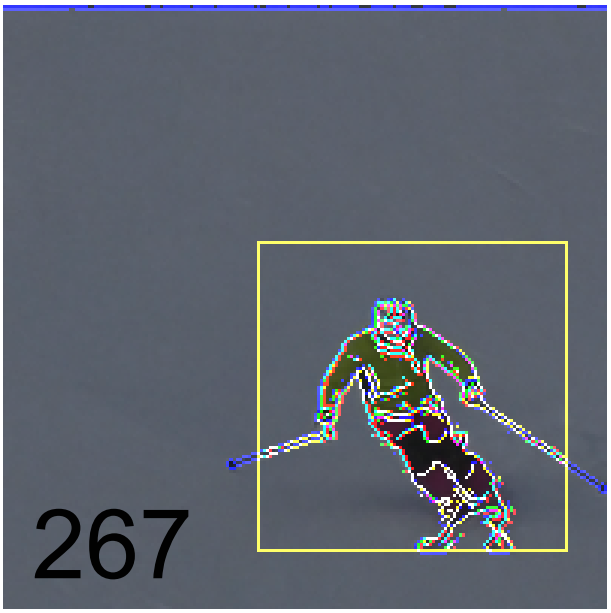


Figure 13. Frame with low shape accuracy for Skier 2.
Un-eliminated background feature points (circles with red) affected the shape and tracking accuracy.



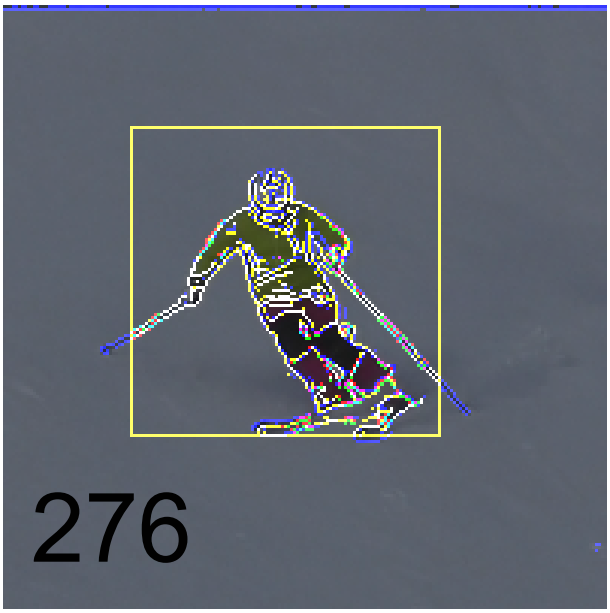


Figure 14. Tracking result for Skier 2 (Hand-held camera.
Blue: Ground Truth; Green: Linear Prediction; Red: Second-order Prediction.

B. Video Sequence captured by Hand-Held Camera (Skier 2)

In the skiing sequence captured by a hand-held camera, the skier was manually "tracked" so as to be shown close to the center of the image frame. Thus, the object tends to exhibit only a small translation in the image frame. However, the object sometimes suffers from a large degree of translation due to manual mis-tracking of the camera. Figure 14 shows the tracking results.

Figure 15 shows the chamfer distance to the ground truth, calculated over frames 230–300. The results show that the second-order prediction attained better accuracy than the linear prediction in 40 out of 70 frames. The second-order prediction is superior during frames 244–249, whereas the linear prediction is preferable from frames 238–240.

Figure 16(a) shows the object translation from frames 244–

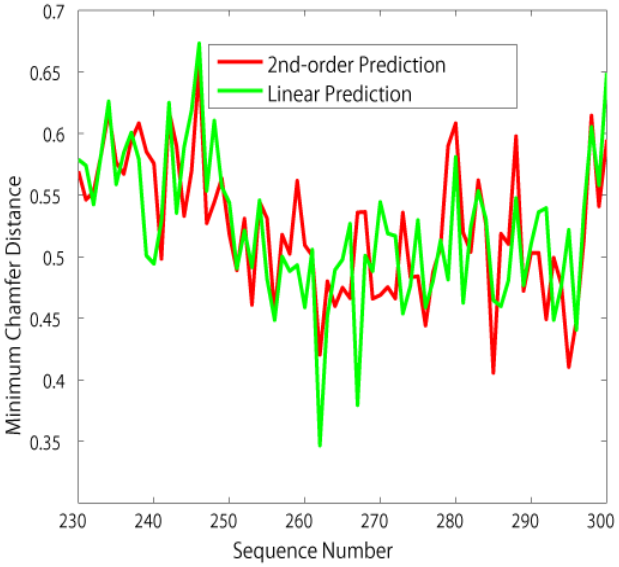


Figure 15. Shape accuracy for Skier 2 by hand-held camera.

248, indicating the direction change at around frame 246. Figure 16(b) shows the object translation from frames 238–240, when the translation direction did not change.

These results indicate that the second-order shape prediction method works well when the direction in which the object must be translated changes.

V. Conclusions

We have evaluated the performance of a second-order shape prediction algorithm. Though the performance is generally higher to that of a linear model, our method outperformed the linear approach in most cases especially when the direction of object movement changed. However, our approach could not outperform the linear approach when the acceleration of the feature points are too high against the frame rate of the video to capture. This evaluation result indicates that the proposed second-order model is robust to objects under acceleration with adequate frame rate.

Acknowledgment

The authors would like to thank Dr. Akaho, group leader of Mathematical Neuroinformatics Group, for his valuable comments and suggestions. This work was supported by JSPS KAKENHI Grant Number 26330217.

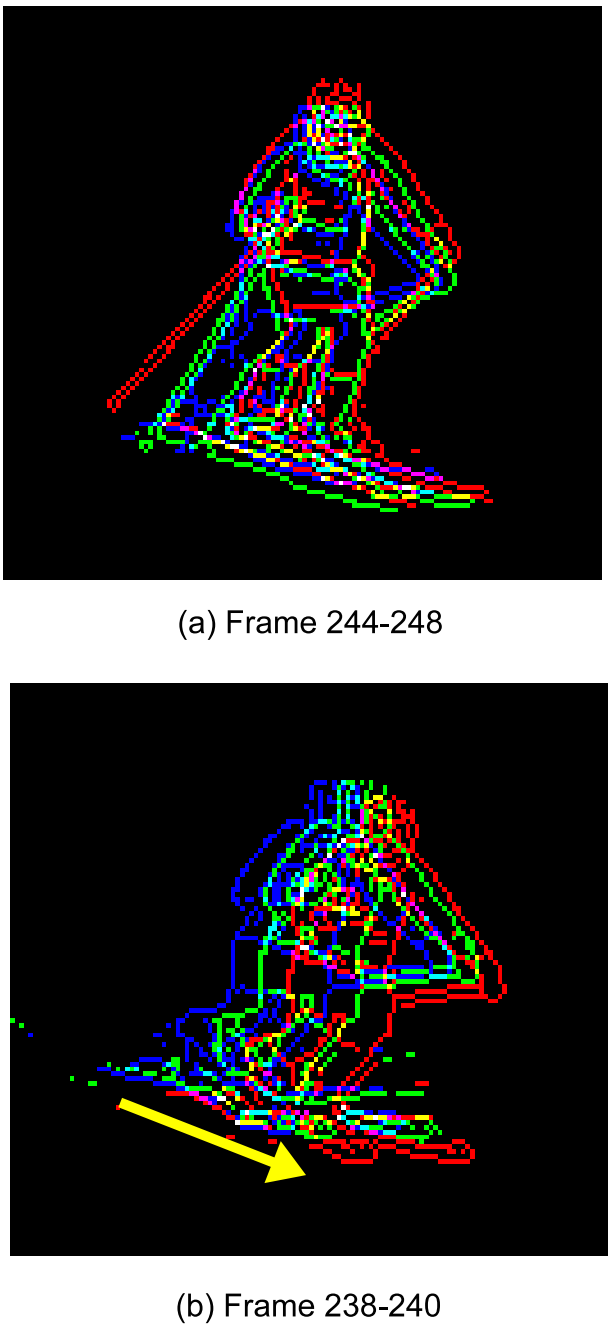


Figure 16. The effect of object translation for prediction accuracy
(a) Blue: frame 244; Green: frame 246; Red: frame 248; Yellow arrow: object translation.
(b) Blue: frame 238; Green: frame 239; Red: frame 240; Yellow arrow: object translation.

REFERENCES

[1] K. Nishida, T. Kobayashi, and J. Fujiki, "The Effect of 2nd-Order Shape Prediction on Tracking Non-Rigid Objects," in *Proc. of the 7th international Conference on Pervasive patterns and Applications*, pp. 60-63, 2015.

[2] G. Sundaramoorthi, A. Mennucci, S. Soatto, and A. Yezzi, "A New Geometric Metric in the Space of Curves, and Applications to Tracking Deforming Objects by Prediction and Filtering," in *SIAM J. of Imaging Science*, Vol. 4, No. 1, pp. 109-145, 2010.

[3] M. Godec, P. M. Roth, and H. Bischof, "Hough-based Tracking on Non-rigid Objects," in *J. of Computer Vision and Image Understanding*, Vol. 117, No. 10, pp. 1245-1256, 2013.

[4] K. Hara, "Real-time Inference of 3D Human Poses by Assembling Local Patches," in *Proc. of IEEE Winter Vision Meeting 2009*, pp. 137-144, 2009.

[5] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient matching of pictorial structures," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2000*, Vol. 2, pp. 66-73, 2000.

[6] K. F. Sim and K. Sundaraj, "Human Motion Tracking of Athlete Using Optical Flow & Artificial Markers," in *Proc. of International Comference on Intelligent and Advanced Systems (ICIAS) 2010*, pp. 1-4, 2010.

[7] K. Nishida, T. Kobayashi, and J. Fujiki, "Tracking by Shape with Deforming Prediction for Non-Rigid Objects," in *Proc. of International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pp. 581-587, 2014.

[8] D. Koller, J. Weber, and J.Malik, "Robust Multiple Car Tracking with Occlusion Reasoning," in *Proc. of European Conference on Computer Vision (ECCV)*, Vol. A, pp. 189-196, 1994.

[9] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A Real-Time Computer Vision System for Measuring Traffic Parameters," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1997*, pp. 495-501, 1997.

[10] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A Real-time Computer Vision System for Vehicle Tracking and Traffic surveillance," in *Transportation Research Part C: Emerging Technologies*, Vol. 6, No. 4, pp. 271-288, 1998.

[11] Z. W. Kim and J. Malik, "Fast Vehicle Detection with Probabilistic Feature Grouping and its Application ot Vehicle Tracking," in *Proc. of 9th International Conference on Computer Vision (ICCV)*, pp. 524-531 2003.

[12] D. Comaniciu and P. Meer, "MeanShift: A Robust Approach Toward Feature Space Analysis," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 5, pp. 603-619, May, 2002.

[13] D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2000*, pp. 142-149, 2000.

[14] S. Avidan, "Ensemble Tracking," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 2, pp. 261-271, 2007.

[15] H. Grabner, M. Grabner, and H. Bischof, "Real-Time Tracking via On-line Boosting," in *Proc. of British Machine Vision Conference (BMVC)*, pp. 47-56, 2006.

[16] R.T. Collins, Y. Liu, and M. Leordeanu, "Online Selection of Discriminative Tracking Features," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 10, pp. 1631-1643, 2005.

[17] V. Mahadevan and N. Vasconcelos, "Salliency-based Discriminant Tracking," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognitio (CVPR) 2009*, pp. 1007-1013, 2009.

[18] H. Grabner, C. Leistner, and H. Bischof, "Semi-Supervised On-Line Boosting for Robust Tracking," in *Proc. European Conference on Computer Vision (ECCV) 2008*, pp. 234-247, 2008.

[19] T. Woodley, B. Stenger, and R. Chipolla, "Tracking using Online Feature Selection and a Local Generative Model," in *Proc. of British Machine Vision Conference (BMVC) 2007*, pp. 86.1-86.10, 2007.

[20] D. M. Gavrila, "Pedestrian Detection from a Moving Vehicle," in Proc. European Conference on Computer Vision (ECCV), 2009, pp. 37-49.

[21] D. Huttenlocher, G. Klanderman, and W. J. Rucklidge, "Comparing Images using the Hausdorff Distance," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 9, pp. 850-863, 1993.

Detection and Resolution of Feature Interactions, the Early Light Way

Carlo Montangero

Dipartimento di Informatica
Università di Pisa, Pisa, Italy
Email: monta@di.unipi.it

Laura Semini

Dipartimento di Informatica
Università di Pisa, Pisa, Italy
Email: semini@di.unipi.it

***Abstract*—The feature interaction problem has been recognized as a general problem of software engineering, whenever one wants to reap the advantages of incremental development. In this context, a feature is a unit of change to be integrated in a new version of the system under development, and the problem is that new features may interact with others in unexpected ways. We introduce a common abstract model, to be built during early requirement analysis in a feature oriented development. The model is common, since all the features share it, and is an abstraction of the behavioural model retaining only what is needed to characterize the features with respect to their possible interactions. The basic constituents are the abstract resources that the features access in their operations, the access mode (read or write), and the reason of each access. Given the model, the interactions between the features are automatically detected, and the goal oriented characterization of the features provides the developers with valuable suggestions on how to qualify them as synergies or conflicts (good and bad interactions), and on how to resolve conflicts. We provide evidence of the feasibility of the approach with an extended example from the Smart Home domain. The main contribution is a lightweight state-based technique to support the developers in the early detection and resolution of the conflicts between features.**

Keywords–Feature interactions; State-based interaction detection; Conflict resolution.

I. INTRODUCTION

This paper extends the approach to the early detection and resolution of feature interactions we introduced in SOFTENG'15 [1]. The feature interaction problem has been recognized as a general problem of software engineering [2] [3] [4] [5], whenever an incremental development approach is taken. In this broader context, the term *feature*, originally used to identify a call processing capability in telecommunications systems, identifies a unit of change to be integrated in a new version of the system under development. The advantages of such an approach lay in the possibility of frequent deliveries and parallel development, in the *agile* spirit. The feature based development is now becoming more and more popular in new important software domains, like automotive and domotics. So, it is worthwhile to take a new look at the main problem with feature based development: a newly added feature may interact with the others in unexpected, most often undesirable, ways. Indeed, the combination of features may result in new behaviours, in general: the behaviours of the combined features may differ from those of the two features in isolation. This is not a negative fact, per se, since a new behaviour may be good, from an opportunistic point of view; however, most often the interaction is disruptive, as some requirements are no longer fulfilled. For instance, consider the following requirements, from the Smart Home domain:

Intruder alarm (IA)	Send an alarm when the main door is unlocked.
Main door opening (MDO)	Allow the occupants to unlock the main door by an interior switch.
Danger prevention (DP)	Unlock the main door when smoke is sensed.

Assuming a feature per requirement, it is easily seen that combining *Intruder alarm* and *Danger prevention* leads to an interaction, since the latter changes the state so that the former raises an alarm. However, an alarm in case of a fire is likely to be seen as a desirable side effect, so that we can live with such an interaction. Also, the combination of the first two features leads to an interaction: an alarm is raised, whenever the occupants decide to open the main door from inside. However, this is likely to be seen as an undesirable behaviour, since the occupants want to leave home quietly.

In general, the process of resolving conflicts in feature driven development has the same cyclic nature: look for interactions in the current specification, identify the conflicts, resolve them updating the specification, cycle until satisfaction.

Many techniques have been proposed to automate (parts of) this process. The search for interactions by manual inspection, as we did above, is obviously unfeasible in practice, due to the number of requirements in current practice. It is also the step with the greatest opportunity for automation. The other steps need human intervention since, at the current state of the art, they cannot be automatized. However, as discussed in Section XII, what is still lacking, in our opinion, is the ability to detect the interactions, identify the conflicts and resolve them by working on a simple model, as it may be available at the beginning of requirements analysis, before any major effort in the development of requirements.

We introduce a technique to support the detection and resolution of feature interactions in the early phases of requirements analysis. The approach is based on a common abstract model of the state of the system, which i) is simple enough to induce a definition of interaction which can be checked by a simple algorithm, and ii) can be modified, together with the feature specification, taking care only of few, essential facets of the system.

The model is *abstract*, since it is an abstraction of the behavioural model retaining only what is needed to characterize each feature with respect to the possible interactions:

the constituents of the model are *resources*, that is, pieces of the state of the system that the features access during their operations. To keep the model, and the analysis, simple, the operations on the resources are abstracted to consider only their access mode, namely *read* or *write*. This way, however, we do not loose in generality since the essential cause of an interaction is a pair of conflicting accesses to a shared resource. In this respect we were inspired by notion of conflict between build tasks introduced by the CBN *software build* model [6].

The work required to build the abstract model can be amortized in two ways. The shared state models can be defined in a reusable and generic manner so that, for a given domain, they can be exploited in many different development efforts, as it happens in Software Product Lines; moreover, the model can be taken as a skeleton to be fleshed out with details as requirements analysis proceeds.

The main concepts and ideas of the approach have been first introduced in [1]. Here, we formalize the proposed detection technique and extend it to deal with *indirect* interactions, i.e., those that depend on the relations among the resources. We also add a feature model to state relations between features such as priorities and mutual exclusions.

The next section summarizes the approach. Subsequent sections describe it in detail: Sections III, IV, and V deal with the definition of the abstract model. Sections VI and VII illustrate the automatic process to derive the interactions from the abstract model. Section VIII discusses synergies and conflicts, and Section IX illustrates some resolution techniques. Section X discusses briefly the complexity of the analysis. Section XI assesses the correctness and completeness of the approach, and Section XII discusses related work. Finally, we draw some conclusions and discuss future work.

In the paper, we use the Smart Home domain described in [7] as a running example. The features are intended to automate the control of a house, managing the home entertainments, providing surveillance and access control, regulating heating, air conditioning, lighting, etc.

II. Summary of the Approach

The lightweight approach to the detection and resolution of feature interaction requires the following activities:

1) Definition of the abstract model. This is obtained by the cooperation of three activities:
   - Domain model building, in terms of the resources accessed by the features.
   - Feature specification. Each feature is described in terms of: its goal; its accesses (r/w) to the resources in the domain model; the goal of each access.
   - Feature model definition, to state mutual exclusion and priority relations among the features.
2) Interaction detection is based on the construction and analysis of an *interaction detection matrix*, which is automatically built in two steps from the abstract model.
   - A basic interaction detection matrix is first derived, where the (direct) accesses mentioned in the features specification are considered.
   - The matrix is filled with indirect resource accesses, which take care of the relations between resources captured in the domain model.
   - Automatic interaction detection: the complete matrix is analyzed to single out possible interactions.
3) Conflict and synergy identification:
   - The interactions identified in the previous step are classified as conflicts or synergies: only conflicts will need to be dealt with in the resolution step.
4) Conflict resolution, that modify the abstract model using various strategies:
   - Restriction
   - Priority
   - Integration
   - Refinement

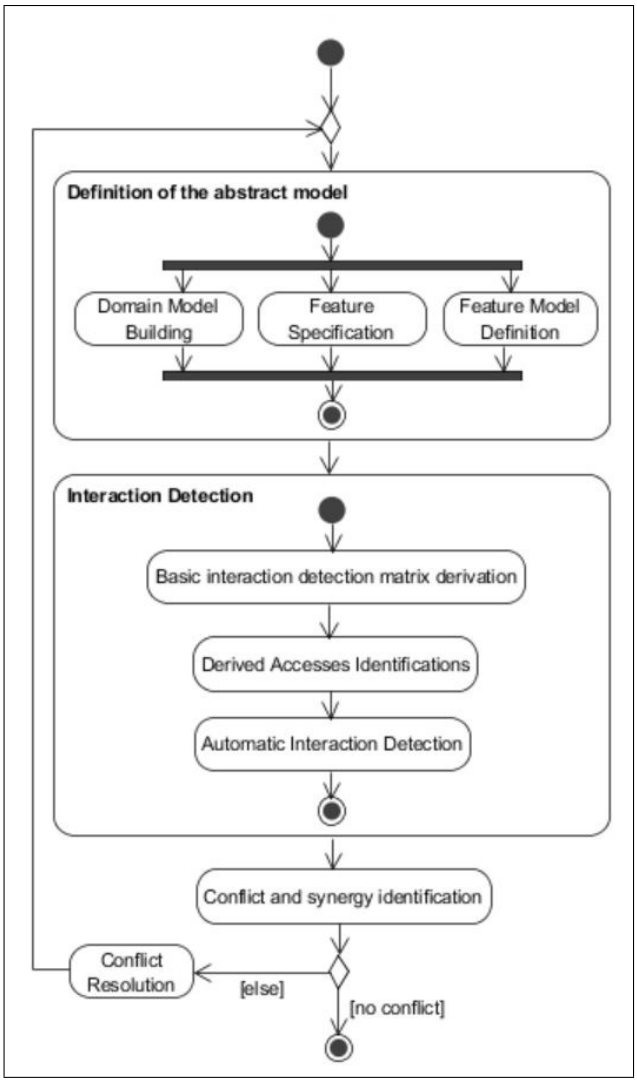


Figure 1. Activities of the lightweight approach.

Figure 1 models the whole process. Note that, from the point of view of the development process, there is no constraint

on how the abstract model is built: in other words, domain model building, feature specification, and feature model definition can be performed in sequence, as well as arm in arm as suggested in Figure 1. All the other activities are each dependent on the outcomes of the previous one in the list. We describe in detail each of them in the next sections.

III. Domain model building

The description of the domain is an integral part of the abstract model. Its purpose is to provide a definition of the accessible resources, i.e., of the shared state that the features access and modify, detailed enough to allow describing the features precisely. There are no special requirements on the notation to express the model. In this paper, we use UML2.0 class diagrams for their wide acceptance.

Given the Smart Home example, so far limited to the features in the Introduction, Figure 2 shows the class diagram modelling the domain. The shared state is made up of the states of the all the resources, which may structured, like Main Door, which owns a Lock.

The structure shown is not final, as new resources can be added by the analyst if he needs them, not only to introduce new features, but also to resolve conflicts, as it happens, for instance, with refinement (Section IX-4).

IV. Feature specification

We model a feature defining: its goal; the resources in the domain model it accesses (r/w); the reason for each access. The feature goal and the resource access reason are used during conflict identification (vs synergies) and resolution.

We introduce a template (Table I), which lists the feature name, its goal, and the involved resources, grouped in two sets (read or written) together with the reason for reading or writing each resource. To make references short, we provide an acronym to each feature. Each access to a resource is identified by its goal.

The three features introduced in the previous section are represented in Table II following the template.

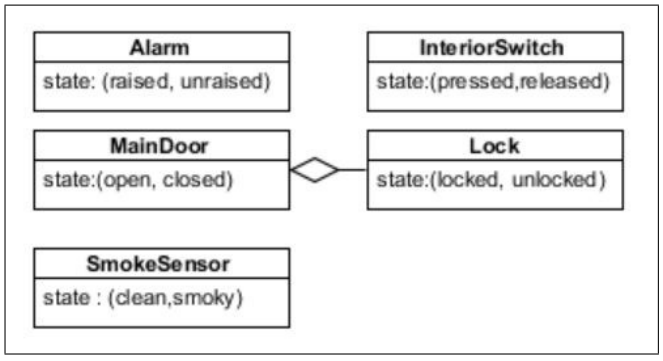


Figure 2. Smart Home Domain.

TABLE I. Feature specification template.

⟨**name**⟩ ⟨**acronym**⟩	**read**	**write**
⟨feature goal⟩	⟨resource⟩ ↪ ⟨access reason⟩	⟨resource⟩ ↪ ⟨access reason⟩

V. Feature Model Definition

A *Feature Model* is a compact representation of the constraints among the features that can be present in a system [8]. In our approach, the feature model records mutual exclusions and priorities between features: in the detection phase this structure is used to disregard the pairs that might interact but will not, since incompatibilities have already been solved by the introduced relations.

Definition (Feature Model) Let $\mathcal{F}$ be the set of features in the abstract model. A Feature Model is a pair

$$\mathcal{FM} = \langle \mathcal{P}, \mathcal{X} \rangle$$

where:

$\mathcal{P} \subseteq \mathcal{F} \times \mathcal{F}$ and $(F, F') \in \mathcal{P}$ when F has priority on F'

$\mathcal{X} \subseteq 2^{\mathcal{F}}$ and $X \in \mathcal{X}$ when the features in X are mutually exclusive.

There is no constraint among IA, DP, and MDO in the initial model, i.e., initially $\mathcal{FM} = \langle \emptyset, \emptyset \rangle$. We will fill it when adding new features, and after the resolution stage.

To focus on the essence of the approach, in the next section we deal with automatic interaction detection on the matrix including only the basic interactions and delay indirect interaction identification to the subsequent one.

VI. Automatic Interaction Detection

Our definition of feature interaction is based on the access mode (read or write) to the resources that make up the shared state of the system. The features access the resources in read mode to assess the state of the system, and in write mode to update it.

Any time two features F and F' access a resource, and at least one of the accesses updates it, there is an interaction: if F updates a resource which is read by F', the new value can change the behaviour of F', hence there is an interaction;

TABLE II. Feature specification: IA, MDO, DP.

Intruder Alarm (IA)	**read**	**write**
To raise an alarm when the main door is unlocked.	main door lock ↪ To know when to raise an alarm	alarm ↪ To raise the alarm
Main door opening (MDO)	**read**	**write**
To manually unlock the door.	InteriorSwitch ↪ To receive the command	Lock ↪ To unlock
Danger prevention (DP)	**read**	**write**
To automatically unlock the door in case of danger	SmokeSensor ↪ To know when there is an alert	Lock ↪ To unlock

TABLE III. INTERACTION DETECTION MATRIX FOR IA, MDO, DP. HERE AND LATER MDOOR STAYS FOR MAINDOOR.

$\mathcal{M}$	Lock	MDoor	Alarm	Interior Switch	Smoke Sensor
IA	r		w		
MDO	w			r	
DP	w				r

if F anf F' both modify the resource, the final value of the resource depends on the feature application order, and there is an interaction too. On the contrary, there is no interaction when F and F' both read a shared resource, since they do not interfere (still, F and F' can interfere if they access, and modify, another resource).

Definition (Interaction)

There is an interaction whenever two features are composed in the same system, and one of them accesses in write mode a resource accessed also by the other, in any mode.

Let us reconsider the features defined in the Introduction and the discussion in the previous section that led to detect some interactions. We can rephrase it in term of resource accesses. For instance, consider the main door lock: accessing it in read mode allows knowing its current state, that is, if the door is locked or unlocked; accessing it in write mode allows locking or unlocking the door. Both IA and MDO access the door lock, in read and write mode, respectively. By definition, we have an interaction. Similarly, also IA and DP interact, since they access the same resource in the same way.

To automate the interaction detection, an *interaction detection matrix* ($\mathcal{M}$) is built, with a row per feature and a column per resource. This is a sparse matrix where each entry is a set that contains information only if the feature in the row accesses the resource in the column, and is empty otherwise:

$$m \in \mathcal{M}_{F,R} \quad iff \quad F \text{ accesses } R \text{ in mode } m$$

As an example, Table III shows the interaction detection matrix for IA, MDO, and DP.

In the interaction detection matrix, it is possible to identify all the pairs of interacting features.

Statement F and F' interact on resource R if and only if

- $w \in \mathcal{M}_{F,R}$ and $\mathcal{M}_{F',R}$ is not empty.
- $(F, F') \notin \mathcal{FM}$, i.e., formally:
  - $(F, F') \notin \mathcal{P}$
  - $(F', F) \notin \mathcal{P}$
  - $\nexists X \in \mathcal{X}$ with $\{F, F'\} \subseteq X$

In other words, any pair of non empty entries in the same column with at least a *w* denotes an interaction of the features in the selected rows, provided the feature model does not prohibit their coexistence in the same system. In the example, we have that all pairs of features, (IA, MDO), (IA, DP), and (MDO, DP) interact on resource Lock, and $\mathcal{FM} = \langle \emptyset, \emptyset \rangle$.

Not all of these interactions are bad ones (conflicts): these have to be identiefied with a subsequent analysis (see Section VIII).

In this view, it is possible that a feature interacts with itself. An example is the following.

Silence at night (SAN) At night, turn off the alarm after three minutes since it started beeping.

The feature specification is in Table IV and matrix is in Table V. The matrix shows an interaction, but it is evident that this is a desired behaviour, that is, a synergy. In other cases, the analysis may discover that the feature is ill-defined and has to be rewritten.

VII. INDIRECT INTERACTIONS IDENTIFICATION

There are other kinds of interactions, that we call *indirect* since they are due to accesses to different, but related, resources. Indeed, the domain model is not made of independent resources: they may be related in such a way that the access to one may entail an access to the other. We consider the following relations that cause *derived* accesses, inducing a state change in a resource as a consequence of a state change in another (related) one:

affects when two resources are associated in such a way that a change in one affects the other;

composition when a resource is a part of another one. This relation, due to its importance in structuring the domain, needs to be considered explicitely but can be reduced to instances of the previous one;

subclass/superclass when a resource belongs to a sub/superclass of another one.

Then, we define how to complete the interaction detection matrix to take into account also these indirect interactions.

A. Affects

Consider the following example dealing with air conditioning (AC):

Natural AC (NAC) If the air temperature in the room is above 27 degrees and the temperature

TABLE IV. FEATURE SPECIFICATION: SAN.

Silence at night (SAN)	**read**	**write**
To turn off the alarm at night	Alarm ↪ To know when it starts beeping	Alarm ↪ To turn it off

TABLE V. INTERACTION DETECTION MATRIX FOR SAN.

$\mathcal{M}$	Lock	MDoor	Alarm	Interior Switch	Smoke Sensor
SAN			r w		

TABLE VI. FEATURE SPECIFICATION: NAC AND ACS.

Natural AC (NAC)	**read**	**write**
To naturally change air.	Room Temp Sensor ↪ To know if room has to be cooled Outside Temp Sens ↪ To know if outside it is cold enought	Window ↪ To open

AC switch-on (ACS))	**read**	**write**
To cool the room with AC.	Room Temp Sensor ↪ To know if room has to be cooled	AirCond ↪ To switch-on

TABLE VII. INTERACTION DETECTION MATRIX FOR IA AND DP2.

$\mathcal{M}$	Lock	MDoor	Alarm	Interior Switch	Smoke Sensor
IA	r		w		
DP2		w			r

outside is below 25, open the windows.

AC switch-on (ACS) If the air temperature in the room is above 27 degrees switch-on the air conditioner.

This is specified in Table VI.

The point here is that in both case there is an effect on the room air. Indeed, a change of state of the windows or the air conditioner affects the air in the room. We want this indirect interaction to be captured as a derived access.

Note that NAC and ACS are applied under the same condition. However, this read coincidence is not relevant for the interaction detection, since no interaction is caused by two read accesses to a resource.

B. Composition

Consider the main door, which is composed of a lock (Figure 3): a write access to the door may result in a write on the lock too, hence we add an *affects* relation between the two resources. For instance, consider a different version of DP, where, rather than simply unlock the door, the home automation system opens it, to facilitate escape and air change:

Danger prevention 2 (DP2) Open the main door when smoke is sensed.

Possibly, DP2 interferes with IA, since to open the door it may be needed to unlock it. However, with the basic matrix of the previous section, this interaction cannot be detected: the features access different resources, and no column in the matrix has more than an element (see Table VII).

In general, changing a resource may change also its parts and cause an interaction with the features accessing one of the parts. On the other side, often, the specifiers forget to mention these derived accesses (e.g., by stating explicitly: *Unlock the main door and* open it when smoke is sensed), and interactions are hardly identified. The domain structure can help in coming up with all interactions automatically.

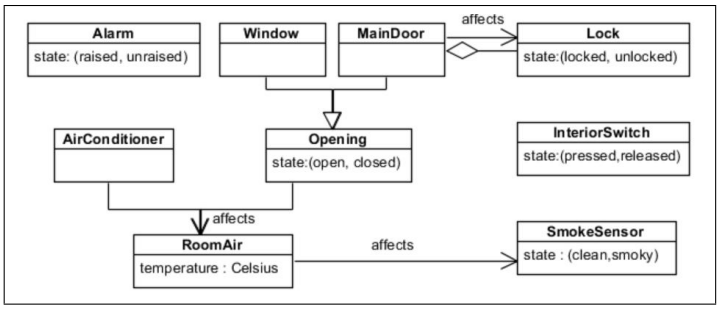


Figure 3. Smart Home Domain, extended.

C. Subclass or Superclass

Let us continue on the same example, with a third version of danger prevention:

Danger prevention 3 (DP3) Open all openings when smoke is sensed.

To cope with this new feature, the domain model needs to include also a new resource, *Opening*, superclass of Door and Window (Figure 3).

Here again, there is an interaction with IA, since a door is an opening (and the door is composed of a lock).

With inheritance we can have a derived access in both directions: a write on a Door may interfere with a feature accessing resource Opening, and hence we derive the write access from Door to Opening. Viceversa, a feature that specifies a change for Opening applies to both Door and Window.

However, there is no derived access between siblings: we must not derive an access to Window from an access to Door or vice-versa.

D. Extending the interaction detection matrix

From now on, to consider the extension just given, we interpret the definition of interaction given in Section VI to include derived accesses. The interaction detection matrix is completed accordingly.

We define a triple of write mode, resource, and relation

$$w \begin{smallmatrix} relation \\ resource \end{smallmatrix}$$

as entry of matrix $\mathcal{M}$, telling that `resource` is indirectly accessed in write mode, through `relation`.

We only derive the write accesses, and not the read accesses, to avoid filling the matrix with redundant information. Indeed, assume a write access on A and a read access on B, with A and B related with one of the aforementioned relations: once we derive a write on B, we can detect the interaction, and there is no need to derive a read on A.

TABLE VIII. EXTENDED INTERACTION DETECTION MATRIX FOR IA AND DP2.

$\mathcal{M}$	Lock	MDoor	Alarm	Interior Switch	Smoke Sensor
IA	r		w		
DP2	w^{comp}_{MDoor}	w			r

Definition ($\mathcal{M}_{F,R}$ extended) Matrix $\mathcal{M}$ is recursively built according to the following rule

$$\mathcal{M}_{F,R} \ni \begin{cases} m & \text{iff } F \text{ accesses } R \text{ in mode } m \\ w^{aff}_{R'} & \text{iff } (w \text{ or } w^{rel}_{res}) \in \mathcal{M}_{F,R'} \text{ and } R' \text{ affects } R \\ w^{sub}_{R'} & \text{iff } (w \text{ or } w^{rel}_{res}) \in \mathcal{M}_{F,R'}, R \text{ is a subclass of } R', \text{ and } rel \neq sup \\ w^{sup}_{R'} & \text{iff } (w \text{ or } w^{rel}_{res}) \in \mathcal{M}_{F,R'}, R \text{ is a superclass of } R', \text{ and } rel \neq sub \end{cases}$$

Specifically, we derive a write access on a resource R if there is a write on R' and the *affects* relation in the domain model tells that a change to R' can lead to a change to R. The derivation is unidirectional, respecting to the direction of the relation.

With inheritance, we derive accesses in both directions. However, derivation paths do not go up and down to avoid deriving an access to a window from an access to the main door (we require $rel \neq sup$). The constraint $rel \neq sub$ applies in the case of multiple inheritance.

Example An example of extended interaction detection matrix is in Table VIII, where w^{aff}_{MDoor} in $\mathcal{M}_{DP2,Lock}$ is added since in the model The Main Door affects the Lock. This triple permits to detect the indirect interaction between IA and DP2.

Example A larger example is Table XI. DP, DP2, and DP3 are alternative versions of danger prevention. They are mutually exclusive, since we want a system to include at most one of them:

$\mathcal{FM} = \langle \emptyset, \{\{\text{DP, DP2, DP3}\}\} \rangle$

This constraint simplifies the analysis since we can discard some pairs of features from the interference analysis. Namely, (DP,DP2), (DP, DP3) and (DP2,DP3).

Note also that the feature model permits to use a unique matrix accommodating various versions of the system instead of using a matrix per each version.

Remark The construction process of $\mathcal{M}$ is finite since a fix-point can always be reached. This is because: the domain model is finite; the matrix elements are sets (and not multisets).

TABLE IX. INTERACTING ACCESS TO LOCK.

–Interaction detected on Lock–			
Feature	Feature Goal	Mode	Access Reason
IA	To raise an alarm when the main door is unlocked.	r	To know if it has been unlocked
MDO	To manually unlock the door.	w	To unlock

TABLE X. INTERACTING DERIVED ACCESS TO LOCK.

–Interaction detected on Lock–			
Feature	Feature Goal	Mode	Access Reason
IA	To raise an alarm when the main door is unlocked.	r	To know if it has been unlocked
DP3	To open all openings in case of smoke.	w^{aff}_{MDoor}	$\Leftarrow (w^{sub}_{Opening}$, MDoor) $\Leftarrow (w$, Opening) $\Leftarrow$ To open.

VIII. CONFLICT AND SYNERGY IDENTIFICATION

For each detected interaction a summarizing table is built, with the information on the goals of the interacting features and on the reasons for the interacting accesses.

As an example, Table IX captures the interaction (IA, MDO) on the main door lock. Such a table will help the expert in the classification of the interaction and its resolution. At this point the expert can state whether the interaction is a synergy or a conflict, as clearly in this case, since we do not want the alarm to be sent when the opening is authorized.

Similar tables are built for the other pairs of interacting features. The expert can recognize that there is a synergy between *Intruder Alarm* and *Danger Prevention*, since sending the alarm is useful when some danger sensor is triggered. Also, the interaction between *Main Door Opening* and *Danger Prevention* is a synergy. Indeed, the two features pursue the same goal, that is to open the door.

In the case of a derived access, the summarizing table reconstructs the chain of the derived accesses, and then gives the reason for the base one, as in Table X.

IX. CONFLICT RESOLUTION

Once an interaction is recognized as a conflict in the analysis phase, we can take some actions to resolve it. In order to discuss possible resolution actions, we need to extend the working example. In addition to IA, MDO, and DP, we also consider a few more features, namely:

Air change (AC)	At 10:00 a.m. open the windows, at 10:30 a.m. close the windows.
Close window with rain (CW)	Close the windows when the rain sensor is triggered.
Video surveillance (VS)	Surveillance cameras are watched remotely via wifi.
Wifi switch-off (WSO)	Switch off the wifi at night.

The extended domain model is in Figure 4, and the specification of the new features is in Table XII.

TABLE XI. COMPLETE INTERACTION DETECTION MATRIX

$\mathcal{M}$	Lock	MDoor	Alarm	Interior Switch	Smoke Sensor	Window	Opening	AirCond	RoomAir	External Temp Sensor
IA	r		w							
MDO	w			r						
DP	w				r					
DP2	w^{aff}_{MDoor}	w			r		w^{sup}_{MDoor}			
DP3	w^{aff}_{MDoor}	$w^{sub}_{Opening}$			r		w			
NAC	w^{aff}_{MDoor}	$w^{sub}_{Opening}$				$w^{sub}_{Opening}$	w		r $w^{aff}_{Opening}$	r
ACS					$w^{aff}_{RoomAir}$			w	r $w^{aff}_{AirCond}$	

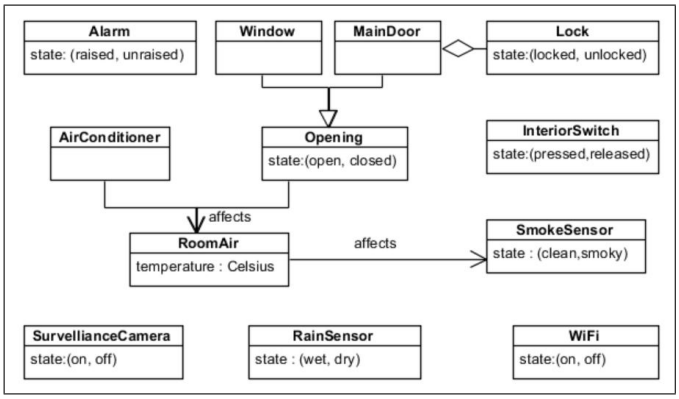


Figure 4. Smart Home Domain, the complete picture.

Various routes to resolution have been proposed in the literature (see [9] [10] [11] for interesting surveys):

1) Restriction: Avoid tout-court that the conflicting features are ever applied in the same system. This is the resolution strategy to be taken when the two features have incompatible goals. In other cases, it is an option the expert can choose. In the running example, we could prevent *Video surveillance (VS)* and *Wifi switch-off (WSO)* from being applied in the same house. We obtain restriction adding a mutual exclusion between the pair of conflicting features in the feature model.

2) Priority between the features: A weaker form of restriction is to guarantee that conflicting features are never applied at the same time. This behaviour can be obtained by defining priorities. Then, in the case two features are both enabled, only the one with higher priority is executed. In our example, priority can be likely used between *Air Change (AC)* and *Close window with rain (CW)*. Both features *write* on the resource *window*. In the case of rain at 10:00 a.m., we want the windows to be closed. The application of this strategy leads to adding a priority pair to the feature model.

3) Integration: According to this resolution strategy, the two interacting features are combined in a new one whose goal encompasses the goals of the two original ones.

VS and WSO can be integrated in a unique feature to switch off the wifi at night, and switch it on if an intruder is sensed, so that surveillance cameras can be watched from a remote machine.

TABLE XII. MORE SMART HOME FEATURES

Air Change (AC)	**read**	**write**
To ventilate the house		Window ↪ To open/close
Close window with rain (CW)	**read**	**write**
To close the windows in case of rain	RainSensor ↪ To know when to close	Window ↪ To close
Video surveillance (VS)	**read**	**write**
To remotely control the house	VideoCamera ↪ To read the recorded data Wifi ↪ To access the camera	
Wifi switch-off (WSO)	**read**	**write**
To switch off the wifi when not used		Wifi ↪ To switch-off

4) Refinement: In any approach based on a shared state, we can apply another resolution strategy, considering if it is possible to add a new resource and make the two conflicting accesses insist on two distinct resources. Since two features conflict only because they access, directly or indirectly the same resource, this refinement solves the problem, by definition. Think again of the conflict between *Intruder Alarm* and *Main door opening*. We might specify a new IA feature excluding the case where the door was unlocked using the interior switch. In some sense, we distinguish between the electrical and mechanical commands to the lock.

It is obvious that, after each resolution step, the features are to be checked again to detect if the changes have solved the conflicts without introducing new ones.

X. Implementation notes

The interaction detection matrix has two properties that are useful for the implementation of the analysis, i.e., to reduce the amount of time and space needed to search the pairs of interacting features: i) the matrix is sparse, and ii) the elaboration of each column (resource) is independent of the others, since it is only necessary to analyze pairs of cells in the same column. According to well known techniques, the matrix can be stored as a list of pairs $\langle resource, listOfAccesses \rangle$, where the second element represents the (sparse) column related to $resource$. Here, $listOfAccesses$ is the list of the non-null matrix entries, each represented as a pair $\langle feature, setOfAccessModes \rangle$.

The average cost of the analysis is then $\mathbf{O}(a^2 \times r)$, where a is the average number of accesses to the resources per feature, and r the number of resources. Note that the structure of the problem is such that it can be profitably attacked by parallel map-reduce, in case of very large matrixes, as it may be the case in real-life projects.

Note that, when creating this structure from the feature specification, there is no need to order the elements in (the lists representing) the columns, due to the independent elaboration of the columns. So, new items can be attached to the front of the list of the accessed resource (linear cost with r), and the matrix can be built in $\mathbf{O}(a \times r)$ in time (and space).

The need to sort the lists of accesses by feature arises only when it is requested to show the whole matrix to the engineers: by memoing the state (ordered or not) of each column, the cost can be made proportional to the number of updates to the matrix.

XI. Discussion

A discussion is needed on the soundness and completeness of our detection method with respect to existing ones. We restrict to design-time techniques, since we are interested in early detection. The most common way to define a feature interaction is based on behaviours [3]:

> A feature interaction occurs when the behavior of one feature is affected by the presence of another feature.

We consider behaviours too, but abstract from their details. Soundness is related with false positives: the rough detection based on the shared resources access model can indeed render false positives, e.g., synergies. These will have to be discarded during the subsequent analysis. However, also the approaches analyzing the concrete behaviour cannot automatically distinguish between conflicts and synergies and some human intervention is still needed to complete the analysis.

On the other side, the completeness problem can be stated as: is it possible that the behaviour of two features interfere even if they do not access, directly or indirectly, any shared resource? This can happen, for instance, if an *hidden resource* is not elicited and is not included in the model.

Often, there are *hidden classes* in a domain description. This is a well known problem in software engineering. In general, when analyzing and modeling a domain, some classes may be intrinsic to the problem, but never explicitly mentioned in the documentation. These classes cannot be found with the noun/verb analysis, and, to be exposed, must be discovered by the analyst.

Let us consider NAC and ACS. The interference between these features is detected since we included the air in the room that has to be cooled in the domain model, and derived an access of both features to this shared resource. If the hidden resource was not elicited, the interference was not found. However, do these feature interfere according to the behaviour based definition? The answer is no, the behaviour of each feature is not affected by the other one. Indeed, the conflict between the actions of opening the windows and switching on air conditioning can be stated only by an expert. So, the situation is similar: with both kind of approaches, the interference can detected thanks to some expert intervention.

Sometimes features interactions are defined in an even more abstract way:

> Features interactions are conflicts between the interests of the involved people.

We express the personal interests in the feature goals, and base the analysis on it. Hence, we are compliant with respect to this notion. Understanding if the persons involved have conflicting interests is a different problem.

Finally, we have restricted our analysis to pairs of features. One further aspect to consider, and this is again based on experience in feature interaction, is the question as to how many features are required to generate a conflict. In the community discussions have taken place around a topic called "three-way interaction". In the feature interaction detection contest at FIW2000 [8] this was an issue, and the community decided that there are two types of three-way interaction: those where there is already an interaction between one or more pairs of the three features and those where the interaction only exists if the triple is present. The latter were termed "true" three-way interactions. Nothing has been written about true three-way interaction, as only one, quite contrived, example of such an interaction has been found. We can hence consider as realistic the assumption that no "true" three-way interaction may occur.

Three-way interactions can occur among features implemented with directives to the preprocessor as done for instance in [12] but this is strictly related with the implementation technique. On the contrary, in our abstract setting, any interaction in a set of three (ore more) features is always caused by the interaction between two of them.

XII. Related Work

In [1], we first described the main concepts and ideas of an early and light analysis of features to detect interactions. Here, the approach is extended along various dimensions:

- We added the feature model definition in the first phase: the feature model describes relations between features such as priorities and mutual exclusions. It helps to accommodate in a unique model various version of a feature based system. At the same time, it is used to discard from the analysis those pairs of features that will never be applied in the same system and hence never interfere.

- We have considered derived accesses to the resources: An interaction can occur between features that are somehow related in the domain. The domain structure can help in coming up with all interactions automatically, instead of needing manual analysis by an expert.
- We have given a formal rule to fill the interaction detection matrix.
- We have discussed the complexity of the implementation.

A. Programming features

Bruns proposed to address the problem at the programming language level, by introducing features as first class objects [2]. Our view is that such an approach is worth pursuing, but needs be complemented by introducing features for features in the early stages of the development process, namely in requirements analysis.

B. Requirements interaction

Taxonomies of feature interaction causes have been presented in the literature [4] [13]. Among the possible causes, there are interactions between feature requirements. We address here a special case of the general problem of requirements interaction. A taxonomy of the field is offered in [14]. It is structured in four levels, and identifies 24 types of interaction collected in 17 categories. It assumes that the requirements specification is structured in system invariants, behavioural requirements, and external resources description. Their analysis is much finer grained than ours. Should the two analysis be performed in sequence, our own should prevent the appearance of some interaction types in the second one, like those of the non-determinism type.

Nakamura et al. proposed a lightweight algorithm to screen out some irrelevant feature combinations before the actual interaction detection, on the ground that the latter may be very expensive [15]. They first build a configuration matrix that represents concisely all possible feature combinations, and is therefore similar in scope to our interaction matrix. However, it is very different in contents, since it is derived from feature requirements specifications in terms of Use Case Maps, which give a very detailed behavioural description of the features. The automatic analysis of the matrix lends to three possible outcomes per pair of features: conflict, no interaction, or interaction prone. In our approach, the automatic analysis gives only two outcomes: no interaction or interaction prone, as one might expect, given the simpler model.

Another similar approach is Identifying Requirements Interactions using Semi-formal methods (IRIS) [7]. Both methods are of general application, and require the construction of a model of the software-to-be. In IRIS the model is given in terms of policies, but the formality is limited to prescribing a tabular/graphical structure to the model. Both methods leave large responsibility to the engineers in the analysis. However, larger effort is required, and larger discretion is left to them in IRIS: in our approach, interaction detection is automatized, and the engineer can focus on conflict identification and resolution. Finally, the IRIS model is much more detailed than ours, so that resolving the identified conflicts may entail much rework, while resolution in our case provides new hints to requirements specification. The last consideration applies as well to the two previous approaches.

C. Design and run-time techniques

As another example of the ubiquity of the feature interaction problem, Weiss et al. show how it appears also in web-services [16]. The approach to design-time conflict detection entails the construction of a goal model where interactions are first identified by inspection, and the subsequent analysis is then conducted on a process algebraic refined formal model. Also in this case, our model is more abstract, and the two techniques may be used synergically.

In a visionary paper, Huang foresees a runtime monitoring module that collects information on running compositions of web-services, and feeds it to an intelligent program that, in turn, detects and resolves conflicts [17].

Several run-time techniques to monitor the actual behaviour of the system and detect conflicts and possibly apply corrective actions, are reported in the literature, as surveyed in [11]: for instance, [18] tackle the problem with SIP based distributed VoIP services; in [19] policies are expressed as safety conditions in Interval Temporal Logic, and they can be checked at run-time by the simulation tool Tempura. These techniques should be seen as complementary to the design-time ones, like ours: the combined use of both approaches can provide the developers with very high confidence in the quality of their product, as suggested also by [10], which discusses the need for both static and dynamic conflict detection and resolution.

D. Aspect oriented techniques

A related topic is that of interactions between aspect-oriented scenarios. A scenario is an actual or expected execution trace of a system under development. The work described in [20] is similar to ours, in so far as they place it in the phase of requirements analysis, propose a lightweight semantic interpretation of model elements. The technique relies on a set of annotations for each aspect domain, together with a model of how annotations from different domains influence each other. The latter allows the automatic analysis of inter-domain interactions. It is likely that, if feature and aspect orientation are combined in the same development, the two techniques could be integrated.

E. Formal methods

A recent trend of design-time conflict detection exploits formal static analysis by theorem proving and model checking. The need for experimentation along this line has been recognized by Layouni et al. in [21], where they exploit the model checker Alloy [22] for automated conflict detection.

In [23], we presented a formal semantics for the APPEL policy language, which so far benefited only from an informal semantics. We also presented a novel method to reason about conflicts in APPEL policies based on the developed semantics and modal logic, and have touched on conflict resolution.

In [24], we show how to express APPEL [25] policies in UML state machines, and exploit the UMC [26] model checker to detect conflicts. In [27], we automate the translation from APPEL to the UMC input language, and address the discovery and handling of conflicts arising from deployment-within the same parallel application-of independently developed policies.

A feature interaction detection method close to model checking is presented in [28]: a model of the features is built using finite state automata, and the properties to be satisfied

are expressed in the temporal logic Lustre. The environment of the feature is described in terms of the (logical) properties it guarantees, and a simulation of its behaviour is randomly generated by the Lutess tool; the advantage is that such an approach helps avoiding state explosion.

F. Abstract Interpretation

We remark a difference with the usual way of performing abstract interpretation [29], where the starting point is a detailed model, which is simplified, by abstracting away the information that is not needed for the intended analysis. An analysis by abstract interpretation defines a finite abstract set of values for the variables in a program, and an abstract version of the program defined on this abstract domain. Here, we abstract the actions to read or write but we only consider the variables (resources) name, and not their values, concrete or abstract.

G. Interactions affecting performance

Recently, work has been done on detecting and resolving interactions that, thought not disrupting the behaviour, impact on the overall performance of the system. The approach described in [30] is based on a simple black box model: interactions are detected using direct performance measurements designed according to few heuristics. It would be interesting to assess whether our technique may supplement advantageously the heuristics to the point of balancing the cost of the required domain model.

H. Best practices in requirements engineering (RE)

We refer to [31], since it emphasizes the identification of the product goals in the early phases of the analysis, and addresses explicitly the problem of conflicts in requirements. More precisely, the advocated RE process foresees the construction of a set of models of the system-to-be, each addressing a dimension of concerns. The most relevant one for our purposes is the Goal Model, which captures the system objectives in a structure of system goals and refines them to software requirements (SR). In this context,

- a goal is a prescriptive statement of intent that the system should satisfy through the cooperation of its agents,
- some of the agents are software ones, that is, they are part of the software-to-be, and
- a (software) requirement is a goal under the responsibility of a single (software) agent.

To see how our approach may fit into this scenario, it is enough to consider each feature as a software agent, whose requirement is given by the associated goal. Moreover, one can easily see that the software requirements in the running example may be the result of refining more general goals, like avoid that people are trapped into the house in case of fire.

The standard validation of the RE Goal Model includes a process to manage goal conflicts, which consists of four steps:

1) identify overlapping statements, i.e., those that refer to some inter-related phenomena;
2) detect conflicts among overlapping statements, possibly using some tool supported heuristics;
3) generate conflict solutions;
4) evaluate solutions and select the best ones.

In our approach, the first step is the construction of the interaction detection matrix, the second one is the process of pairwise feature interaction analysis described in Section VIII, and the third one what suggested in Section IX.

According to the agile nature of feature oriented software development, our approach to conflict detection entails also the construction of modelling items that belong to down stream activities in the RE process of [31], namely, building an Object and an Operation model.

The Object model provides a structural view of the *system-to-be*, showing how the concepts involved in the relevant phenomena are structured in terms of individual attributes and relationships with other concepts. As such, it is often conveniently represented by UML Class diagrams. Among the types of object to be considered in this model we find the already mentioned Agents, and the Entities, i.e., passive objects. The collection of the entities defines the state-space of the system, in terms of the object instances that may be present, each with its own internal state, as defined by the values of its attributes. Being at the RE level, there are no issues of information hiding, that is, a shared state is assumed.

The Operation model provides an operational rather than a declarative view of the system-to-be, unlike the previous models. This one is essential in Goal *operationalization*, that is, the process of mapping the requirements (leaf goals) to a set of operations ensuring them. Here, an operation is characterized by necessary and sufficient conditions for its application, which yields a state transition, in turn characterized by the operation post-condition.

In our approach, the collection of the resources and of the features constitutes the Object model. Any resource is an entity and any feature is an Agent. However, we depart from van Lamsweerde's process with respect to the Operation model, since we do not share his goal that the model contains enough information to allow its validation, that is, providing evidence that the operations of each agent ensure the goals. As we have shown, abstracting operations to their mode (read/write) and goal is sufficient to support feature conflict detection and resolution.

The integration of the support to interaction detection described here with the standard tools that support Requirements Engineering (RE), like DOORS, can be foreseen to occur in two modes, namely, loosely or tightly. In either cases, the information collected in the RE tool can be exploited to provide the engineer in chase of interactions with the structure of the tables of the features, i.e., names and definitions. In the case of loose coupling, these information need be exported in a dedicated tool: The engineer can then complete the tables adding the affected resources and the access modes, which are unlikely to be available in a standard RE tool. Once the analysis and resolution have been performed, the relevant information have to be fed back into the RE tool. A dedicated tool, equipped with interfaces supporting the most popular standard RE data interchange XML based standard, would support the interaction detection technique presented here for a wide range of RE tools. To get a tight coupling, one has to rest on the extension features the RE tool at hand offers: given that the computations needed to put our technique to work are essentially simple, there should be no major problem with most RE tools.

XIII. Conclusions

We present a state based approach to the early detection, analysis and resolution of interactions in feature oriented software development. Starting with a light model of the state that the features abstractly share, the main steps of our approach are the generation of an interaction matrix, the assessment of each interaction (conflict or synergy), and the update of the model to resolve conflicts. The abstraction is such that only the mode (read or write) of an access to the shared state is considered; each access is characterized by its contribution to the overall goal of the feature it pertains to.

We provide a proof of concept of how interactions can be detected automatically, as well as of how the developers can get support in their assessment of the interactions and resolution of the conflicts, looking at the well known Smart Home domain.

An interesting development will be to evaluate whether to formalize the goal model, and how, in view of a (partial) automatic support to the developers' analysis tasks. Another line of development of the approach would be to supplement each resource in the shared space with a standard access protocol, to prevent conflicting interactions. Inspiration in this direction may come from well established practices, like access control schemes and concurrency control.

Acknowledgments

The work was partly supported by the Italian MIUR PRIN project "Security Horizons".

References

[1] C. Montangero and L. Semini, "A lightweight approach to the early detection and resolution of feature interactions," in International Conference on Advances and Trends in Software Engineering (SOFTENG 2015). Barcelona, Spain: ThinkMind digital library, 2015, pp. 72–77.

[2] G. Bruns, "Foundations for Features," in Feature Interactions in Telecommunications and Software Systems VIII, S. Reiff-Marganiec and M. Ryan, Eds. IOS Press (Amsterdam), June 2005, pp. 3–11.

[3] S. Apel, J. M. Atlee, L. Baresi, and P. Zave, "Feature interactions: The next generation (dagstuhl seminar 14281)," vol. 4, no. 7. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014, pp. 1–24, URL: drops.dagstuhl.de/opus/volltexte/2014/4783/ [retrieved: Nov, 2015].

[4] A. Nhlabatsi, R. Laney, and B. Nuseibeh, "Feature interaction: the security threat from within software systems," Progress in Informatics, no. 5, 2008, pp. 75–89.

[5] Various Editors, "Feature Interactions in Software and Communication Systems," international conference series.

[6] D. Coetzee, A. Bhaskar, and G. Necula, "A model and framework for reliable build systems," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2012-27 arxiv.org/pdf/1203.2704.pdf, Feb 2012, URL: www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-27.html [retrieved: Nov, 2015].

[7] M. Shehata, A. Eberlein, and A. Fapojuwo, "Using semi-formal methods for detecting interactions among smart homes policies," Science of Computer Programming, vol. 67, no. 2-3, 2007, pp. 125–161.

[8] P. Asirelli, M. ter Beek, A. Fantechi, and S. Gnesi, "A compositional framework to derive product line behavioural descriptions," in 5th Int. Symp. on Leveraging Applications of Formal Methods, Verification and Validation, ser. LNCS, vol. 7609. Heraklion, Crete: Springer, 2012, pp. 146–161.

[9] D. O. Keck and P. J. Kuehn, "The feature and service interaction problem in telecommunications systems: A survey," IEEE Transactions on Software Engineering, vol. 24, no. 10, Oct. 1998, pp. 779–796.

[10] N. Dunlop, J. Indulska, and K. Raymond, "Methods for conflict resolution in policy-based management systems," in Enterprise Distributed Object Computing Conf. IEEE Computer Society, 2002, pp. 15–26.

[11] M. Calder, M. Kolberg, E. H. Magill, and S. Reiff-Marganiec, "Feature interaction: A critical review and considered forecast," Computer Networks, vol. 41, 2001, pp. 115–141.

[12] S. Apel, D. Batory, C. Kästner, and G. Saake, Feature-Oriented Software Product Lines: Concepts and Implementation. Springer, 2013.

[13] S. Reiff-Marganiec and K. J. Turner, "Feature interaction in policies," Comput. Networks, vol. 45, no. 5, 2004, pp. 569–584.

[14] M. Shehata, A. Eberlein, and A. Fapojuwo, "A taxonomy for identifying requirement interactions in software systems," Computer Networks, vol. 51, no. 2, 2007, pp. 398–425.

[15] M. Nakamura, T. Kikuno, J. Hassine, and L. Logrippo, "Feature interaction filtering with use case maps at requirements stage," in [32], May 2000, pp. 163–178.

[16] B. E. M. Weiss, A. Oreshkin, "Method for detecting functional feature interactions of web services," Journal of Computer Systems Science and Engineering, vol. 21, no. 4, 2006, pp. 273–284.

[17] Q. Zhao, J. Huang, X. Chen, and G. Huang, "Feature interaction problems in web-based service composition," in Feature Interactions in Software and Communication System X, S. Reiff-Marganiec and M. Nakamura, Eds. IOS Press, 2009, pp. 234–241.

[18] M. Kolberg and E. Magill, "Managing feature interactions between distributed sip call control services," Computer Network, vol. 51, no. 2, Feb. 2007, pp. 536–557.

[19] F. Siewe, A. Cau, and H. Zedan, "A compositional framework for access control policies enforcement," in Proceedings of the 2003 ACM workshop on Formal Methods in Security Engineering. NY, NY, USA: ACM Press, 2003, pp. 32–42.

[20] G. Mussbacher, J. Whittle, and D. Amyot, "Modeling and detecting semantic-based interactions in aspect-oriented scenarios," Requirements Engineering, vol. 15, 2010, pp. 197–214.

[21] A. Layouni, L. Logrippo, and K. Turner, "Conflict detection in call control using first-order logic model checking," in Proc. 9th Int. Conf. on Feature Interactions in Software and Communications Systems, L. du Bousquet and J.-L. Richier, Eds. France: IMAG Laboratory, University of Grenoble, 2007, pp. 77–92.

[22] Alloy Community, URL: alloy.mit.edu [retrieved: Nov, 2015].

[23] C. Montangero, S. Reiff-Marganiec, and L. Semini, "Logic-based conflict detection for distributed policies," Fundamenta Informaticae, vol. 89, no. 4, 2008, pp. 511–538.

[24] M. ter Beek, S. Gnesi, C. Montangero, and L. Semini, "Detecting policy conflicts by model checking uml state machines," in Feature Interactions in Software and Communication Systems X, International Conference on Feature Interactions in Software and Communication Systems, ICFI 2009, 11-12 June, 2009, Lisbon, Portugal. IOS Press, 2009, pp. 59–74.

[25] K. J. Turner, S. Reiff-Marganiec, L. Blair, J. Pang, T. Gray, P. Perry, and J. Ireland, "Policy support for call control," Computer Standards and Interfaces, vol. 28, no. 6, 2006, pp. 635–649.

[26] M. ter Beek, A. Fantechi, S. Gnesi, and F. Mazzanti, "A state/event-based model-checking approach for the analysis of abstract system properties," Sci. Comput. Program., vol. 76, no. 2, 2011, pp. 119–135.

[27] M. Danelutto, P. Kilpatrick, C. Montangero, and L. Semini, "Model checking support for conflict resolution in multiple non-functional concern management," in Euro-Par 2011 Parallel Processing Workshop Proc., ser. LNCS, vol. 7155. Bordeaux: Springer, 2012, pp. 128–138.

[28] L. du Bousquet, F. Ouabdesselam, J.-L. Richier, and NicolasZuanon, "Feature interaction detection using a synchronous approach and testing," Computer Networks, vol. 32, no. 4, 2000, pp. 419–431.

[29] P. Cousot, "Abstract interpretation based formal methods and future challenges," in Informatics - 10 Years Back. 10 Years Ahead, ser. Lecture Notes in Computer Science, R. Wilhelm, Ed., vol. 2000. Springer-Verlag, 2001, pp. 138–156.

[30] N. Siegmund, S. S. Kolesnikov, C. Kästner, S. Apel, D. S. Batory, M. Rosenmüller, and G. Saake, "Predicting performance via automated feature-interaction detection," in 34th Int. Conf. on Software Engineering, ICSE 2012, Zurich, Switzerland, 2012, pp. 167–177.

[31] A. van Lamsweerde, Requirements Engineering. John Wiley & Sons, Chichester, UK, 2009.

[32] M. Calder and E. Magill, Eds., Feature Interactions in Telecommunications and Software Systems VI. IOS Press (Amsterdam), May 2000.

A Cloud Computing Benchmark Methodology using Data Envelopment Analysis (DEA)

Leonardo Menezes de Souza
Universidade Estadual do Ceará (UECE)
Av. Silas Munguba 1700
Fortaleza/CE - Brazil
Email: leonardo@insert.uece.br

Marcial Porto Fernandez
Universidade Estadual do Ceará (UECE)
Av. Silas Munguba 1700
Fortaleza/CE - Brazil
Email: marcial@larces.uece.br

***Abstract*—Cloud Computing is a new distributed computing model based on the Internet infrastructure. The computational power, infrastructure, applications, and even collaborative content distribution is provided to users through the Cloud as a service, anywhere, anytime. The adoption of Cloud Computing systems in recent years is remarkable, and it is gradually gaining more visibility. The resource elasticity with the cost reduction has been increasing the adoption of cloud computing among organizations. Thus, critical analysis inherent to cloud's physical characteristics must be performed to ensure consistent system deployment. Some applications demand more computer resources, other requests more storage or network resource. Therefore, it is necessary to propose an approach to performance measurement of Cloud Computing platforms considering the effective resource performance, such as processing rate, memory buffer refresh rate, disk I/O transfer rate, and the network latency. It is difficult to discover the amount of resources are important to a particular application. This work proposes a performance evaluation methodology considering the importance of each resource in a specific application. The evaluation is calculated using two benchmark suites: High-Performance Computing Challenge (HPCC) and Phoronix Test Suite (PTS). To define the weight for each resource, the Data Envelopment Analysis (DEA) methodology is used. The methodology is tested in a simple application evaluation, and the results are analyzed.**

Keywords–Cloud Computing; Performance evaluation; Benchmark; Methodology.

I. INTRODUCTION

The cloud computing infrastructure meets several workload requirements simultaneously, which of these are originated from Virtual Machine (VM). The evaluation addressed in this work is focused on criticality and performance on the cloud platform virtualized resources. Such evaluation is required because the performance of virtualized resources is not transparent to the network management, even when using a software monitor. Thus, it is demanded a methodology that allows to quantify the performance according to the platform particularity, using it to performance periodic measurements and to assure the promised available and reducing malfunctioning risks.

In this work, we propose a generic methodology to assess the performance of a cloud computing infrastructure; standardizing the method and covering a wide range of systems. Such methodology will serve any cloud computing structure, since it is oriented to the resources' performance. The assessment must consider the influence of each resource on the overall system performance. Then it is determined which of these resources has greater relevance to the system, aiding in deciding which infrastructure model will provide the best consumption efficiency to users, developers and managers.

This paper is an extended version of the paper presented in The Fourteenth International Conference on Networks (ICN 2015) [1]. Comparing to the original paper, this one shows more results to validate the proposal.

We consider the average performance of the hardware and network critical points, such as processing, memory buffer refresh rate, storage Input/Output (I/O) and network latency. We used two benchmarking suites to evaluate these important points: High Performance Computing Challenge (HPCC) and Phoronix Test Suite (PTS).

HPCC uses real computing kernels, allowing variable inputs and runtimes according to system capacity [2]. It consists of seven benchmarks responsible for each critical component individual analysis according to its specificity.

The PTS [3] is the basic tool of the *Cloud Harmony* [4] website, which analyzes public cloud systems all over the world. It consists of over 130 system analysis tests, which were selected by its effective handling and compatibility of results, with higher stability and likelihood when compared to benchmarks with the same goal.

From the results obtained in both benchmark suites, we analyze it using Data Envelopment Analysis (DEA), which will assign weights according to each resource's relevance in the infrastructure; then transcribe a formulation considering each resource's average performance in each deployed VM instance. The formulation considers the overhead attached to each evaluated resource, culminating in its real performance representation. The proposal was validated in a experiment done in a Datacenter running a typical Web application.

The rest of the paper is structured as follows. In Section II, we present some related work, and Section III introduces the proposed performance evaluation methodology. Section IV shows the results and Section V concludes the paper and suggests future work.

II. RELATED WORK

Ostermann [5] and Iosup [6] created a virtual platform using the Amazon Elastic Compute Cloud (EC2) [7] instances. In this scenario, the infrastructure is shared by many independent tasks, and the benchmarks will run over the Multi-Job Multi-Instance (MJMI) sample workloads. It was noticeable two main performance characteristics: the workload makespan stability, and the resource's aquisition/liberation overhead.

The performance of several cloud computing platforms, e.g., Amazon EC2, Mosso, ElasticHost and GoGrid, were suitable to using the HPCC benchmark suite. It was noticeable that cloud computing is a viable alternative to short deadline applications, because it presents low and stable response time.

It brings a much smaller delay for any cloud model when compared to scientific environment, meeting effectively to the stability, scalability, low overhead and response time criteria. The contribution of these works stands for the methodology and the metrics evaluation, besides the pioneering idea of analyzing the performance of cloud computing systems [6].

Benchmark's references for performance verification and infrastructure limitations were made in [8]. The benchmarks were classified in three categories according to the moment of the infrastructure (deployment, individual or cluster). All of them brings a sense of loss carried by virtualization. In this work, it was executed simulations to assess the Central Processing Unit (CPU)/Random Access Memory (RAM), storage I/O and network usage metrics.

It was verified that CPU usage tests have a little overhead introduced by virtualization. The I/O tests show performance gain caused by virtualization. Such fact possibly occurs because virtualization creates a new cache level, improving the I/O performance. On the other hand, there are components, which execute I/O functions that are affected by large cache, reducing performance and becoming the cache useless. It is difficult to predict the performance behavior in a specific I/O task.

The increasing complexity and dynamics in deployment of virtualized servers are highlighted in Huber [9]. The increasing of complexity is given by gradual introduction of virtual resources, and by the gap left by logical and physical resource allocation. The dynamics increasing is given by lack of direct control over hardware and by the complex iterations between workloads and applications. Results of experimentations using benchmarks presented that performance overhead rates to CPU virtualization is around 5%. Likewise, the performance overhead to memory (RAM), networks and storage I/O virtualizations reach 40%, 30% and 25%, respectively.

Different from cited works, this paper presents a proposal to evaluate a cloud computing system considering the application demand. Although it is possible to use HPCC or PTS metrics and calculate an index weighted by parameters based in operator experience, the results are not precise. Our proposal uses DEA methodology to define the relevance of each parameter and calculate a unique value to compare against other cloud providers.

III. A Methodology to Evaluate the Performance of a Cloud Computing System

Amazon Elastic Compute Cloud (Amazon EC2) is a service provided by Amazon cloud computing platform. The users can access the platform by the Amazon Web Services (AWS) interface. Amazon's offer the Amazon Machine Image in order to create a Virtual Machine (VM), which is called an *instance*, containing user's software. A user can create, deploy, and stop server instances as needed. They pay the service by the amount of hours of active server instance it used.

In each Amazon's VM, or VM instance, works as a virtual private server. To facilitate for user to choose the amount of resources they would buy, Amazon defines a set of instance size based on Elastic Compute Units. Each instance type offers different quantity of memory, CPU cores, storage and network bandwidth. The Amazon's pre-defined VM types used in this work are shown in Table I.

TABLE I. Amazon EC2 Virtual Machine Model [7].

VMs	CPUs(Cores)	RAM[GB])	Arch[bit]	Disk[GB]
m1.small	1 (1)	1,7	32	160
c1.medium	5 (2)	1,7	32	350
m1.large	4 (2)	15	64	850
m1.xlarge	8 (4)	15	64	1690
c1.xlarge	20 (8)	7	64	1690

First, we deploy VMs based on the model provided by Amazon EC2 [7]. The overall performance of the resources is not used, since virtualization generates communication overhead in the resource management. After the allocation of resources in need, we installed the benchmark suites to run the tests.

According to Jain [10], the confidence interval only applies to large samples, which must be considered from 30 (thirty) iterations. Therefore, we ran the experiments for each resource of each VM instance at least thirty times, ensuring the achievement of a satisfactory confidence interval (95%). Then, we can state that each benchmark will follow this mandatory recommendation to achieve an effective confidence interval. After the tests, we calculate the mean and the confidence interval of the obtained results, presenting a high reliability level.

In order to ponder the performed experiments, we opted for the DEA methodology; using the BCC model output-oriented (BCC-O), which involves an alternative principle to extract information from a population of results. Then, we determine the weights inherent to the VMs and the resources analyzed. We used the results of each benchmark iteration in each VM as an input, achieving the weights for each benchmark. Finally, we apply this procedure in the formulation which will be detailed later.

In short, we analyze a cloud performance simulating the behavior of applications by running benchmarks. We did an efficiency analysis from the achieved results, assigning weights to each one of them. Then, we proposed a formulation which showed the consumption ratio of each platform resource, considering the associated overhead. The execution order of activities for cloud computing performance evaluation methodology is shown in Figure 1.

A. Benchmarks

In this work, we use two benchmark suites, the HPCC [2] and PTS [3]), which will measure the performance of critical points in a cloud computing system. These benchmarks require the Message Passing Interface (MPI) [11] and Basic Linear Algebra Subprogram (BLAS) [12] library's availability to run the tests.

The benchmarks from HPCC suite ran both in local and online environments and has shown favorable results to its utilization. Then, the benchmark results showed independence and adaptability within the cloud nodes.

The HPCC benchmark suite comprises seven different tests that will stress the system hardware critical points such as is presented as follows:

- High-Performance Linpack (HPL) [13] uses 64-bit double precision arithmetics in distributed memory

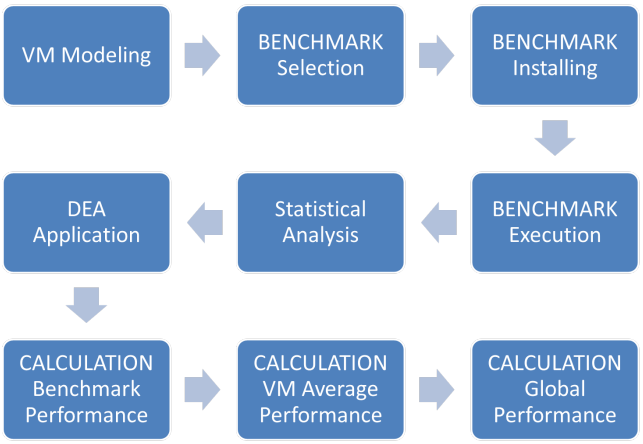


Figure 1. Cloud Computing performance evaluation methodology flowchart.

computers to measure the floating point rate of execution for solving matrices through random dense linear equations systems.

- Double-precision General Matrix Multiply (DGEMM) [14] simulates multiple floating point executions, stressing the process through double-preciosion matrix multiplication.
- PTRANS [15] has several kernels where pairs of processors communicate with each other simultaneously, testing the network total communication capability. It transposes parallel matrices and multiplies dense ones, apllying interleaving techniques.
- Fast Fourier Transform (FFT) [16] measures the floating point rate through unidimensional double-precision discrete Fourier transforms (DFT) in arrays of complex numbers.
- STREAM [17] measures the memory bandwidth that supports the processor communication (in GB/s). It also measures the performance of four long-vector operations. The array is defined to be larger than the cache of the machine, which is running the tests, privileging the memory buffer updates through inter-dependence between memory and processor.
- Random Access [18] measures the performance of random memory (main) and access memory (cache) buffer updates in multiprocessor systems. The results are given in Giga Updates Per Second (GUPS), calculated by updated memory location identification in one second. This update consists in a Read-Modification-Write (RMW) operation controlled by memory buffer and the processor.
- Effective Bandwidth Benchmark (b_{eff}) [19] measures the bandwidth efficiency (effective) through estimated latency time for processing, transmission and reception of a standard message. The message size will depend on the quotient between memory-processor ratio and 128.

Beyond the HPCC, we also used another benchmark suite to run the remaining tests and enable a bigger coverage of evaluated resources.

The PTS suite comprises more than 130 system analysis tests. We have selected the benchmarks to be part of this experiment according to its importance within the benchmarking set, minimizing inconsistencies and improving our sample space. Finally, we achieve the three most adaptive benchmarks that will be presented as follows:

- Loopback Transmission Control Protocol (TCP) Network Performance [20] is a simple Peer-to-Peer (P2P) connectivity simulation, which measures the network adapter performance in a loopback test through the TCP performance. This test is improved on this benchmark to transmit 10GB via loopback.
- RAM Speed SMP [21] measures the performance of the interaction between cache and main memories in a multiprocessor system. It allocates some memory space and starts a write-read process using 1Kb data blocks until the array limit, checking the memory subsystem speed.
- PostMark [22] creates a large pool of little files constantly updating just to measure de workload transaction rate, simulating a big Internet e-mail server. The creation, deletion, read, and attaching transactions have minimum and maximum sizes between 5Kb and 512Kb. PostMark executes 25.000 transactions with 500 files simultaneously, and after the transactions, the files are deleted, producing statistics relating its contiguous deletion.

In short, we present all benchmarks used in this work and its basic characteristics in Table II.

TABLE II. BENCHMARKS CHARACTERISTICS.

RESOURCE	BENCHMARK	UNIT
CPU	HPL DGEMM PTRANS	GFLOPs
	FFT	GB/s
MEM	STREAM RAM Speed SMP	GB/s
	Random Access	GUPS
STO	PostMark	Transactions/s
NET	b_{eff}	μs
	Loopback TCP	s

B. Resources Overhead

Simplifying the organization of the resources' performance analysis in a cloud computing system, we can split them into two requirement groups: CPU and I/O resources. Performance studies utilizing general benchmarks show that the overhead due to CPU virtualization reach 5% as was mentioned before at Section II. The host hypervisor directly controlling the hardware and managing the actual operational system, showing low overhead.

Virtualization also imposes I/O overhead, concerning memory, networks and storage. Cloud applications have specific requirements, according to their main goal. In this way, the network is critical to every single cloud application because it determines the speed with which each remaining I/O resource will work. In other words, the network must provide capability, availability, and efficiency enough to allocate resources without compromising delays.

The online content storage is just one of the most popular features of cloud computing systems. Its performance is so much dependent on memory buffer updates rate as regarding the processing rate that feeds the buffer. These two active functions significantly affect the storage services on the cloud.

Lastly, but not less important, memory is the most required resource on a cloud computing system. In distributed systems, it is considered a critical issue, because it works along with processing in the updates of running applications, user requirements, and in the data read/write coming through network adapter or storage component. So, many functions overload the resource, representing the biggest bottleneck in whole cloud computing infrastructure.

TABLE III. VIRTUALIZATION OVERHEADS [9].

RESOURCE		OVERHEAD (%)
I/O	Memory	40
	Network	30
	Storage	25
CPU	Processing	5

Each hardware resource available in the cloud computing infrastructure possesses a unique utilization quota regarding its own functioning. However, they feature interdependencies between to each other. Table III shows the overhead portions to each resource analyzed in this work. Then, we address weights based on the significance of each resource in a cloud computing infrastructure using the DEA methodology.

C. DEA Methodology

The DEA methodology is a linear programming mathematical technique, which consists of a multicriteria decision support, analyzing multiple inputs and outputs simultaneously. In this way, the DEA is capable of modeling real-world problems meeting the efficiency analysis [23].

This methodology provides comparative efficiency analysis from complex organizations obtained by its unit performance revelation so that its reference is obtained by the observation of best practices. The organizations once under DEA analyses are called Decision Making Unit (DMU)s and must utilize common resources to produce the same results. With this, will be defined efficient DMUs (those which produce maximum outputs by inputs) and the inefficient ones. The first ones are located on the efficiency frontier while the later ones under that same frontier.

In this work, we chose one model among all DEA methodology models, which is the Multipliers BCC-O model. The output orientation was chosen because of the input variables (VM instances) are fixed. The main goal is to obtain the best benchmarks' performance executed on the VMs, then we intend to obtain the larger amount of outputs by inputs. By the way, the DEA methodology was applied to parametrize the benchmarks results calculated for each resource in all VM instances.

The required terms to the weighting on the proposed formulation are generated by the BCC-O model. This mathematical model consists of the calculation of the input (VM resources) and output (benchmarks results) variables weights. In the model objective function we minimize the input weighted sum (product from input value by its respective weight) subjected to four restrictions, presented on the formulation shown in (1).

Running the model shown earlier in a linear programming solver, we can get the weight sum equal to 1, showed in (1c). The restriction of the inequality (1d) will be performed for each one of the 1500 total iterations from running instances. This model allows weights to be chosen for each DMU (VM iteractions) in a way that suits it better. The calculated weights must be greater than or equal to zero as it is shown on inequalities (1f) and (1g). The efficiency ratios of each DMU is calculated by the objective function too. Thus, the number of models to be solved is equal to the number of problem DMU.

In order to achieve the best performance of the resulting benchmarks (outputs) ran on the five VMs showed in Table I. The weights are obtained by a weighted average according to the significance of each test on the system. The greater values will have the higher weights. We consider each one of the ten benchmarks executed ran, at least, 30 times for each one of the five VMs used in this experiment, accounting for 1500 iterations. Each one of these had its respective weight calculated by DEA, then we ran a solver (BCC-O) to calculate the inputs and outputs weighted sum obeying the methodology constraints.

$$\text{Minimize} \quad ef(0) = \sum_{i=1}^{m} v_i X_{i0} + v \tag{1a}$$

$$\tag{1b}$$

$$\text{Subject to} \sum_{j=1}^{S} u_j Y_{j0} = 1 \tag{1c}$$

$$\sum_{j=1}^{S} u_j Y_{jk} - \sum_{i=1}^{m} v_i X_{ik} - v \leq 0 \tag{1d}$$

$$k = 1 \ldots n \tag{1e}$$

$$u_j \geq 0, \forall j \tag{1f}$$

$$v_i \geq 0, \forall i \tag{1g}$$

Where: $v \in \Re$, v unrestricted
u_j = output j weight
v_i = input i weight
$k \in \{1 \ldots n\}$ DMUs
$j \in \{1 \ldots s\}$ outputs of DMUs
$i \in \{1 \ldots m\}$ inputs of DMUs
Y_{jk} = output j value of DMU k
X_{ik} = input i value of DMU k

Running the model shown earlier in a linear programming solver, we can get the weight's values. The restriction of the inequality (1d) will be performed for each one of the 150 iteractions of running instances. This model allows weights to be chosen for each DMU (VM iteractions) in a way that suits it better. The calculated weights must be greater than or equal to zero, as it is shown on inequalities (1f) and (1g). The efficiency ratios of each DMU is calculated by the objective function too. Thus, the number of models to be solved is equal to the number of problem's DMU.

In order to achieve the best performance of the resulting benchmarks (outputs) ran on the VMs, the weights are obtained by a weighted mean according to the significance of each test to the system. The greater values will have the higher weights. We consider each one of the ten benchmarks executed ran at least thirty (30) times for each one of the five (5) VMs used in this experiment, accounting for 1500 iteractions. Each one of these had its respective weight calculated by DEA, then we ran a solver (BCC-O) to calculate the inputs and outputs weighted sum obeying the methodology constraints.

Concerning the constraints, first of all, the outputs' weighted sum must be equal to one, setting a parameter for assigning weights in each VM. The inputs and outputs' weights must be greater than or equal to zero. Lastly, the subtraction between the inputs and outputs' weighted sums and the scale factor, must be lower than or equal to zero. The scale factor will not be considered because it will just determine if the production feedback is increasing, decreasing or constant to a set of inputs and products. This way, weights are the factors considered on the formulation.

D. Formulation

In a cloud computing system, the required resources are allocated automatically according to user needs. All of them have a standard overhead and significance variable level according to hosted application guidance. To analyze the system performance, we used a mathematical formulation that provides evidence from utilization levels measured, and from the iteractions among resources. The DEA was used to define the weights of Performance Index.

We must consider that benchmark execution will simulate an application that overloads the assessed resource. Then, we adopted PI_{R_G} as the Resource Global Performance Index, whose variable will assume the resulting value from the product between RPI_R (Resource Real Performance Index) and the API_{R_j} (Average Performance Index by Resource in each VM Instance), as shown in (2).

$$PI_{R_G} = RPI_R \times API_{R_j} \tag{2}$$

The term RPI_R is the result from the subtraction between the maximum theoretical performance (100%) and the overhead associated to each running resource, shown in Table III. The relation is shown in (3).

$$RPI_R = (100\% - Ov_R\%) \tag{3}$$

The term API_{R_J} is calculated by the mean of each BPI_{R_j} (Benchmark Performance Index by Resource in each Instance), as it is shown in (4). BPI_{R_j} is calculated by the product sum between weights (U_{iR_j}) obtained from DEA methodology for benchmarks (i) by resource (R) in each instance (j). The term n_j stands for the amount of VMs where benchmarks were hosted. In this case, five VMs were implemented to run the tests based on the Amazon EC2 infrastructure.

$$API_{R_j} = BPI_{R_j} \div n_j \tag{4}$$

The results (X_{iR_j}) obtained from benchmarks (i), by resource (R) in each instance (j), as shown in (5), where p is the number of benchmarks and q is the number of instances. The X_{iR_j} is normalized related to maximum theoretical performance in order to permit an index independent from benchmark units, e.g., GB/s, GFLOPS, Sec.

$$BPI_{R_j} = \sum_{\substack{1 \le i \le p \\ 1 \le j \le q}} (U_{iR_j} \times X_{iR_j}) \tag{5}$$

The benchmark suites were set up to simulate each resource behavior in a cloud computing infrastructure. We will calculate the (BPI_{R_j}) Benchmarks Performance Index to each resource (R) in each instance (j), considering each benchmark running to its respective resource, and after that we calculated the mean for each resource, obtaining the API_{R_j} dividing each BPI_{R_j} by the number of VM instances n_j. In following formulation, CPU means computing resource, MEM means memory, STO means storage resource and NET means network resource.

$$\begin{aligned}
BPI_{CPU_j} &= (U_{HPL} \times X_{HPL}) + (U_{DGEMM} \times X_{DGEMM}) \\
&\quad + (U_{FFT} \times X_{FFT}) + (U_{PTRANS} \times X_{PTRANS}) \\
BPI_{MEM_j} &= (U_{STREAM} \times X_{STREAM}) + (U_{RA} \times X_{RA}) \\
&\quad + (U_{RSMP} \times X_{RSMP}) \\
BPI_{STO_j} &= (U_{BB} \times X_{BB}) + (U_{PM} \times X_{PM}) \\
BPI_{NET_j} &= (U_{BE} \times X_{BE}) + (U_{LTCP} \times X_{LTCP})
\end{aligned}$$

$$\begin{aligned}
API_{CPU_j} &= \sum BPI_{CPU_j} \div n_j \\
API_{MEM_j} &= \sum BPI_{MEM_j} \div n_j \\
API_{STO_j} &= \sum BPI_{STO_j} \div n_j \\
API_{NET_j} &= \sum BPI_{NET_j} \div n_j
\end{aligned}$$

The next step consists in solving the global performance expression:

$$\begin{aligned}
PI_{CPU_G} &= RPI_{CPU} \times API_{CPU_j} \\
PI_{MEM_G} &= RPI_{MEM} \times API_{MEM_j} \\
PI_{STO_G} &= RPI_{STO} \times API_{STO_j} \\
PI_{NET_G} &= RPI_{NET} \times API_{NET_j}
\end{aligned}$$

IV. Results and Discussion

The proposed methodology was tested in a real environment, composed by servers and network typically used in a datacenter. Although it was a small environment, all the machines was running only the benchmark software, providing correct measurements without any external interference.

The hardware used was a Dell Power Edge M1000e enclosure with six blades powered by Intel Xeon x5660 2.8 GHz processor and 128 GB 1333 MHz DDR3 RAM. All blades have 146 GB SAS HDs. The storage was a Dell Compellent with six 600 GB SAS disk and six 2.0 TB NL-SAS disk. The OS was the Linux Ubuntu 12.04 over VMWare ESXi 5.0.0 hypervisor.

All the results are based on the initial set of benchmarks showed in Section III. As we could see in Table I, we created a homogeneous environment from 1 to 21 cores based on five Amazon EC2 instances, where we run the benchmarks which will evaluate the performance on the cloud environment. The application chose was an XAMPP 1.8.1 Web server [24].

A. Benchmark Evaluation per Resource

This section shows the graphs of each benchmark evaluation by resource. It is shown the results in running the benchmark in each VM tested.

The first evaluation was the CPU performance. Figure 2 shows the results of HPCC Benchmark (CPU), comparing HPL, DGEMM and FFT benchmarks, all of then giving results in GFLOPS. There is a small discrepancy in c1.medium and m1.large result due different VMs profiles. The c1 profile provides more cache than m1 that produces a better results in DGEMM benchmark because it is based on matrix operation that takes advantage of cache memory.

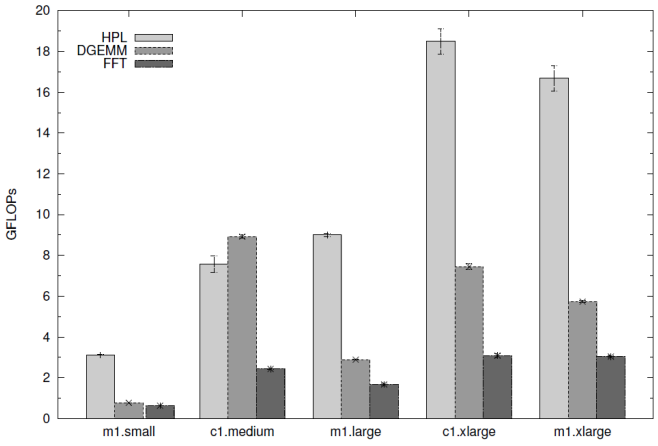


Figure 2. HPCC Benchmark (CPU): HPL x DGEMM x FFT (GFLOPS).

Figure 3 shows the results of HPCC Benchmark (CPU) PTRANS, that gives the results in GB/s. As showed in DGEMM results, there is a small discrepancy in c1.medium and m1.large result due the same reason. PTRANS benchmark is based on matrix operation that takes advantage of more cache in c1 profile.

The second evaluation was the Memory performance. Figure 4 shows the results of Phoronix Test Suite (Memory): RAM Speed SMP Integer x Float, that gives the results in MB/s. The

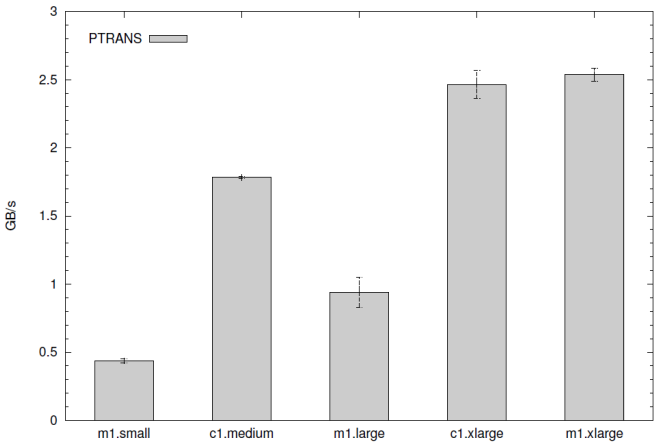


Figure 3. HPCC Benchmark (CPU): PTRANS (GB/s).

results shows small difference in all tested VMs. The PTS test is affected by hypervisor's cache memory.

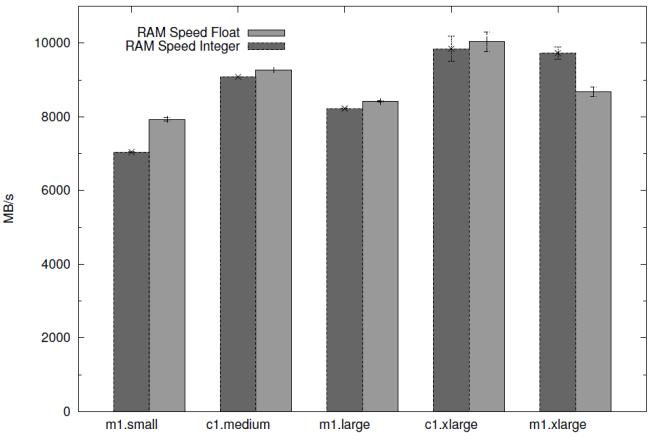


Figure 4. Phoronix Test Suite (Memory): RAM Speed SMP Integer x Float (MB/s).

Figure 5 shows the results of HPCC Benchmark (Mem) STREAM, which gives the results in GB/s. There is a small discrepancy in c1.xlarge and m1.xlarge comparing to m1.large results due the fact that xlarge profiles have more CPU cores that dispute the internal bus that reduces the memory reading performance.

Figure 6 shows the results of HPCC Benchmark (Mem) Random Access, which gives the results in Giga Updates Per Second (GUPS). There is a small discrepancy in c1.medium and m1.large result due the difference on VM profile. As the c1 profile provides more cache than m1, the memory update process takes advantage of cache memory.

The third evaluation was the Network performance. Figure 7 shows the results of HPCC Benchmark (Network): B_{eff}, which gives the results in milliseconds. The results shows that m1 profile gives bad network performance.

Figure 8 shows the results of HPCC Benchmark (Network): Loopback TCP, which gives the results in seconds. These tests measure the network adapter performance, affected by processor performance.

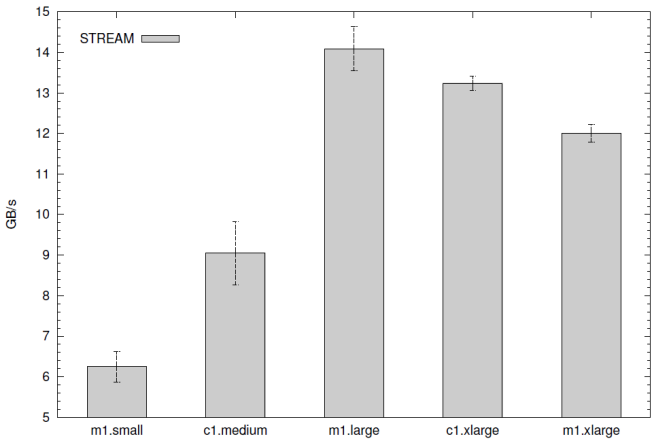


Figure 5. HPCC Benchmark (Memory): STREAM (GB/s).

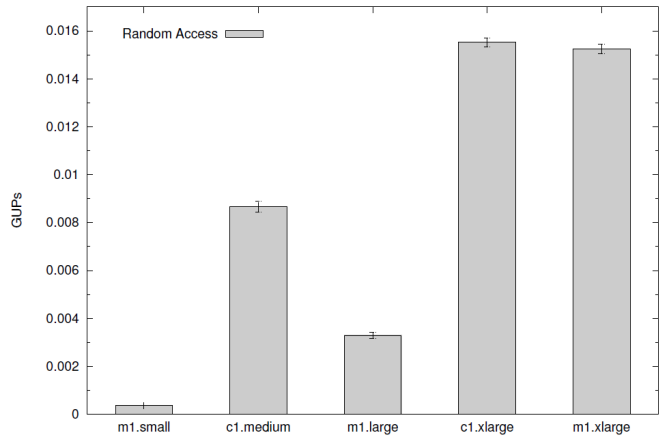


Figure 6. HPCC Benchmark (Memory): Random Access (GUPs).

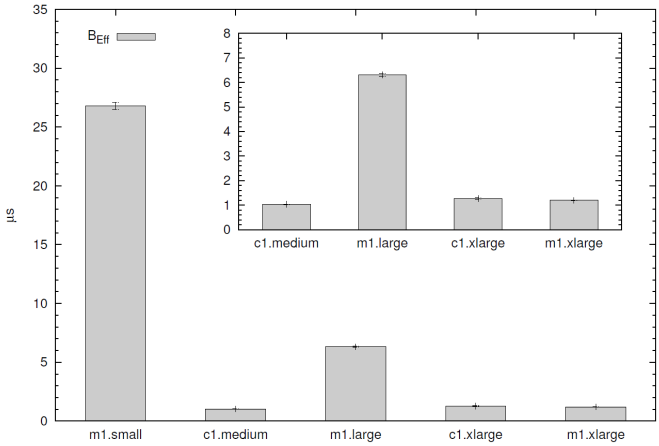


Figure 7. HPCC Benchmark (Network): B_{eff} (ms).

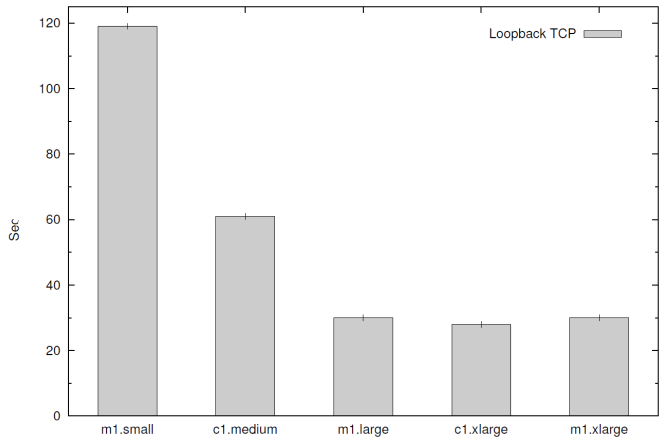


Figure 8. HPCC Benchmark (Network): Loopback TCP.

The fourth evaluation was the Network performance. Figure 9 shows the results of Phoronix Test Suite (Storage): PostMark, which gives the results in transactions per seconds. The results show the performance is related to VM performance because all of then used the same disk storage system.

B. DEA Analysis

After running each benchmark, we generate Table IV, which shows the efficiency index of each experiment related to maximum theoretical performance. The normalization is necessary to compare different units from benchmarks. Then, we calculated its efficiency percentage to use it on the proposed formulation.

In order to consider the results from the benchmark experiments, we used DEA methodology through BCC-O model (output-oriented). Beyond the efficiency index calculation, we calculate the output variable weights (benchmark results). In this way, we minimize the inputs weighted sum dividing it by the outputs' weighted sum of the benchmark at hand. After that, we ran a BCC-O solver to address weights to each benchmark, considering each VM instance according to its influence in the obtained results shown in Table IV. Table V shows the weights calculated by the BCC-O solver that will influence the performance of each resource attached to each benchmark in each VM.

The benchmark results were shown in Table IV and the efficiency index were calculated by DEA methodology (BCC-O) in Table V. Applying these results on (5), its two factors will assume values for benchmark performance to each resource in each instance (X_{iRj}), considering the DEA assigned weight to each benchmark result (U_{iRj}). We can observe the more the resource is used, greater is the weight assigned to it.

We can see in Figure 10 that the network performance is clearly greater than the rest, and the memory is the only resource that has an index relatively close. These resources are the most affected ones by the overhead issue, justifying their bottleneck condition. Figure 11 shows the relevance of each instance through benchmark execution. The c1 instances have very similar performances because they both have a processor/memory ratio which allows achieving quite satisfying performance levels.

With the two performance results in hands, we should get the resource's average by instance, applying the formulation shown in (4). Then, we calculate the global index considering the overhead rate by the index found by each resource as shown in (2).

TABLE IV. BENCHMARK RESULT FOR EACH VM (X_{iRj}) RELATED TO MAXIMUM THEORETICAL PERFORMANCE.

BENCHMARKS		m1.small	c1.medium	m1.large	m1.xlarge	c1.xlarge
CPU	HPL	4.64%	11.27%	14.84%	24.81%	27.51%
	DGEMM	1.15%	13.27%	4.30%	8.54%	11.08%
	FFT	0.94%	3.62%	2.49%	4.52%	4.59%
	PTRANS	6.83%	27.86%	14.71%	39.63%	38.52%
MEM	RAMSpeed SMP/Integer	22.01%	28.38%	25.7%	30.4%	30.77%
	RAMSpeed SMP/Float	24.46%	28.96%	26.30%	27.13%	31.36
	STREAM	19.53%	28.27%	44.02%	37.53%	41.36%
	RandomAccess	0.41%	9.82%	3.73%	17.3%	17.6%
NET	b_{eff}	98.2%	99.9%	98.8%	99.5%	99.4%
	Loopback TCP	0.58%	62.07%	92.65%	94.34%	96.02%
STO	PostMark	3.75%	4.42%	13.99%	13.00%	14.26%

TABLE V. WEIGHTS ADDRESSED TO RESOURCES TO EACH VM (U_{iRj}).

BENCHMARKS		m1.small	c1.medium	m1.large	m1.xlarge	c1.xlarge
CPU	HPL	0.77	0.13	0.66	0.28	0.51
	DGEMM	0.88	0.42	0.23	0.29	0.20
	FFT	0.003	0.58	0.25	0.5	0.58
	PTRANS	0.15	0.32	0.07	0.33	0.43
MEM	RAMSpeed SMP/Integer	0.38	0.78	0.17	0.42	0.37
	RAMSpeed SMP/Float	0.46	0.96	0.3	0.68	0.65
	STREAM	0.14	0.18	0.67	0.33	0.61
	RandomAccess	0.91	0.93	0.37	0.73	0.56
NET	b_{eff}	0.42	0.59	0.48	0.55	0.43
	Loopback TCP	0.24	0.43	0.33	0.28	0.48
STO	PostMark	0.57	0.24	0.19	0.42	0.62

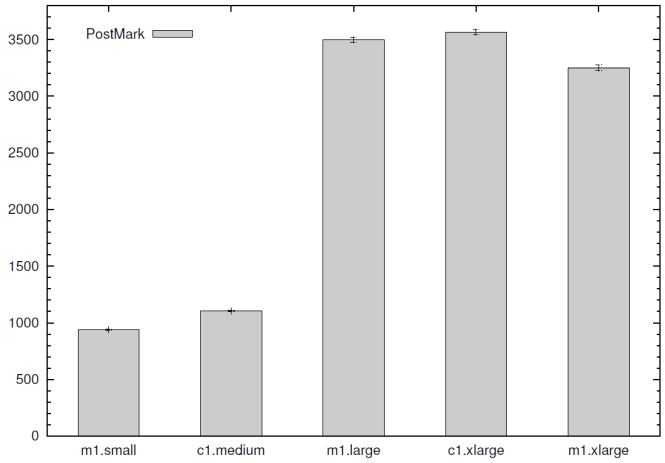


Figure 9. Phoronix Test Suite (Storage): PostMark (Transaction/s).

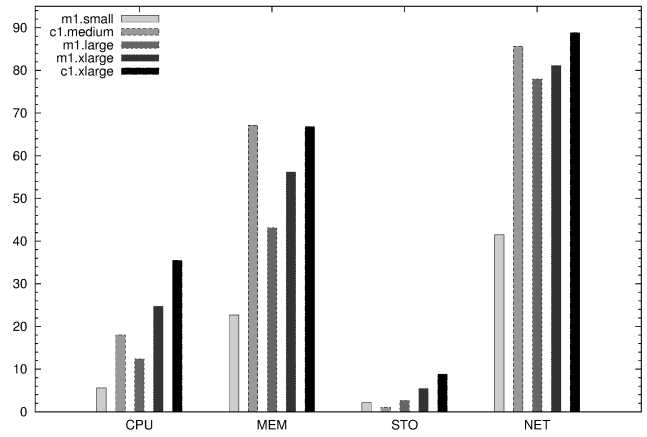


Figure 10. Benchmark Performance by Resource.

From these results we verified that the memory and network performances are the most relevant to a cloud computing system. These two resources, when well balanced, leverage the cloud computing infrastructure managing workloads, reaffirming its bottleneck condition. In this way, this proposal gives more information regarding resource performance relevance in application when comparing to the work of Huber [9].

To analyze the proposal scalability, we repeat the data collected from the six servers 10,000 times, in order to emulate the benchmark of 60,000 server, comparable to a big datacenter. The overall time to execute the methodology was below 5 seconds, a reasonable time to obtain the results.

V. CONCLUSION AND FUTURE WORK

In this work, we could observe that the benchmarks had met the simulation needs very well, overloading the resources efficiently, returning real-world results. The DEA methodology helped us to analyze the efficiency of each experiment, providing an efficiency index (weight) to benchmarks in each instance implemented, for each resource evaluated. Finally, the proposed formulation highlighted the impact of resource's overhead on the global performance evaluation.

Then, we concluded that, in a generic Web application, the memory and network resource performance is the most relevant to a cloud computing system, and for this reason, they are considered the bottlenecks. We confirmed that the resource performance evaluated here is directly proportional to the overhead execution rates, assigned in [9].

Since develop an application to be hosted on a cloud environment to measure its resource consumption rate, or its behavior during a VM migration process, until configure the benchmarks in a more aggressive way, generating more data

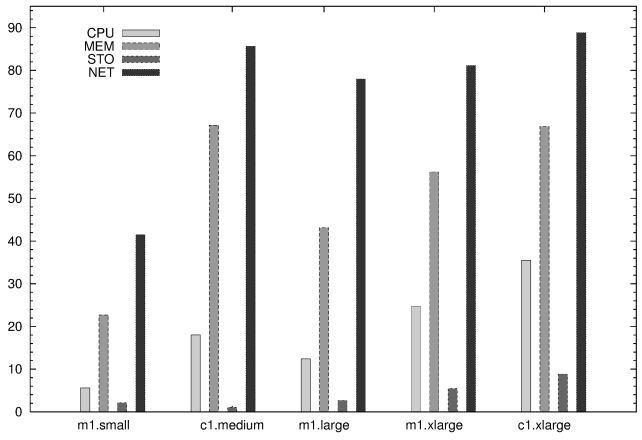


Figure 11. Benchmark Performance by Instance.

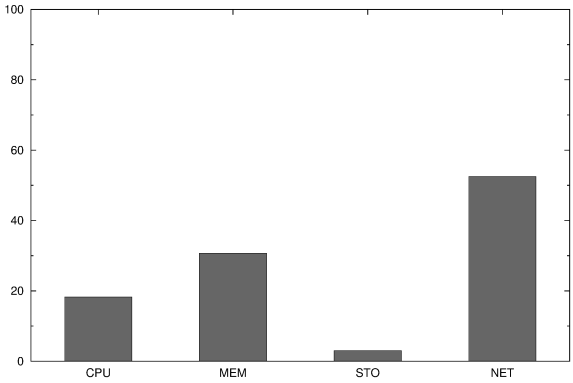


Figure 12. Global Resource Performance.

blocks. Then, we should pay attention to cloud computing system constant evolution to make possible the use of the approach proposed in this work.

REFERENCES

[1] L. M. Souza and M. P. Fernandez, "Performance evaluation methodology for cloud computing using data envelopment analysis," in The Fourteenth International Conference on Networks (ICN 2015), IARIA. Barcelona, Spain: IARIA, April 2015, pp. 58–64.

[2] J. Dongarra and P. Luszczek, "HPCC High Performance Computing Challenge," Last accessed, March 2015. [Online]. Available: http://icl.eecs.utk.edu/hpcc

[3] M. Larabel and M. Tippett, "Phoronix Test Suite," Last accessed, March 2015. [Online]. Available: http://www.phoronix-test-suite.com

[4] J. Read, "Cloud Harmony: Benchmarking the Cloud," Last accessed, March 2015. [Online]. Available: http://www.cloudharmony.com

[5] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "An Early Performance Analysis of EC2 Cloud Computing Services for Scientific Computing," Cloud Computing, 2010, pp. 115–131.

[6] A. Iosup, S. Ostermann, M. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "Performance analysis of cloud computing services for many-tasks scientific computing," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, 2011, pp. 931–945.

[7] Amazon, "Amazon Elastic Compute Cloud EC2," Last accessed, March 2015. [Online]. Available: http://aws.amazon.com/ec2

[8] N. Cardoso, "Virtual clusters sustained by cloud computing infrastructures," Master's thesis, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal, December 2011.

[9] N. Huber, M. von Quast, M. Hauck, and S. Kounev, "Evaluating and Modeling Virtualization Performance Overhead for Cloud Environments," in 1st International Conference on Cloud Computing and Services Science, 2011, pp. 7–9.

[10] R. Jain, The Art of Computer Systems Performance Analysis. John Wiley & Sons, 2008.

[11] J. Dongarra, R. Hempel, T. Hey, and D. Walker, "The Message Passing Interface (MPI) Standard," Last accessed, March 2015. [Online]. Available: https://mcs.anl.gov/research/projects/mpi

[12] C. Lawnson, R. Hanson, D. Kincaid, and F. Krogh, "Basic Linear Algebra Subprograms for FORTRAN Usage," ACM Transactions on Mathematical Software (TOMS), vol. 5, no. 3, 1979, pp. 308–323.

[13] A. Petitet, R. Whaley, J. Dongarra, and A. Cleary, "HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers," Last accessed, March 2015. [Online]. Available: http://netlib.org/benchmark/hpl/

[14] J. Dongarra, I. Duff, J. Croz, and S. Hammarling, "Subroutine DGEMM," Last accessed, March 2015. [Online]. Available: http://www.netlib.org/blas/dgemm

[15] T. Hey, J. Dongarra, and H. R., "Parkbench Matrix Kernel Benchmarks," Last accessed, March 2015. [Online]. Available: http://www.netlib.org/parkbench/html/matrix-kernels.html

[16] M. Frigo and S. Johnson, "benchFFT," Last accessed, March 2015. [Online]. Available: http://www.fftw.org/benchfft/

[17] J. McCalpin, "STREAM: Sustainable Memory Bandwidth in High Performance Computers," Last accessed, March 2015. [Online]. Available: http://www.cs.virginia.edu/stream/

[18] D. Koester and B. Lucas, "Random Access," Last accessed, March 2015. [Online]. Available: http://icl.cs.utk.edu/projectsfiles/hpcc/RandomAccess/

[19] R. Rabenseifner and G. Schulz, "Effective Bandwidth Benchmark," Last accessed, March 2015. [Online]. Available: https://fs.hlrs.de/projects/par/mpi//b_eff/

[20] M. Larabel and M. Tippett, "Loopback TCP Network Performance," Last accessed, March 2015. [Online]. Available: http://openbenchmarking.org/test/pts/network-loopback

[21] R. Hollander and P. Bolotoff, "RAMspeed," Last accessed, March 2015. [Online]. Available: http://alasir.com/software/ramspeed/

[22] J. Katcher, "PostMark: A New File System Benchmark," Last accessed, March 2015. [Online]. Available: http://www.netapp.com/technology/level3/3022.html

[23] W. W. Cooper, L. M. Seiford, and K. Tone, Data Envelopment Analysis: A Comprehensive Text with Models, Applications, References and DEA-Solver Software. Second Edition. Springer, 2007.

[24] K. Seidler and K. Vogelgesang, "XAMPP Distribution Apache + MySQL + PHP + Perl," Last accessed, March 2015. [Online]. Available: https://www.apachefriends.org

Frameworks for Natural Language Processing of Textual Requirements

Andres Arellano
Government of Chile,
Santiago, Chile
Email: andres.arellano@gmail.com

Edward Zontek-Carney
Northrop Grumman Corporation,
Baltimore, MD 21240, USA
Email: Ecarney1@umd.edu

Mark A. Austin
Department of Civil Engineering,
University of Maryland,
College Park, MD 20742, USA
Email: austin@isr.umd.edu

***Abstract*—Natural language processing is the application of automated parsing and machine learning techniques to analyze standard text. Applications of NLP to requirements engineering include extraction of ontologies from a requirements specification, and use of NLP to verify the consistency and/or completeness of a requirements specification. This paper describes a new approach to the interpretation, organization, and management of textual requirements through the use of application-specific ontologies and natural language processing. We also design and exercise a prototype software tool that implements the new framework on a simplified model of an aircraft.**

Keywords-Systems Engineering; Ontologies; Natural Language Processing; Requirements; Rule Checking.

I. INTRODUCTION

Problem Statement. Model-based systems engineering development is an approach to systems-level development in which the focus and primary artifacts of development are models, as opposed to documents. This paper describes a new approach to the interpretation, organization, and management of textual requirements through the use of application-specific ontologies and natural language processing. It builds upon our previous work in exploring ways in which model-based systems engineering might benefit from techniques in natural language processing [1] [2].

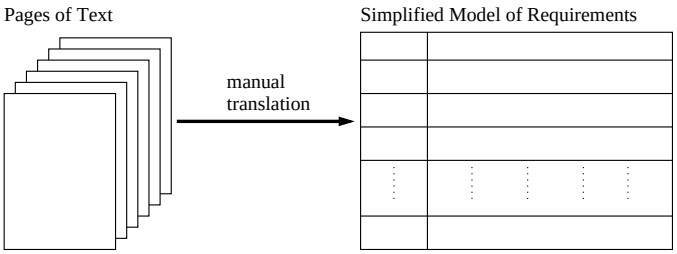


Figure 1. Manual translation of text into high-level textual requirements.

As engineering systems become increasingly complex the need for automation arises. A key required capability is the identification and management of requirements during the early phases of the system design process, when errors are cheapest and easiest to correct. While engineers are looking for semi-formal and formal models to work with, the reality remains that many large-scale projects begin with hundreds – sometimes thousands – of pages of textual requirements, which may be inadequate because they are incomplete, under specified, or perhaps ambiguous. State-of-the art practice (see Figure 1) involves the manual translation of text into a semi-formal format (suitable for representation in a requirements database). This is a slow and error prone process. A second key problem is one of completeness. For projects defined by hundreds/thousands of textual requirements, how do we know a system description is complete and consistent?

Scope and Objectives. Looking ahead, our work is motivated by a strong need for computer processing tools that will help requirements engineers overcome and manage these challenges. During the past twenty years, significant work has been done to apply natural language processing (NLP) to the domain of requirements engineering [3] [4] [5]. Applications range from using NLP to extract ontologies from a requirements specification, to using NLP to verify the consistency and/or completion of a requirements specification.

Our near-term research objectives are to use modern language processing tools to scan and tag a set of requirements, and offer support to systems engineers in their task of defining and maintaining a comprehensive, valid and accurate body of requirements. The general idea is as follows: Given a set of textual descriptions of system requirements, we could analyze them using natural language processing tools, extracting the objects or properties that are referenced within the requirements. Then, we could match these properties against a defined ontology model corresponding to the domain of this particular requirement. Such a system would throw alerts in case of system properties lacking requirements, and requirements that are redundant and/or conflicting.

Figure 2 shows the framework for automated transformation of text (documents) into textual requirements (semi-formal models) described in this paper. Briefly, NLP processing techniques are applied to textual requirements to identify parts of speech – sentences are partitioned into words and then classified as being parts of speech (e.g., nouns, verbs, etc.). Then, the analyzed text is compared against semantic models consisting of domain ontologies and ontologies for specific applications. System ontologies are matched with system properties; subsystem ontologies are matched with subsystem properties, and component ontologies are matched with component properties. Feedback is necessary when semantic descriptions of applications do not have complete coverage, as defined by the domain ontologies.

The contents of this paper are as follows: Section II

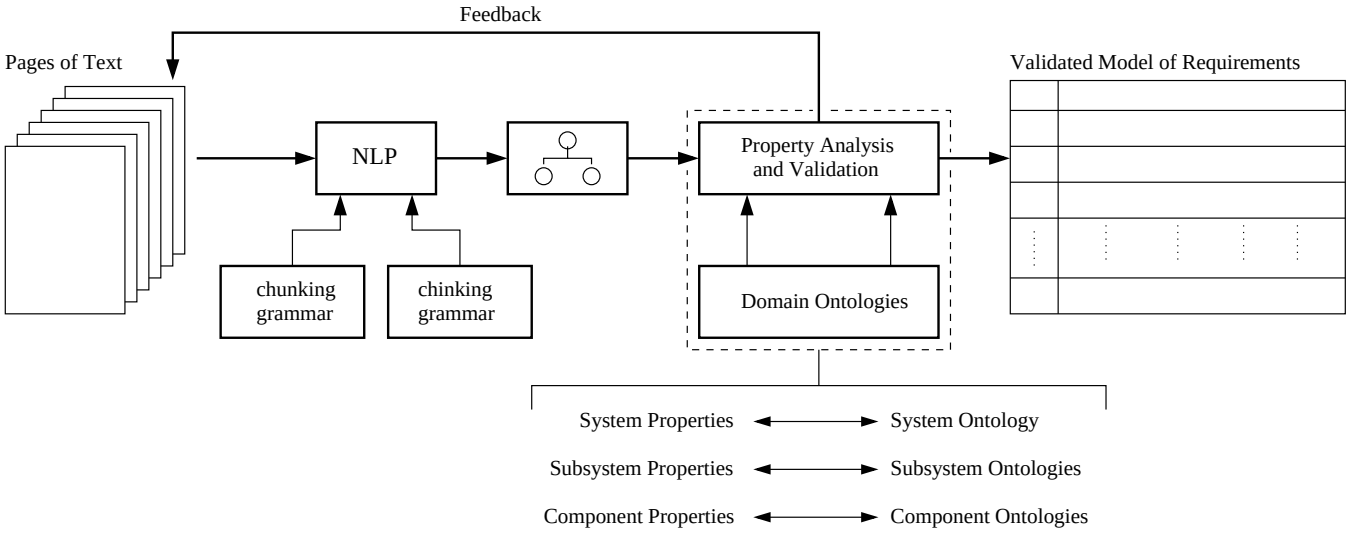


Figure 2. Framework for automated transformation of text (documents) into textual requirements (semi-formal models).

explains the role that semantics can play in modern engineering systems design and management. Its second purpose is to briefly explain state-of-the-art capability in automatic term recognition and automatic indexing. Section III describes two aspects of our work: (1) Working with NLTK, and (2) Chunking and Chinking. The framework for integration of NLP with ontologies and textual requirements is covered in Section IV. Two applications are presented in Section V: (1) Requirements and ontologies for a simple aircraft application, and (2) A framework for the explicit representation of multiple ontologies. Sections VI and VII discuss opportunities for future work and the conclusions of this study.

II. State-of-the-Art Capability

Role of Semantics in Engineering Systems Design and Management. A tenet of our work is that methodologies for strategic approaches to design will employ semantic descriptions of application domains, and use ontologies and rule-based reasoning to enable validation of requirements, automated synthesis of potentially good design solutions, and communication (or mappings) among multiple disciplines [6] [7] [8]. A key capability is the identification and management of requirements during the early phases of the system design process, where errors are cheapest and easiest to correct. The systems architecture for state-of-the-art requirements traceability and the proposed platform model [9], [10] is shown in the upper and lower sections of Figure 3. In state-of-the-art traceability mechanisms design requirements are connected directly to design solutions (i.e., objects in the engineering model). Our contention is that an alternative and potentially better approach is to satisfy a requirement by asking the basic question: What design concept (or group of design concepts) should I apply to satisfy a requirement? Design solutions are the instantiation/implementation of these concepts. The proposed architecture is a platform because it contains collections of domain-specific ontologies and design rules that will be reusable across applications. In the lower half of Figure 3, the textual requirements, ontology, and engineering models provide distinct views of a design: (1) Requirements are a statement of "what is required." (2) Engineering models are a statement of "how the required functionality and performance might be achieved," and (3) Ontologies are a statement of "concepts justifying a tentative design solution." During design, mathematical and logical rules are derived from textual requirements which, in turn, are connected to elements in an engineering model. Evaluation of requirements can include checks for satisfaction of system functionality and performance, as well as identification of conflicts in requirements themselves. A key benefit of our approach is that design rule checking can be applied at the earliest stage possible – as long as sufficient data is available for the evaluation of rules, rule checking can commence; the textual requirements and engineering models need not be complete. During the system operation, key questions to be answered are: What other concepts are involved when a change occurs in the sensing model? What requirement(s) might be violated when those concepts are involved in the change? To understand the inevitable conflicts and opportunities to conduct trade space studies, it is important to be able to trace back and understand cause-and-effect relationships between changes at system-component level, and their effect on stakeholder requirements. Present-day systems engineering methodologies and tools, including those associated with SysML [11] are not designed to handle projects in this way.

Automatic Term Recognition and Automatic Indexing. Strategies for automatic term recognition and automatic indexing fall into the general area of computational linguistics [12]. Algorithms for single-term indexing date back to the 1950s, and for indexing two or more words to the 1970s [13]. Modern techniques for multi-word automatic term recognition are mostly empirical, and employ combinations of linguistic information (e.g., part-of-speech tagging) and statistical information acquired from the frequency of usage of terms in candidate documents [14] [15]. The resulting terms can be useful in more complex tasks such as semantic search, question-answering, identification of technical terminology, automated construction of glossaries for a technical domain, and ontology construction [16] [17] [18].

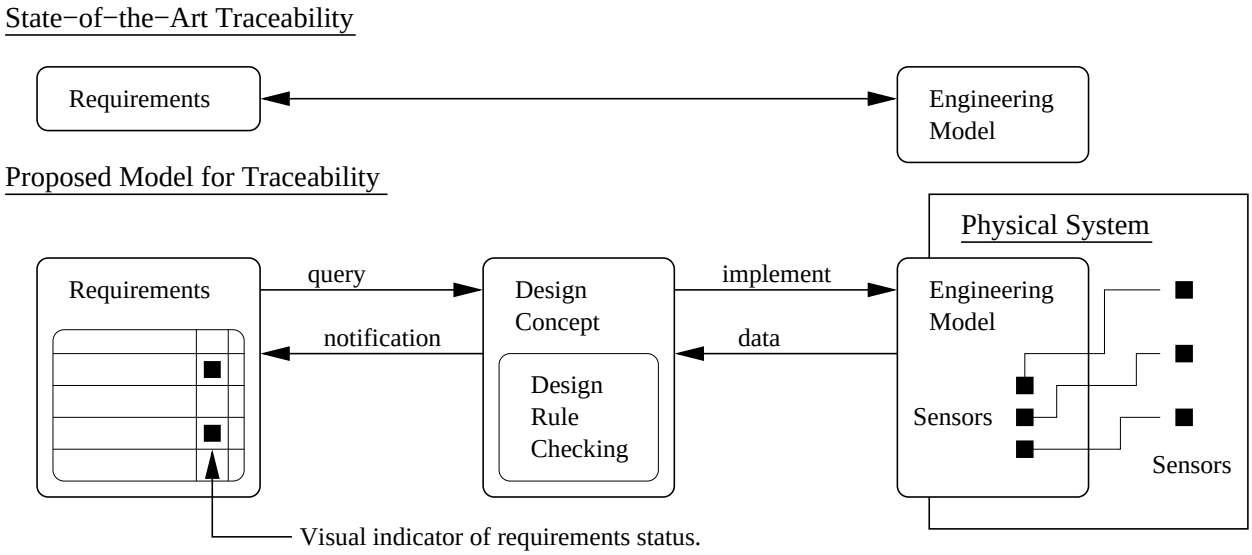


Figure 3. Schematics for: (top) state-of-the-art traceability, and (bottom) proposed model for ontology-enabled traceability for systems design and management.

III. NATURAL LANGUAGE PROCESSING OF REQUIREMENTS

Working with NLTK. The Natural Language Toolkit (NLTK) is a mature open source platform for building Python programs to work with human language data [19].

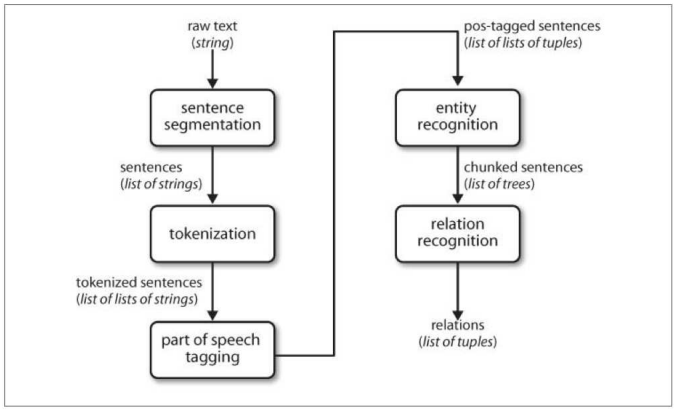


Figure 4. Information extraction system pipeline architecture.

Figures 2 and 4 show the essential details of a pipeline for text (documents) to textual requirements (semi-formal models) transformation. NLTK provides the basic pieces to accomplish those steps, each one with different options and degrees of freedom. Starting with an unstructured body of words (i.e., raw text), we want to obtain sentences (the first step of abstraction on top of simple words) and have access to each word independently (without losing its context or relative positioning to its sentence). This process is known as *tokenization* and it is complicated by the possibility of a single word being associated with multiple token types. Consider, for example, the sentence: “These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule.” The abbreviated script of Python code is as follows:

```
text = "These prerequisites are known as (computer)
        system requirements and are often used as a
        guideline as opposed to an absolute rule."
tokens = nltk.word_tokenize(my_string)
print tokens
=>
['These', 'prerequisites', 'are', 'known', 'as',
 '(', 'computer', ')', 'system', 'requirements',
 'and', 'are', 'often', 'used', 'as', 'a',
 'guideline', 'as', 'opposed', 'to', 'an',
 'absolute', 'rule', '.']
```

The result of this script is an array that contains all the text’s tokens, each token being a word or a punctuation character. After we have obtained an array with each token (i.e., word) from the original text, we may want to normalize these tokens. This means: (1) Converting all letters to lower case, (2) Making all plural words singular ones, (3) Removing *ing* endings from verbs, (4) Making all verbs be in present tense, and (5) Other similar actions to remove meaningless differences between words. In NLP jargon, the latter is known as *stemming*, in reference to a process that strips off affixes and leaves you with a stem [20]. NLTK provides us with higher level *stemmers* that incorporate complex rules to deal with the difficult problem of stemming. The Porter stemmer that uses the algorithm presented in [21], the Lancaster stemmer, based on [22], or the built in lemmatizer – Stemming is also known as *lemmatization*, referencing the search of the *lemma* of which one is looking an inflected form [20] – found in WordNet. Wordnet is an open lexical database of English maintained by Princeton University [23]. The latter is considerably slower than all the other ones, since it has to look for the potential stem into its database for each token.

The next step is to identify what role each word plays on the sentence: a noun, a verb, an adjective, a pronoun, preposition, conjunction, numeral, article and interjection [24]. This process is known as *part of speech tagging*, or simply *POS tagging* [25]. On top of POS tagging we can identify the *entities*. We can think of these *entities* as “multiple word nouns” or objects that are present in the text. NLTK provides

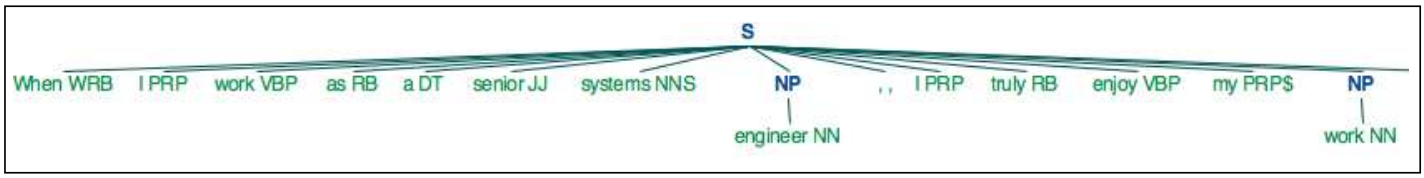


Figure 5. Output from first step on building chunking grammar. Purpose: Simply pick nouns from test sentence.

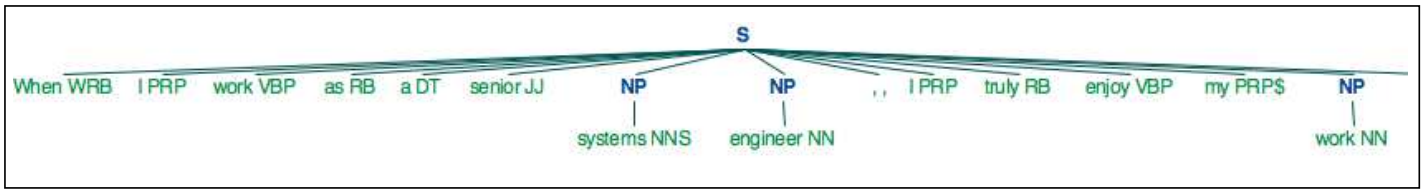


Figure 6. Output from second step on building chunking grammar. Purpose: Identify noun phrases.

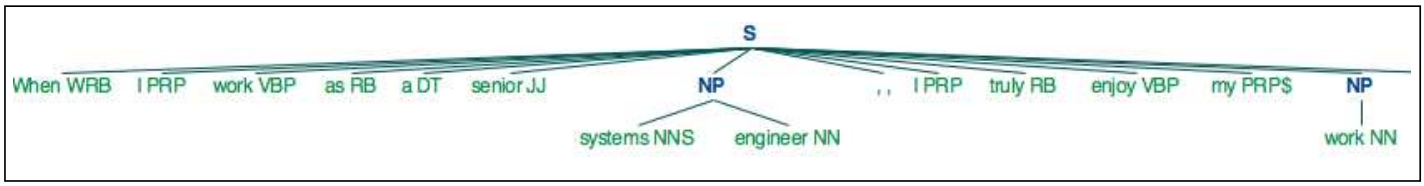


Figure 7. Output from third step on building chunking grammar. Purpose: Form noun phrases.

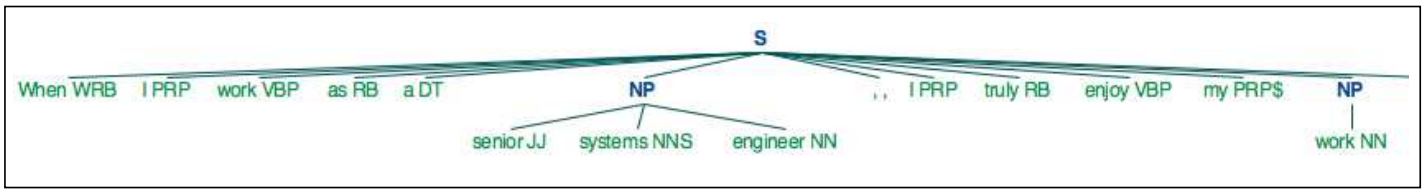


Figure 8. Output from fourth step on building chunking grammar. Purpose: Identify the adjective preceding the first noun phrase.

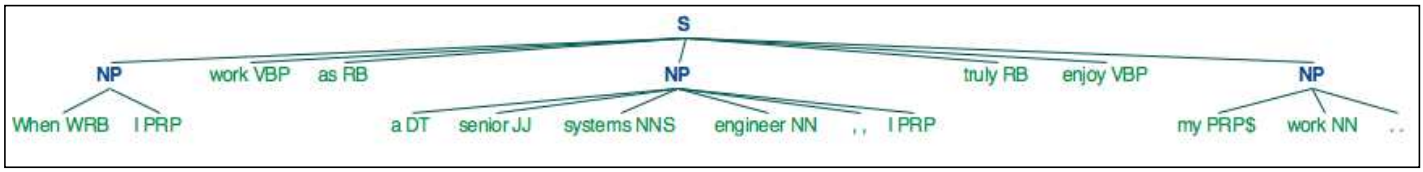


Figure 9. Output from the example on chinking. Purpose: Exclude base verbs and adverbs.

an interface for tagging each token in a sentence with supplementary information such as its part of speech. Several taggers are included, but an *off-the-shelf* one is available, based on the Penn Treebank tagset [26]. The following listing shows how simple is to perform a basic part of speech tagging.

```
my_string = "When I work as a senior systems
            engineer, I truly enjoy my work."
tokens = nltk.word_tokenize(my_string)
print tokens

tagged_tokens = nltk.pos_tag(tokens)
print tagged_tokens
=>
[('When', 'WRB'), ('I', 'PRP'), ('work', 'VBP'),
 ('as', 'RB'), ('a', 'DT'), ('senior', 'JJ'),
 ('systems', 'NNS'), ('engineer', 'NN'), (',', ','),
 ('I', 'PRP'), ('truly', 'RB'), ('enjoy', 'VBP'),
 ('my', 'PRP$'), ('work', 'NN'), ('.', '.')]
```

The first thing to notice from the output is that the tags are two or three letter codes. Each one represent a lexical category or part of speech. For instance, WRB stands for *Wh-adverb*, including *how*, *where*, *why*, etc. PRP stands for *Personal pronoun*; *RB* for *Adverb*; *JJ* for *Adjective*, *VBP* for *Present verb tense*, and so forth [27]. These categories are more detailed than presented in [24], but they can all be traced back to those ten major categories. It is important to note the possibility of one-to-many relationships between a word and the possible tags. For our test example, the word *work* is first classified as a verb, and then at the end of the sentence, is classified as a noun, as expected. Moreover, we found two nouns (i.e., objects), so we can affirm that the text is saying something about *systems*, *an engineer* and *a work*. But we know more than that. We are not only referring to *an engineer*, but to a *systems engineer*, and not only a *systems engineer*, but

a *senior systems engineer*. This is our *entity* and we need to *recognize* it from the text. In order to do this, we need to somehow tag groups of words that represent an entity (e.g., sets of nouns that appear in succession: *('systems', 'NNS')*, *('engineer', 'NN')*). NLTK offers regular expression processing support for identifying groups of tokens, specifically noun phrases, in the text.

Chunking and Chinking. Chunking and chinking are techniques for extracting information from text. Chunking is a basic technique for segmenting and labeling multi-token sequences, including noun-phrase chunks, word-level tokenization and part-of-speech tagging. To find the chunk structure for a given sentence, a regular expression parser begins with a flat structure in which no tokens are chunked. The chunking rules are applied in turn, successively updating the chunk structure. Once all of the rules have been invoked, the resulting chunk structure is returned. We can also define patterns for what kinds of words should be excluded from a chunk. These unchunked words are known as chinks. In both cases, the rules for the parser are specified defining *grammars*, including patterns, known as *chunking*, or excluding patterns, known as *chinking*.

Figures 5 through 8 illustrate the progressive refinement of our test sentence by the chunking parser. The purpose of the first pass is to simply pick the nouns from our test sentence. This is accomplished with the script:

```
grammar     = "NP: {<NN>}"
chunker     = nltk.RegexpParser(grammar)
chunks_tree = chunker.parse(tagged_tokens)
```

Figure 5 is a graphical representation of the results – NLTK identifies "engineer" as a noun. But even this seems not to be correctly done since we are missing the noun systems. The problem is that our grammar is overly simple and cannot even handle noun modifiers, such as NNS for the representation of plural nouns. The second version of our script:

```
grammar = "NP: {<NN.*>}"
chunker = nltk.RegexpParser(grammar)
chunks_tree = chunker.parse(tagged_tokens)
```

aims to include different types of nouns. The output is shown in Figure 6. Now we can see all three nouns properly identified. Unfortunately, the first two are not forming a single noun phrase, but two independent phrases. The refined script:

```
grammar = "NP: {<NN.*>+}"
chunker = nltk.RegexpParser(grammar)
chunks_tree = chunker.parse(tagged_tokens)
```

take care of this problem by adding a match-one-or-more operator `*`. The output is shown in Figure 7. The final script:

```
grammar = "NP: {<JJ.*>*<NN.*>+}"
chunker = nltk.RegexpParser(grammar)
chunks_tree = chunker.parse(tagged_tokens)
```

advances the parsing process a few steps further. We already know that we want to consider any kind of adjectives, so we add the match-one-or-more operator `*` after the adjective code `JJ`. And we use `*` to permit other words to be present between the adjective and the noun(s). Figure 8 shows the output for this last step. We have identified two entities, *senior systems engineer* and *work*, and that is precisely what we want. Incremental development of the chunking grammar is complete.

Chinking is the complementary process of removing tokens from a chunk. The script:

```
grammar = r"""
  NP: {<.*>+}
      }<VB.*>{
      }<RB.*>{
  """
chunker = nltk.RegexpParser(grammar)
chunks_tree = chunker.parse(tagged_tokens)
```

says chunk everything (i.e., `NP: {<.*>+}`, and then remove base verbs (i.e., `VB`) and adverbs (i.e., `RB`) from the chunk. When this script is executed on our test sentence the result is three noun phrase (i.e., `NP`) trees, as shown along the bottom of Figure 9.

IV. Systems Integration

Integration of NLP with Ontologies and Textual Requirements. In order to provide a platform for the integration of natural language processing, ontologies and systems requirements, and to give form to our project, we built *TextReq Validation*, a web based software that serves as a proof of concept for our objectives. The software stores ontology models in a relational database (i.e., tables), as well as a system with its requirements. It can do a basic analysis on these requirements and match them against the model's properties, showing which ones are covered and which ones are not.

The software has two main components: The web application that provides the user interfaces, handles the business logic, and manages the storage of models and systems. This component was built using Ruby on Rails (RoR), a framework to create web applications following the Model View Controller pattern [28]. The views and layouts are supported by the front-end framework Bootstrap [29]; these scripts are written using Python.

Figure 10 is collage of elements in the system architecture and application models and controllers. The model-view-controller software architecture for TextReq is shown in the top left-hand schematic. The interface between the web application and the Python scripts is handled through streams of data at a system level. The content of the streams uses a simple *key/value* data structure, properly documented. The right-hand schematic is a UML diagram of the application models. The *models* corresponding to the MVC architecture of the web application, reveal the simple design used to represent an Ontology and a System. The first one consists of a Model – named after an Ontology Model, and not because it is a MVC model – that has many Entities. The Entities, in turn, have many Properties. The latter is even simpler, consisting of only a *System* that has many *System Requirements*. Most of the business logic resides in the models; notice, in particular, system-level interpretation of results from the natural language processing. And finally, the bottom left schematic is a collection of UML diagrams for the application controllers. Due to TextReq's simplicity, its controllers and views are mostly

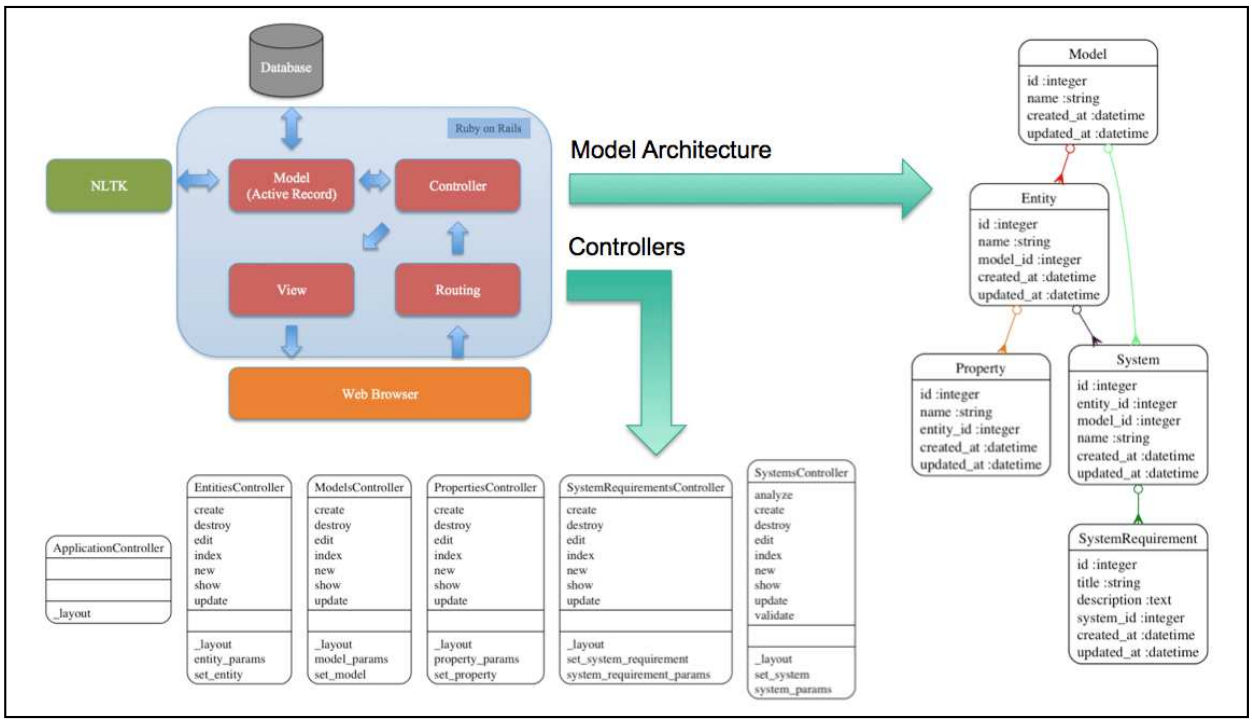


Figure 10. System architecture collage. Top left: Software architecture for TextReq validation. Bottom left: UML diagram of application controllers. Right-hand side: UML diagram of application models.

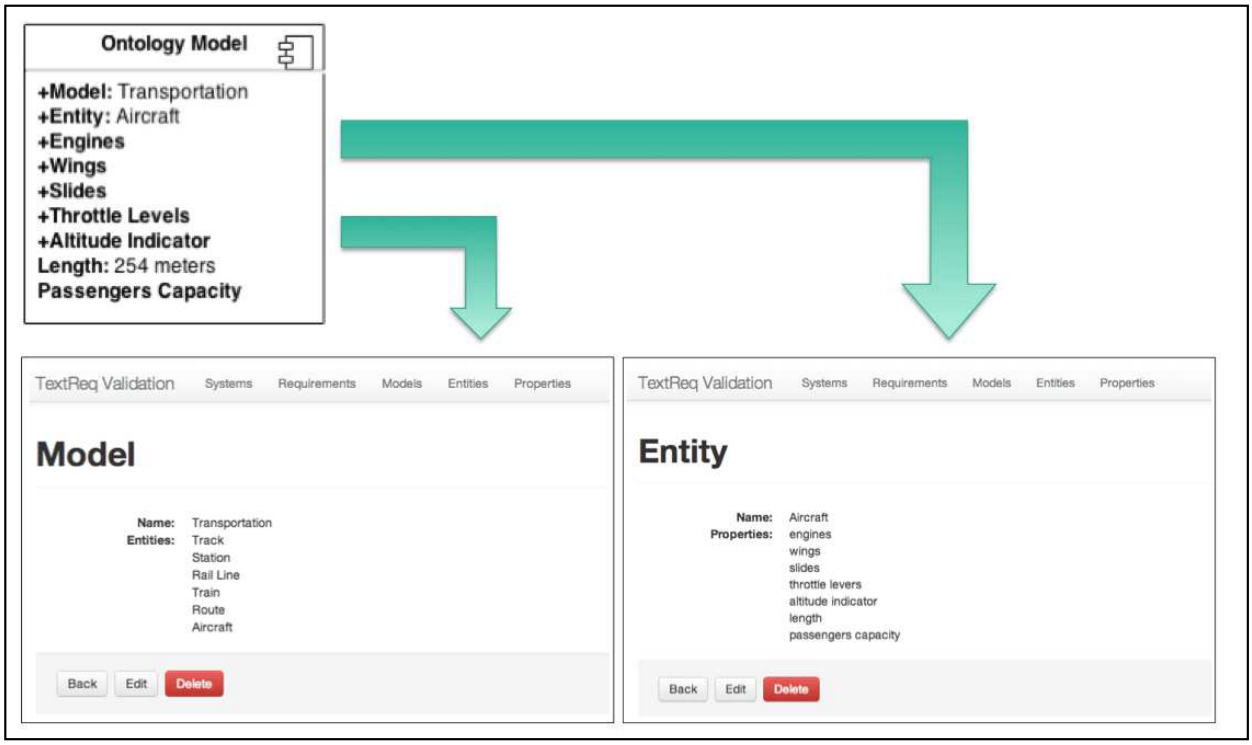


Figure 11. Relationship among aircraft and transportation ontology models, and an aircraft entity model. Top left: Simplified ontology model for an aircraft. Bottom left: Detailed view of the Transportation ontology model. Bottom right: Detailed view for the entity Aircraft.

boilerplate. We have one controller for each part of the model of the application plus an overall "application controller." Each model's controller implements the methods required to handle the client's requests, following a REST (representational, state, transfer) architecture.

The source code for both the web application and the Python scripts are openly hosted in GitHub, in the repository https://github.com/aarellano/textrv.

V. Case Study Problems

We now demonstrate the capabilities of the proposed methodology by working through two case study problems.

Case Study 1: Simple Aircraft Application. We have exercised our ideas in a prototype application, step-by-step development of a simplified aircraft ontology model and a couple of associated textual requirements. The software system requires two inputs: (1) An ontology model that defines what we are designing, and (2) A system defined by its requirements. It is worth noting that while the ontology model and system requirements are unrealistically simple, and deal with only a handful of properties, a key benefit is that we can visualize them easily.

The upper left-hand side of Figure 11 shows the aircraft model we are going to use. We manage a flattened (i.e., tabular) version of a simplified aircraft ontology. This simple ontology suggests usage of a hierarchical model structure, with aircraft properties also being represented by their own specialized ontology models. For instance, an ontology model for the *Wings*, which in turn could have more nested models, along with leaf properties like *length*. Second, it makes sense to include a property in the model even if its value is not set. Naturally, this lacks valuable information, but it does give us the knowledge that that particular property is part of the model, so we can check for its presence.

The step-by-step procedure for using *TextReq Validation* begins with input of the ontology model, then its entities, and finally the properties for each entity. The next step is to create a system model and link it to the ontology. We propose a one-to-one association relationship between the system and an ontology, with more complex relationships handled through hierarchical structures in ontologies. This assumption simplifies development because when we are creating a system we only need to refer to one ontology model and one entity.

The system design is specified through *textual system requirements*. To enter them we need a system, a title and a description. For example, Figure 12 shows all the system Requirements for the system *UMDBus 787*. Notice that each requirement has a title and a description, and it belongs to a specific system. The prototype software has views (details not provided here) to highlight connectivity relationships between the requirements, system model (in this case, a simplified model of a UMDBus 787), and various aircraft ontology models.

Figure 13 is a detailed view of the System UMDBus 787. Besides the usual actions to Edit or Delete a resource, it is important to notice that this view has the *Analyze* and *Validate* actions whose purpose is to trigger the information extraction process described in Section III. The output from these actions is shown in Figures 14 and 15, respectively. The analysis and validation actions match the system's properties taken from its ontology model against information provided in the requirements. In this case study example, the main point to note is that the Aircraft ontology has the property slides (see Figures 11 and 13), but slides is not specified in the textual requirements (see Figure 12). As a result, `slides` shows up as an unverified property in Figure 15.

Case Study 2: Framework for Explicit Representation of Multiple Ontologies. In case study 1, a one-to-one association relationship between the system and an ontology was employed, with more complex relationships handled through hierarchical structures in ontologies. These simplifying assumptions are suitable when we simply want to show that such a simple system setup can work. However, as the number of design requirements and system heterogeneity (i.e., multiple disciplines, multiple physics) increases, the only tractable pathway forward is to make the ontology representations explicit, and to model cause-and-effect dependency relationships among domains in design solutions (i.e., having mixtures of hierarchy and network system structures). While each of the participating disciplines may have a preference toward operating their domain as independently as possible from the other disciplines, achieving target levels of performance and correctness of functionality nearly always requires that disciplines coordinate activities at key points in the system operation. These characteristics are found in a wide range of modern aircraft systems, and they make design a lot more difficult than it used to be.

To see how such an implementation might proceed, Figure 16 illustrates systems validation for requirements covering system-level aircraft specification and detailed wheel system specification. Requirements would be organized into system level requirements (for the main aircraft system) and subsystem level requirements (for the wheels, power systems, and so forth). Full satisfaction of the high-level wheel requirements specification is dependent on lower-level details (e.g., diameter, width, material) being provided for the wheel

VI. Discussion

We have yet to fully test the limits of NLP as applied to requirements engineering. The two case studies presented here demonstrate a framework for using NLP in conjunction with domain ontologies in order to verify requirements coverage. There may be other applications of NLP. A framework for verifying requirements coverage while maintaining consistency by using "requirements templates" has been proposed [30]. For this paradigm, all requirements describing a specific capability must be structured according to a predetermined set of templates. Coverage can then be verified by mapping instances of templates in a set of decomposed requirements to an original list of required capabilities. Figure 17 shows a workflow that combines the requirements template framework with our own. Since the requirements follow templates, it is straightforward for NLP to extract high-level information. Capabilities can then be flowed down for decomposition of each systems requirements. An even further extension of this idea is to use NLP while writing requirements in real time. If an ontology

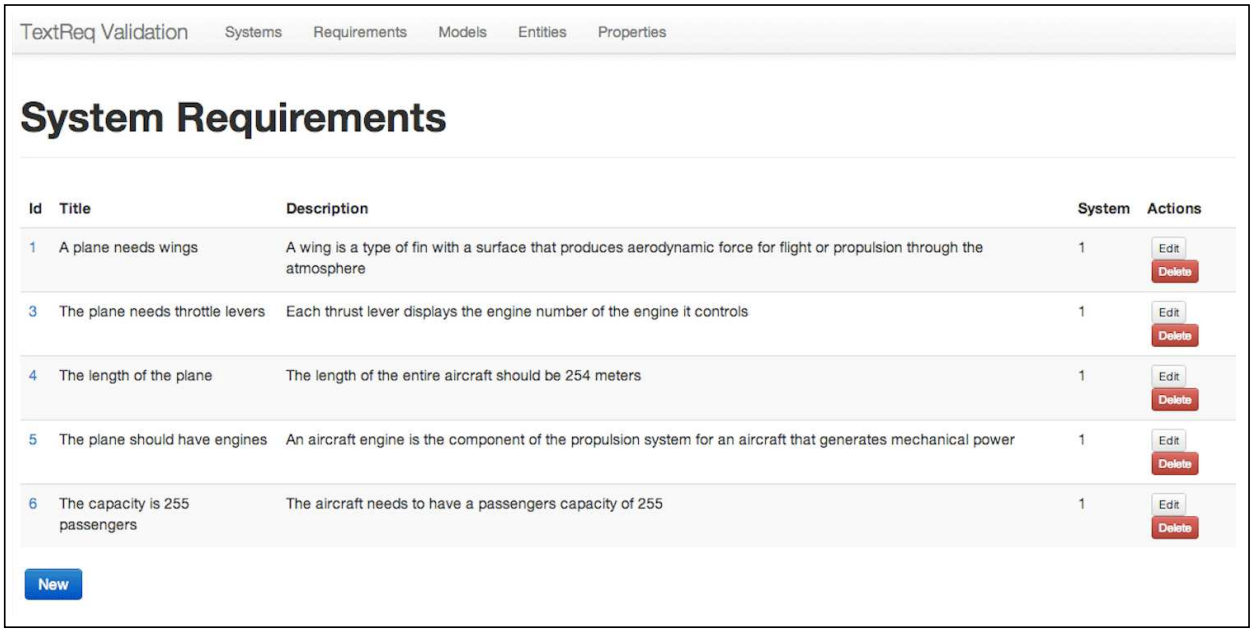


Figure 12. Panel showing all the requirements for the system *UMDBus 787*.

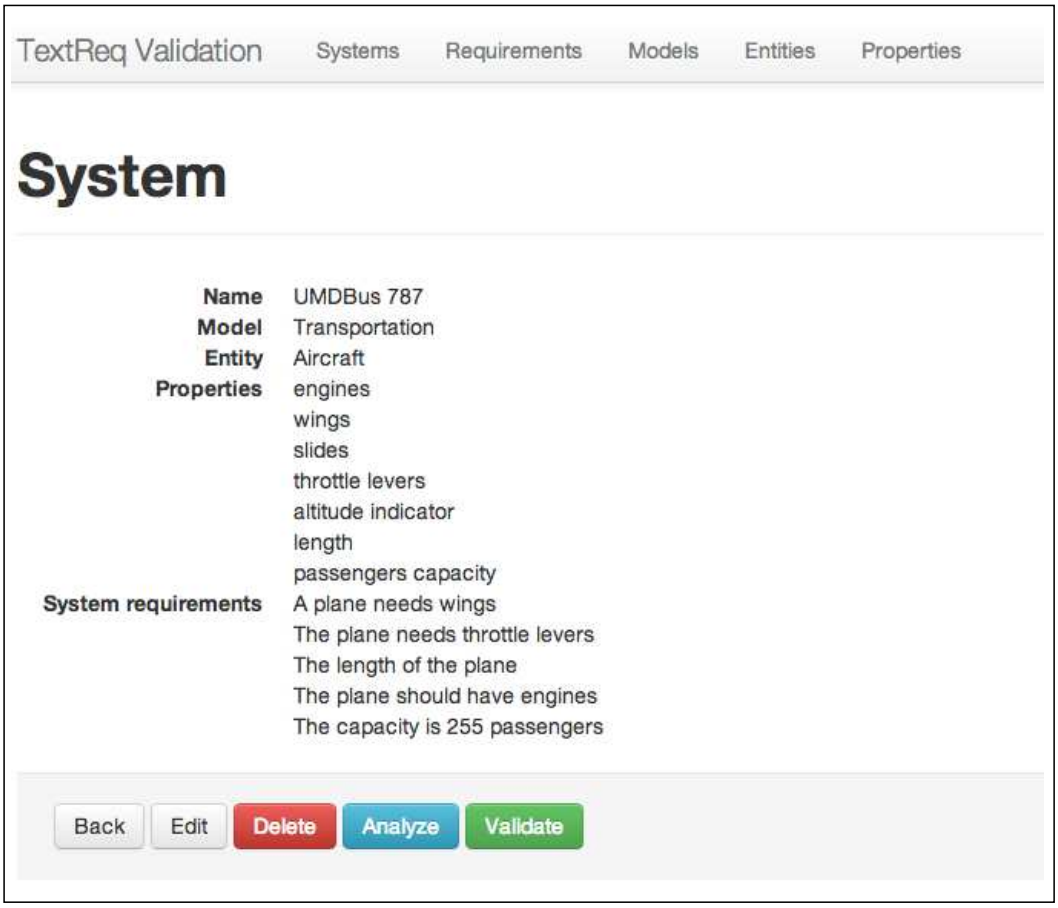


Figure 13. Detailed view for the System *UMDBus 787*.

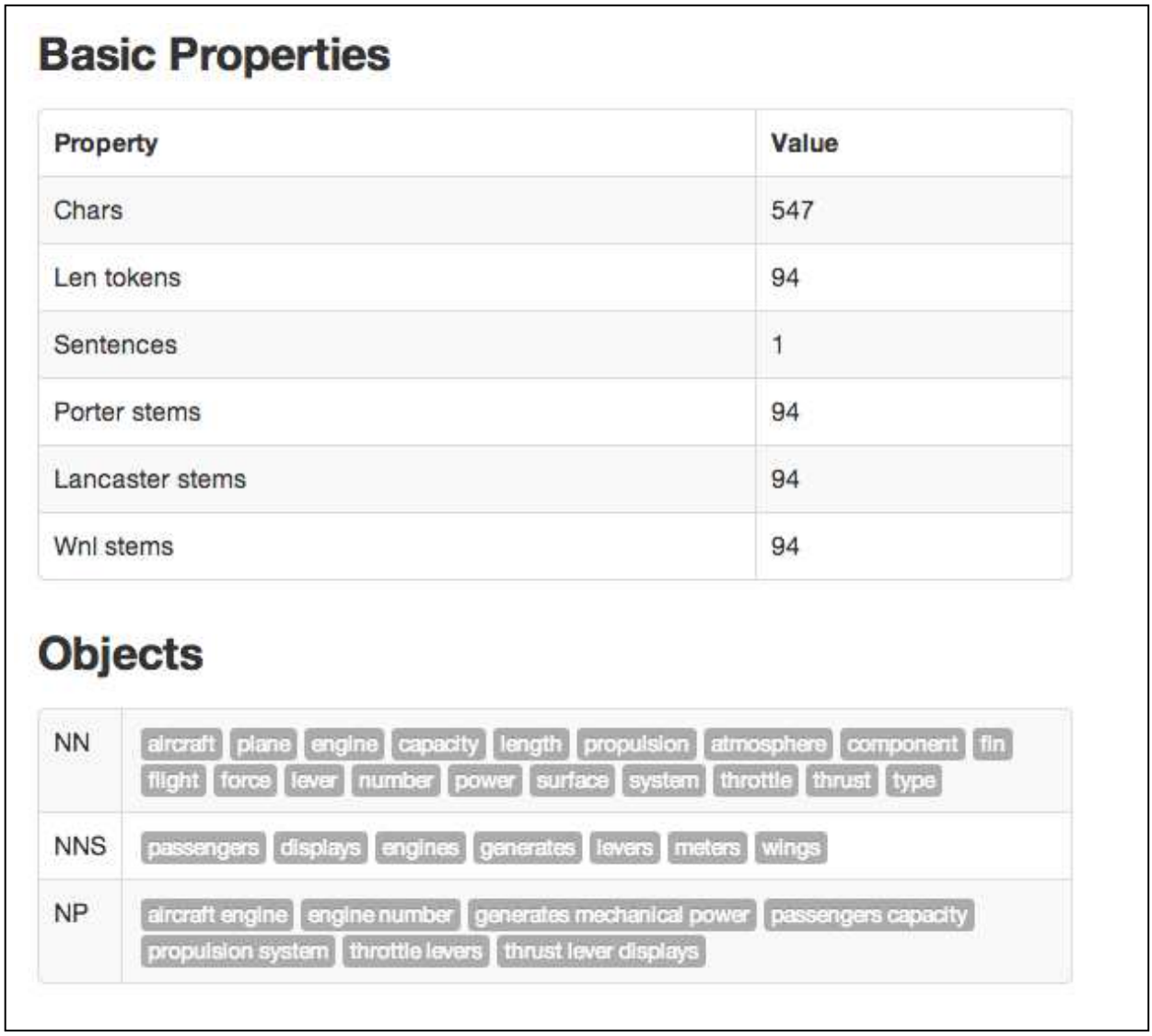


Figure 14. Basic stats from the text, and a list of the entities recognized in it.

Figure 15. This is the final output from the application workflow. It shows what properties are verified (i.e., are present in the system requirements) and which ones are not.

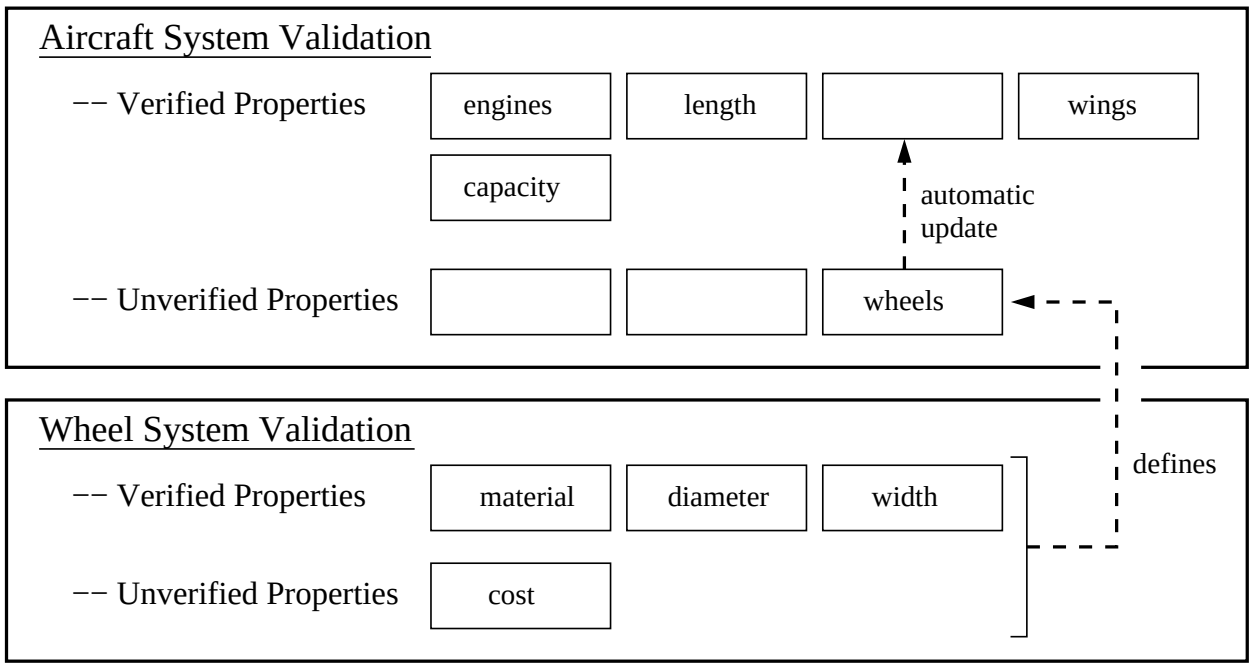


Figure 16. Systems validation for requirements covering system-level aircraft specification and detailed wheel system specification.

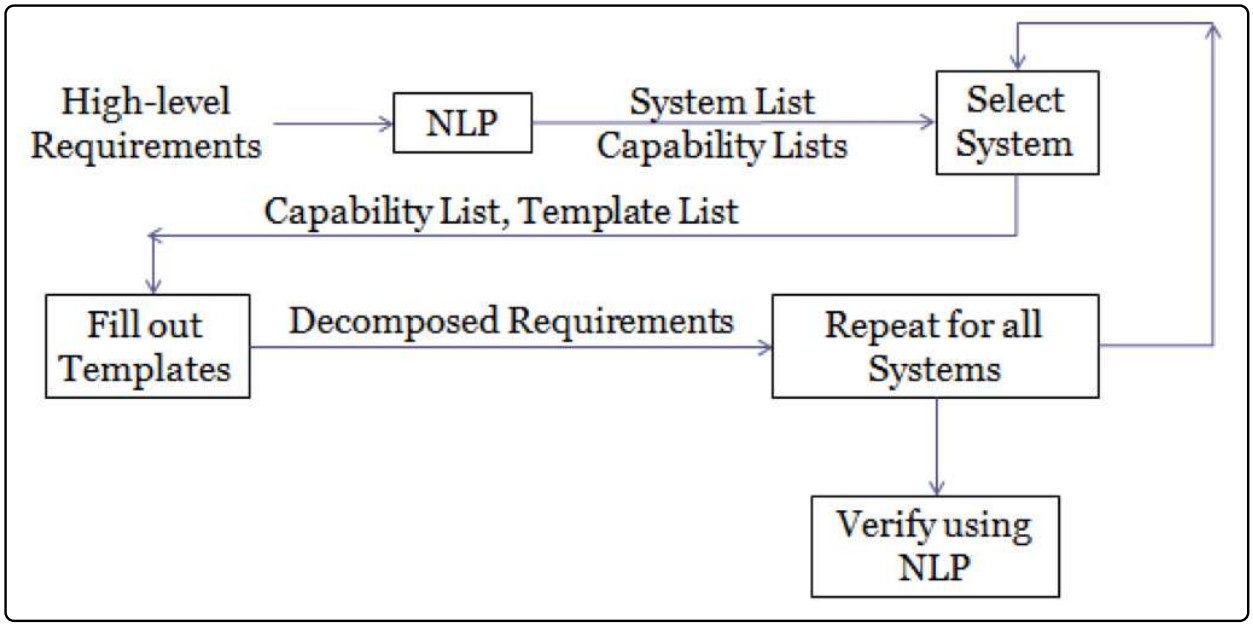


Figure 17. Framework for NLP of textual requirements with templates.

of requirements templates exists, perhaps application-specific NLP could be incorporated into a tool that helps construct and validate requirements as they are written. Some engineers will complain that they are being forced to comply to prescribed standards for writing requirements. Perhaps they will have difficulty in expressing their intent? Our view is that: (1) writing requirements in a manner to satisfy template formats is not much different than being asked to spell check your writing, and (2) the existence of such templates may drastically increase the opportunity for automated transformation of textual requirements into semi-formal models (see Figure 1).

VII. Conclusions and Future Work

When a system is prescribed by a large number of (non formal) textual requirements, the combination of previously defined ontology models and natural language processing techniques can play an important role in validating and verifying a system design. Future work will include formal analysis on the attributes of each property coupled with use of NLP to extract ontology information from a set of requirements. Rigorous automatic domain ontology extraction requires a deep understanding of input text, and so it is fair to say that these techniques are still relatively immature. As noted in Section VI, a second opportunity is the use of NLP techniques in conjunction with a repository of acceptable "template sentence structures" for writing requirements [30]. Finally, there is a strong need for techniques that use the different levels of detail in the requirements specification, and bring ontology models from different domains to validate that the requirements belongs to the supposed domain. This challenge belongs to the NLP area of *classification*.

VIII. Acknowledgments

Financial support to the first author was received from the Fulbright Foundation.

References

[1] A. Arellano, E. Carney, and M.A. Austin, "Natural Language Processing of Textual Requirements," The Tenth International Conference on Systems (ICONS 2015), Barcelona, Spain, April 19–24, 2015, pp. 93–97.

[2] M.A. Austin and J. Baras, "An Introduction to Information-Centric Systems Engineering," Tutorial F06, INCOSE, Toulouse, France, June, 2004.

[3] V. Ambriola and V. Gervasi, "Processing Natural Language Requirements," Proceedings 12th IEEE International Conference Automated Software Engineering, IEEE Computer Society, 1997, pp. 36–45.

[4] C. Rolland and C. Proix, "A Natural Language Approach for Requirements Engineering," Advanced Information Systems Engineering, Springer, 1992, pp. 257–277.

[5] K. Ryan, "The Role of Natural Language in Requirements Engineering," Proceedings of the IEEE International Symposium on Requirements Engineering, IEEE Comput. Soc. Press, 1993, pp. 240–242.

[6] M.A. Austin, V. Mayank, and N. Shmunis, "PaladinRM: Graph-Based Visualization of Requirements Organized for Team-Based Design," Systems Engineering: The Journal of the International Council on Systems Engineering, Vol. 9, No. 2, May, 2006, pp. 129–145.

[7] M.A. Austin, V. Mayank, and N. Shmunis, "Ontology-Based Validation of Connectivity Relationships in a Home Theater System," 21st International Journal of Intelligent Systems, Vol. 21, No. 10, October, 2006, pp. 1111–1125.

[8] N. Nassar and M.A. Austin, "Model-Based Systems Engineering Design and Trade-Off Analysis with RDF Graphs," 11th Annual Conference on Systems Engineering Research (CSER 2013), Georgia Institute of Technology, Atlanta, GA, March 19–22, 2013.

[9] P. Delgoshaei, M.A. Austin, and D. A. Veronica, "A Semantic Platform Infrastructure for Requirements Traceability and System Assessment," The Ninth International Conference on Systems (ICONS2014), Nice, France, February 23–27, 2014, pp. 215–219.

[10] P. Delgoshaei, M.A. Austin, and A. Pertzborn, "A Semantic Framework for Modeling and Simulation of Cyber-Physical Systems," International Journal On Advances in Systems and Measurements, Vol. 7, No. 3-4, December, 2014, pp. 223–238.

[11] S. Fridenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML*. MK-OMG, 2008.

[12] K. Kageura and B. Umino, "Methods of Automatic Term Recognition: A Review," Terminology, Vol. 3, No. 2, 1996, pp. 259-289.

[13] L.L. Earl, "Experiments in Automatic Extracting and Indexing," Information Storage and Retrieval, Vol. 6, No. 6, 1970, pp. 273–288.

[14] S. Ananiadou, "A Methodology for Automatic Term Recognition," Proceedings of 15th International Conference on Computational Linguistics (COLING94), 1994, pp. 1034-1038.

[15] K Frantzi, S. Ananiadou, and H. Mima, "Automatic Recognition of Multi-Word Terms: The C-Value/NC-Value Method," International Journal on Digital Libraries, Vol. 3, No. 2., 2000, pp. 115-130.

[16] D. Fedorenko, N. Astrakhantsev, and D. Turdakov, "Automatic Recognition of Domain-Specific Terms: An Experimental Evaluation," Proceedings of SYRCoDIS 2013, 2013, pp. 15-23.

[17] A. Judea, H. Schutze, and S. Bruegmann, "Unsupervised Training Set Generation for Automatic Acquisition of Technical Terminology in Patents," Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland: Dublin City University and Association for Computational Linguistics, 2014, pp. 290-300.

[18] L. Kozakov, Y. Park, T. Fin, et al., "Glossary Extraction and Utilization in the Information Search and Delivery System for IBM Technical Support," IBM Systems Journal, Vol. 43, No. 3, 2004, pp. 546-563.

[19] NLTK Project, "Natural Language Toolkit NLTK 3.0 Documentation," See http://www.nltk.org/ (Accessed: December 1, 2015).

[20] C. Manning and H. Schuetze, "Foundations of Statistical Natural Language Processing," The MIT Press, 2012.

[21] M.F. Porter, "An Algorithm for Suffix Stripping," Program: Electronic Library and Information Systems, MCB UP Ltd, Vol. 14, No. 3, 1980, pp. 130–137.

[22] C.D. Paice, "Another Stemmer," ACM SIGIR Forum, ACM, See: http://dl.acm.org/citation.cfm?id=101306.101310, Vol. 24, No. 3, November, 1990, pp. 56–61.

[23] Princeton University, "About WordNet - WordNet - About WordNet," See https://wordnet.princeton.edu/ (Accessed: December 1, 2015).

[24] M. Haspelmath, "Word Classes and Parts of Speech," See http://philpapers.org/rec/HASWCA, (Accessed: December 1, 2015).

[25] S. Bird, E. Klein, and E. Loper, "Natural Language Processing with Python," O'Reilly Media, Inc., 2009.

[26] University of Pennsylvania, Penn Treebank Project, See http://www.cis.upenn.edu/ treebank/ (Accessed, December 1, 2015).

[27] B. Santorini, "Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision)," Technical Reports (CIS), http://repository.upenn.edu/cis_reports/570, 1990.

[28] Ruby on Rails. See http://rubyonrails.org/ (Accessed, December 1, 2015).

[29] Bootstrap. See http://getbootstrap.com/2.3.2/ (Accessed, December 1, 2015).

[30] E. Hull, K. Jackson and J. Dick, Requirements Engineering, Springer, 2002.

Safe Traffic Intersections: Metrics, Tubes, and Prototype Simulation for Solving the Dilemma Zone Problem

Leonard Petnga
Department of Civil and Environmental Engineering
University of Maryland, College Park, MD 20742, USA
Email: lpetnga@umd.edu

Mark A. Austin
Department of Civil and Environmental Engineering
and Institute for Systems Research
University of Maryland, College Park, MD 20742, USA
Email: austin@isr.umd.edu

***Abstract*—Our research is concerned with the modeling and design of cyber-physical transportation systems (CPTS), a class of applications where the tight integration of software with physical processes allows for the automated management of system functionality, superior levels of performance, and safety assurance. Part of the safety assurance problem is prevention of deadly accidents at traffic intersections and, in particular, finding ways for vehicles to traverse the dilemma zone (DZ), an area at a traffic intersection where drivers are indecisive on whether to stop or cross at the onset of a yellow light. State-of-the-art approaches to the dilemma zone problem treat the cars and stoplights separately, with the problem formulation being expressed exclusively in either spatial or temporal terms. In this paper, we formulate a methodology that accounts for two-way interactions between the cars and stoplights, and propose quantitative metrics and three-dimensional dilemma tubes as a means for compactly describing sets of conditions for which the vehicle-light system will be in an unsafe state. The proposed metrics enable simple and actionable decision capabilities to deal with unsafe configurations of the system. The second purpose of this paper is to describe a pathway toward the integration of dilemma metrics and dilemma tubes with an ontological framework. The associated platform infrastructure supports algorithmic implementations of simulation and reasoning for resolving unsafe configurations of CPTS, such as those created by the DZ problem.**

Keywords-Dilemma Zone; Metrics; Cyber-Physical Transportation Systems; Artificial Intelligence; Safety.

I. Introduction

This paper describes the development and simulation of metrics for safety analysis of cyber-physical transportation systems (CPTS). It builds upon our previous work [1] on tubes and metrics for solving the dilemma zone problem at traffic intersection. During the past three decades, transportation systems have been transformed by remarkable advances in sensing, computing, communications, and material technologies. The depth and breadth of these advances can be found in superior levels of automobile performance and new approaches to automobile design that are becoming increasing reliant on sensing, electronics, and computing to achieve target levels of functionality, performance and cost. By 2016, as much as 40% of an automobile's value will be embedded software and control related components [2][3]. Looking ahead, even greater levels of automation will be needed for self-driving cars [4][5].

While consumers applaud the benefits of these advances and the products they enable, engineers are faced with a multitude of challenges that are hindering the system-level development of cyber-physical transportation systems (CPTS). These challenges include: (1) the integration of cyber-physical systems (CPS) technologies into existing infrastructure, (2) the realization of "zero fatality" transportation systems, and (3) the development of formal models and credible, actionable performance and safety metrics [6]. To this end, metrics for system safety are needed to: (1) evaluate the operation and control of transportation systems in a consistent and systematic way, (2) identify, measure, and predict dynamic interactions among system components, (3) set standards that serve as measure of effectiveness (MoEs) and can guide model-based systems engineering (MBSE) efforts. And yet, despite these advances, accidents at traffic intersections claim around 2,000 lives annually within the US alone [7]. A key component of this safety problem is the dilemma zone (DZ), which is an area at a traffic intersection where drivers are indecisive on whether to stop or cross at the onset of a yellow light.

In this project, we consider the interplay among the key elements of transportation systems at traffic intersections, and the consequences of their interactions on overall traffic system level safety. This paper focuses on one aspect of the dilemma zone problem, namely, development of metrics to capture the essence of these interactions, and support the characterization of the problem and its representation using three-dimensional dilemma tubes. Section II is a review of existing approaches to the dilemma zone problem and their limitations with regard to the current trend toward CPTS. Section III introduces the new dilemma zone metrics and their tubular representation. Sections IV and V describe the system architecture and simulation prototype, respectively. Metrics for the assessment of safety analysis are introduced in Section VI. The paper concludes with discussion, conclusions and future work.

II. Dilemma Zone Problem and Cyber-Physicality of Traffic Systems

Dilemma Zone: Definition and Existing Solution Approaches. Also called the twilight zone, Amber signal or decision zone, the dilemma zone is the area at a traffic intersection where drivers are indecisive on whether to stop or cross at the onset of a yellow light. Research [8] indicates that under such circumstances only 90% of drivers will "play it safe" and decide to stop. Consequently, the behavior of users

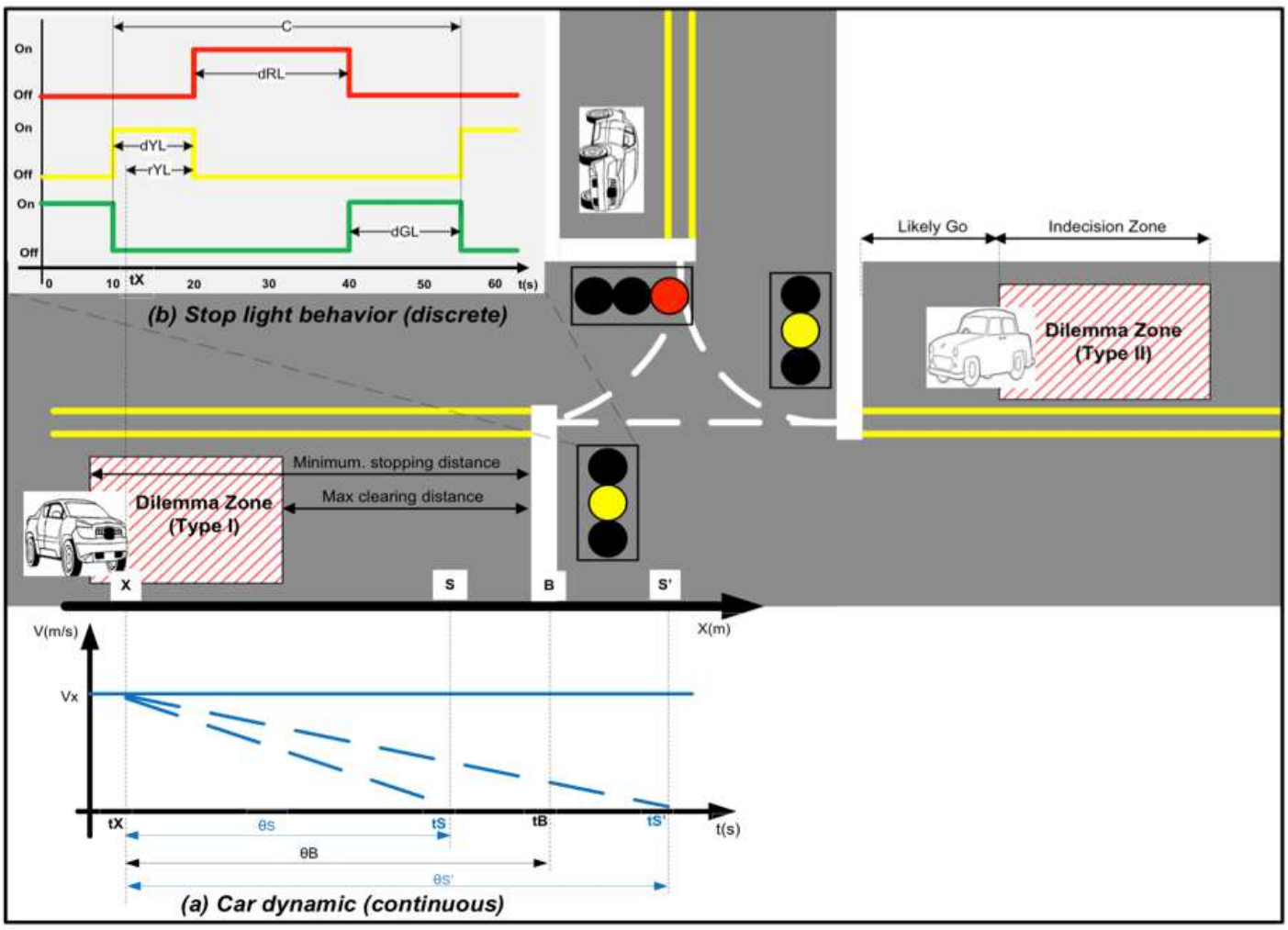


Figure 1. Schematic of spatial and temporal concerns in the dilemma zone problem. Traffic lights have discrete state behavior versus time. Here, C is the total cycle time for the lights. Variables dGL, dYL and dRL represent the duration of the green, yellow, and red lights, respectively. Variables rYL is the time remaining for the yellow light. Vehicles have dynamic behavior that varies continuously with time. Here, θS is the time it takes the vehicle to fully stop before the stopline, θB is the time to reach the intersection while traveling at speed V_x, and $\theta S^{'}$ is the time it takes the vehicle to fully stop after the stopline.

in "twilight zones" is responsible for hundreds of lives lost and billions of dollars in damages at stop light intersections in the United States [7].

From an analysis standpoint (see Figure 1), scholars distinguish two types of dilemma zone that differ by the perspective adopted on the problem. Type I dilemma zone formulations place the "physics of the vehicle" at the center of the problem formulation and are concerned with the difference between the distance from the stop line to the nearest vehicle that can stop safely (i.e., minimum stopping distance) and the distance from the stop line of the farthest vehicle that can cross the intersection at the onset of the yellow light (i.e., maximum clearing distance) [9][10]. Therefore, the physical parameters of the situation (e.g., car speed, road and car conditions, and so forth) are the key determinant of whether the car will be able to safely cross the intersection or stop prior to the stop line. Type II dilemma zone formulations (see the right-hand side of Figure 1) are defined with regard to the driver's behavior and decision-making as the vehicle approaches the intersection and the onset of a yellow light. The boundaries of this type of DZ are also sometimes measured with a temporal tag (i.e., representing the duration to the stop line) added to the probabilistic estimate [11]. In this work, we will adopt the Type I definition of the dilemma zone.

Past research has focused on finding ways to mitigate, or eliminate, DZs using mostly a pure traffic control engineering view of the problem. These efforts have resulted in signal timing adjustment solutions that ignore or cannot properly account for the physics of vehicles or driver's behaviors [12][13][14]. In order to deal with uncertainties, other scholars have used stochastic approaches such as fuzzy set [9] and Markov chains [10]. For all of these traditional techniques, the baseline of the solution can be either reduced (explicitly or not) to a space- or temporal-based dilemma zone, but not both.

Autonomous Cars and Intelligent Traffic Control Systems. Recent work [15][16] illustrates the switch of researchers' interest toward investigating solutions to the DZ problem

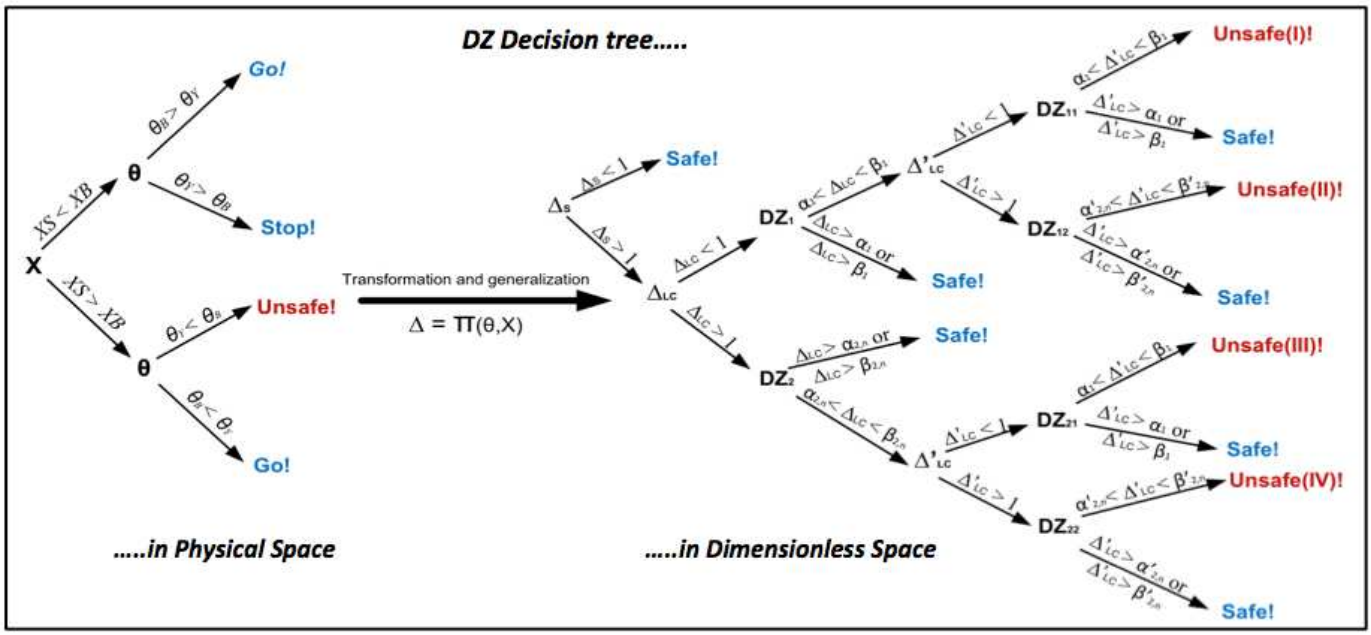


Figure 2. Framework for decision-making. Left: decision-making in the physical space. Right: decision-making in the dimensionless space.

that incorporate both the car physics and light timing, while also providing a pathway forward for vehicle-to-infrastructure (V2I) interactions and integration. These solutions will soon become a reality, in part, because of an increased use of artificial intelligence in automating the command and operation of both cars and traffic signals. For automobiles, many aspects of autonomy – from braking to cruise control and driving functions – are in advanced stages of experimentation. Finding ways to put smartness into vehicles has contributed to reduced fatalities on highways mostly in the developed world. The enhancement of traffic signal controls with artificial intelligence is an idea whose time has arrived – indeed, we now have the capability to determine the position, speed and direction of vehicles, and adjust light cycling times in a coordinated way to make the intersection crossing more efficient. Researchers have been developing and testing various technologies with mixed results [17][18][19]. As a case in point, a pilot study conducted by Carnegie Mellon University, reports a 40% reduction of intersection waiting times, an estimated 26% decrease in travel time, and a projected 21% decrease of CO_2 emissions [19]. Tapping into the full potential of these intelligence capabilities is hindered by practical constraints that include: (1) most vehicles cannot currently communicate with traffic light controllers, and (2) autonomous vehicles still struggle in operating safely in adverse weather conditions (heavy rain, snow covered roads, etc.) and changing environment (temporary traffic signals, potholes, human behaviors, etc.). In this paper, we assume that these problems will be resolved by ongoing research activities.

Toward Cyber-Physical Traffic Management Systems. Real-time situational awareness (e.g., traffic, location, speed) and decision, combined with vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications and control are valid and effective pathways for a solution to both congestion and safety at intersections. As such, we fully adopt a CPS view of the traffic system with regard to the DZ problem. The value of this perspective has already been demonstrated by Petnga and Austin [20]. Autonomous vehicles (i.e., the physical system) interact with the light (i.e., the cyber system) with the objective of maximizing traffic throughput, while ensuring vehicle crossings are safe at the intersection. Enhanced performance and safety at the intersection have been proven possible, thanks to the critical role of temporal semantics in improving system level decision-making. Also, when bi-directional connections between the vehicle and light are possible, new relationships can be established to characterize their tight coupling – this, in turn, enables the various computers in the CPTS to exchange information, reason, and make informed decisions. These capabilities become safety-critical for situations – hopefully, rare situations – where behavior/physics of a vehicle is such that they can neither stop, nor proceed, without entering and occupying the intersection while the traffic light is red. Therefore, the development of metrics for the DZ problem will greatly benefit from and enrich the CPTS perspective.

III. Metrics for Characterizing the Dilemma Zone Problem

Safety Requirements to Decision Trees and Dilemma Metrics. The core safety requirement for the car-light system that must prevail at all times is as follows: "No vehicle is allowed to cross the intersection when the light is red." This is a hard constraint whose violation is the driving force behind accidents at intersections.

Understanding the mechanisms by which system-level safety is achieved or violated is critical to addressing the DZ challenge. This task is complicated by the need to work with mixtures of continuous (vehicle) and discrete (traffic light) behavior as illustrated in Figure 1 (a) and (b). We propose that decision trees are a suitable framework for representing

the multitude of decision-making pathways. Some of these pathways will correspond to behaviors that are safe. Others will be unsafe and need to be avoided. The tree shown on the left-hand side of Figure 2 shows the decision tree of the autonomous car - in the physical space - when it knows the traffic lights critical parameters at the time the decision is made. Petnga and Austin [20][21] have shown that the probability of the car making the right decision is higher when it knows before hands the following: (1) Duration Θ_Y of the yellow light before it turns red; (2) Vehicle stopping distance XS, and (3) Travel duration, Θ_B, or distance, XB, to the traffic light.

Moving forward requires a deep understanding of the interrelationships between cross-cutting system parameters from the various domains (car, light, time, space) involved at meta level. Also, the ability of the system to efficiently reason about unsafe situations and propose a satisfactory way out is critical. We argue that this complexity can be kept in check by casting the problem in dimensionless terms and setting up a transformation,

$$\Delta = \Pi(\Theta, X), \tag{1}$$

of the initial decision tree from the physical space to a dimensionless space. Expressing the system decision tree in dimensionless space as a result of the transformation Π necessitates the definition of intermediary variables and parameters.

We begin by noting that the car will not always catch the onset of the yellow light; thus, what is really relevant for efficient decision-making here is the time left before the stop light turns red. Using the remaining duration of the yellow light r_{YL}, its full duration d_{YL} and the ones of the green and red lights ie d_{GL} and d_{RL}, we define the duration of a stop light cycle C, reduced cycle C_{YL} and cycle index k as follows:

$$C = d_{YL} + d_{RL} + d_{GL} \tag{2}$$
$$C_{YL} = r_{YL} + d_{RL} + d_{GL} \tag{3}$$
$$k = \frac{C}{C_{YL}} \tag{4}$$
$$\tag{5}$$

The short (α_1) and full (α_2) yellow light duration as well as the short (β_1) and full (β_2) stop light indexes are defined as follows:

$$\alpha_1 = \frac{r_{YL}}{C_{YL}} \tag{6}$$
$$\alpha_2 = \frac{d_{YL}}{C_{YL}} \tag{7}$$
$$\beta_1 = \frac{r_{YL} + d_{RL}}{C_{YL}} \tag{8}$$
$$\beta_2 = \frac{d_{YL} + d_{RL}}{C_{YL}}. \tag{9}$$

We add to the aforementioned physical variables the stopping duration $\Theta_B^{'}$ of the car – should it decide to stop – and define the **car stopping distance metric** Δ_S, the **light-car crossing time metric** Δ_{LC} and the **light-car stopping time metric** $\Delta_{LC}^{'}$ as follows:

$$\Delta_S = \frac{XS}{XB} \tag{10}$$
$$\Delta_{LC} = \frac{\Theta_B}{C_{YL}} \tag{11}$$
$$\Delta_{LC}^{'} = \frac{\Theta_B^{'}}{C_{YL}}. \tag{12}$$

All these metrics are dimensionless and serve as the key decision points of the dimensionless decision tree shown on the right-hand side of Figure 2.

Navigating the Decision Tree. Navigation of the decision tree is facilitated by the equation pair:

$$n = E\left(\frac{\Delta_{LC} - 1}{k}\right) \tag{13}$$
$$n^{'} = E\left(\frac{\Delta_{LC}^{'} - 1}{k}\right) \tag{14}$$

We employ the integer part function E to define indexes n and $n^{'}$. Equations (13) and (14) simplify the definition of α and β indexes when $\Delta_{LC} > 1$ or $\Delta_{LC}^{'} > 1$ as follows.

$$\alpha_{2,n} = k * \alpha_2 + k * n + 1 \tag{15}$$
$$\beta_{2,n} = k * \beta_2 + k * n + 1 \tag{16}$$
$$\alpha_{2,n}^{'} = k * \alpha_2 + k * n^{'} + 1 \tag{17}$$
$$\beta_{2,n}^{'} = k * \beta_2 + k * n^{'} + 1 \tag{18}$$

Along with equations (6) through (9), the values of α and β (see equations (15) through (18)) are necessary and sufficient to constrain the dimensionless metrics Δ_S, Δ_{LC} and $\Delta_{LC}^{'}$ and render a complete view of all possible outcomes of the decision tree in a dimensionless space Δ. From the right-hand side of Figure 2, we can see that there are four possible configurations of the system for which it is unsafe.

From Dilemma Metrics to Dilemma Tubes. Each of the decision tree pathways on the right-hand side of Figure 2 that leads to an unsafe system state can be represented as a "dilemma tube" in the Δ space, as shown in Figure 3. For instance, equations (6), (8), and (10) through (12) provide the foundational elements for defining Tube I. The boundaries of each of the four tubes (i.e., I, II, III and IV) correspond to the above-mentioned parameters, with the maximum value of Δ_S i.e., Δ_{Smax} corresponding to the maximum value of all the Δ_S values in the system. Physically, this is determined by the physics of the family of vehicles crossing the intersection and the configuration of the traffic intersection as captured by equation (10). If, at any point in time, the system is projected to enter an unsafe state, this situation will be materialized as a point coordinate $P_\Delta(\Delta_S, \Delta_{LC}, \Delta_{LC}^{'})$ that is located inside a particular tube. The physical interpretation of such

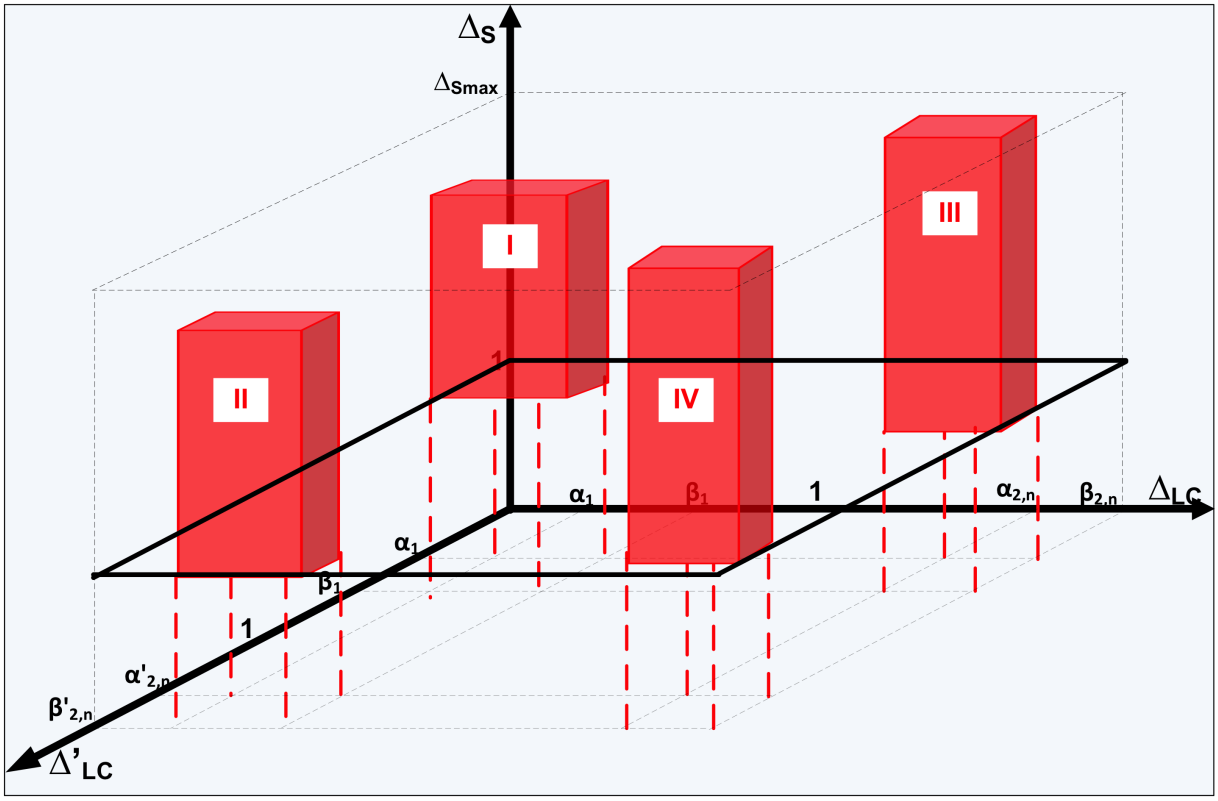


Figure 3. Dilemma tubes in the dimensionless (Δ) space.

phenomenon is that the autonomous car does not have a good decision option, and will need external help to safely cross the intersection.

Scenarios that lead to unsafe system configurations (e.g., see the right-hand side of Figure 2) will follow branches of the decision tree that terminate with an "Unsafe" system state. While the actual behaviors might not evolve along the pathways presented in the decision tree, the end result will invariably be the same (i.e., the system will be projected to enter an unsafe state). In practice, simulation and safety calculations can be done concurrently and the location of the resulting point coordinate relative to any of the four dilemma tube types easily determined. A final important point to note is that since each of the tubes is mutually exclusive, a vehicle can only be in one of the four dilemma tubes at a time, or in any location in the remaining part of the Δ space, i.e., a safe region.

Knowing in which tube the unsafe state has been materialized is critical in determining the appropriate course of action to prevent the occurrence of an accident.

IV. System Architecture

This section introduces a Java-based software system infrastructure that adheres to the CPTS perspective and supports the tube framework described in Sections II and III. As illustrated in Figure 4, the system architecture contains workspaces for traffic intersection simulation. The main modules of the infrastructure are as follows:

1. Component Modeling. The component modeling module plays a central role in the system simulation. Physical entity models are organized into static and dynamic components, as shown in the mid-section of Figure 4. Examples of the former include the traffic intersection (i.e., the spatial boundary), traffic lights, and their associated sensors. Their key attributes are not expected to change over time such as the stoplight durations d_{YL}, d_{RL} and d_{GL} for the yellow, red and green for each cycle. The remaining duration of the yellow light (r_{YL}) is a key attribute of interest for our study that does decrease with time. As such, the component modeling module needs a clock to account for the elapsed time. In our formulation, sensors play a key role in determining the location (X) and velocity (v) of a vehicle as a function of time. With X and v in place, vehicle accelerations can be computed from the underlying equations of motion. Also, the vehicle braking force (F_b) is subject to change over time; thus, it is a variable of the system.

2. Tube Modeling and Metrics Computation Support. DZ tubes are modeled as software entities because they are not physical entities. In order to properly account for the multiple facets of tubes in this framework, and provide flexibility in the architecture, we propose that tube models serve as a data repository platform and bridge between the computation and the integration modules (see the dashed boxes and connecting arrows in Figure 4).

The interface for the data repository platform distinguishes *base tubes* (not visualized) from *dilemma tubes*. The former

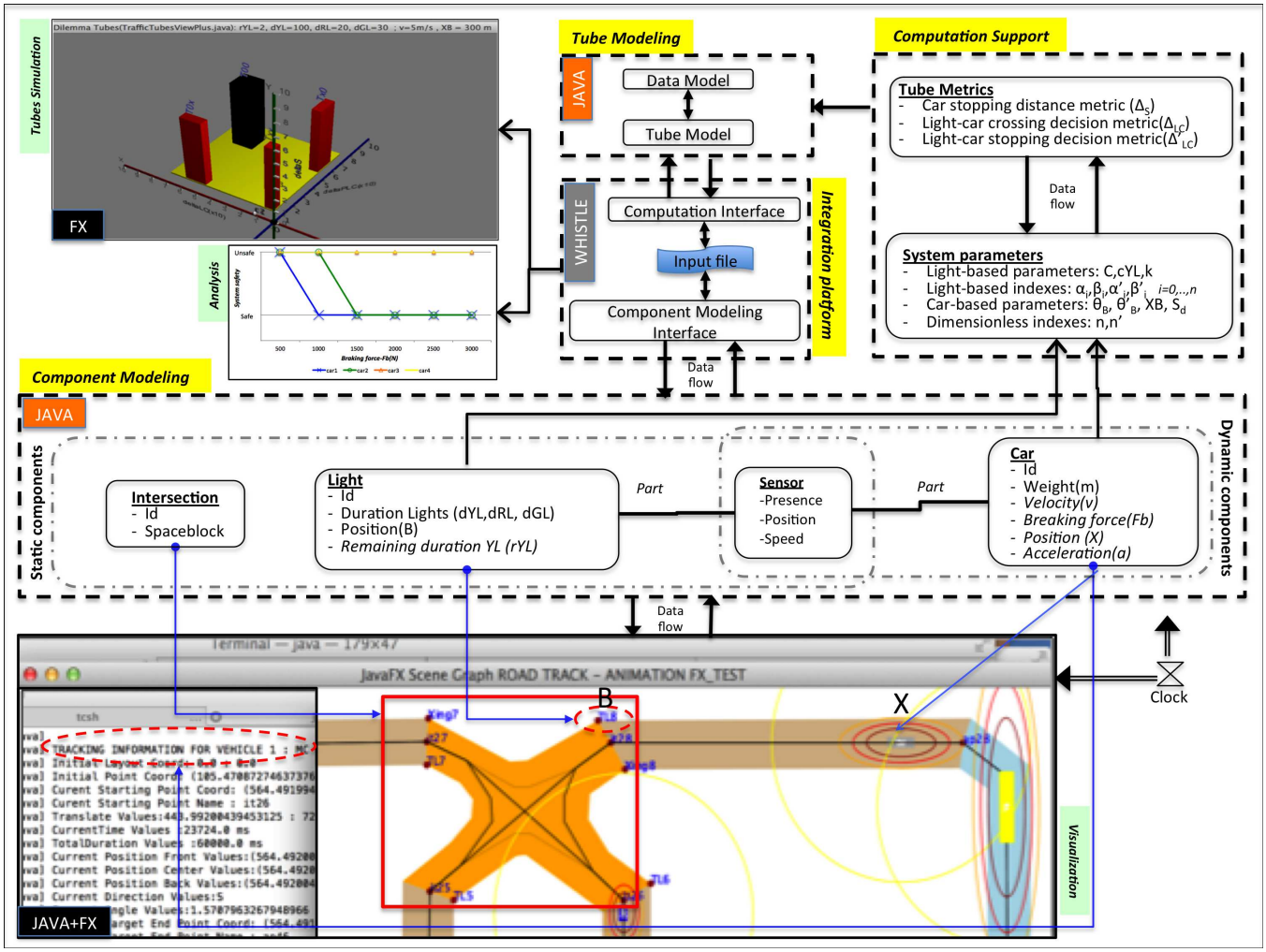


Figure 4. Dilemma tubes simulation system architecture.

store the basic initial configuration of the stop light, and information that will be used to create the latter (i.e., dilemma tubes). Dilemma tubes of various types allow for the representation of unsafe system states as defined by the car stopping distance metric Δ_S, the light-car crossing time metric Δ_{LC}, and the light-car stopping time metric Δ'_{LC} and specifications in equations (4) to (18). This separation of concerns provides modularity and flexibility to the architecture, enabling the support for modeling of complex intersections with multiple stop lights on multi-lanes and/or complex intersection configurations (T,Y,X, etc.).

The visualization system interface (not shown) connects with the integration module, thereby allowing for flows of data to/from the visualization display, and in accordance with the adopted GUI technology. In our software prototype (see the top left-hand corner of Figure 4), the display is controlled from the integration module.

On the interface with the computation support module, a *traffic tube* model is created as an extension of a more basic tube model. It is the ultimate data structure of the tube as it links predefined and computed tubes variables. The initial traffic tube is linked to the base tube, and dilemma tubes are created from updates of corresponding traffic tubes for various values of r_{YL}. The number of dilemma tubes to be visualized is computed by the system based on values of n and n' as defined by equations (13) and (14).

The computation support module enables the correct calculation of the various metrics and variables needed to efficiently characterize the dilemma zone using the tube framework. It receives input data from both the component and the tube modules, processes computation request following formulae in equations (2) thru (18). We distinguish *system parameters* from the three *tube metrics* Δ_S, Δ_{LC}, Δ'_{LC} introduced above. The former are computed car, light or dimension parameters and indexes that will contribute in the computation of the latter. Dimensionless indexes are parameters as they are, by definition, dependent on Δ_{LC} and Δ'_{LC}. Most of these parameters are defined as attributes of the traffic tube model thus, the results are stored as per the specification of that data structure.

3. System Integration. Reaping the benefits of the system architecture requires bringing together its various modules and pieces in an organized but systematic way. Thus, we need a way to assemble system models for the purpose of the various

TABLE I. Summary of simulation parameters.

Element	Variable	Unit	Min	Max	Set value	Predefined parameters
Car	XB	m	10	60	30	m_1=1,500 kg, m_2=2,800 kg,
	F_b	N	3000	8000	5000	m_3=16,500 kg,
	v	m/s	5	30	10	m_4=24,000 kg
Light	r_{YL}	s	0	5	2	d_{RL}=20s
	d_{YL}	s	3	17	5	d_{GL} = 30s

analysis needs. We solve this problem with Whistle [22][23], a tiny scripting language where physical units are deeply embedded within the basic data types, matrices, branching and looping constructs, and method interfaces to external object-oriented software packages. Whistle is designed for rapid, high-level solutions to software problems, ease of use, and flexibility in gluing application components together. Currently, computational support is added enabling Whistle to handle input and output of model data from/to files in various formats (XML, Open Street Map (OSM), Java, etc.). Therefore, an input file (containing any Whistle-compliant program) is an integral and central part of this module. It provides access to other system modules and needed functionality via interfaces encoded as scripts. Also, the sequencing and timing in the execution of the commands is encoded in the program, giving the analyst/modeler the control of the execution of the simulation.

V. Simulation Prototype

We describe in this section an implementation of the framework for a scenario where the system configuration leads to a system state inside Tube I, as shown in Figure 3. The implementation consists of step-by-step assembly of a (typical) dilemma zone scenario, simulation, and analysis of the results. It is subject to three simplifying assumptions: (A1) the air resistance is negligible, (A2) there is a two-way, delay-free communication between the light and the autonomous car, and (A3) computation and reaction times are negligible.

1. Step-by-Step Assembly of a Real-World Scenario. The step-by-step details are as follows:

(i) A traffic system controller of a smart traffic system computes and stores in real-time each stoplight indexes (C, C_{YL}, k, α_i, β_i, i=1,2) based on its corresponding parameters (r_{YL}, d_{GL}, d_{YL}, d_{RL}) using equations (2) through (12).

(ii) An autonomous car approaching the intersection at speed s is given its distance XB to the stop line in real-time. This information is provided either by its on-board radar coupled with its computer or by the intersection controller. The car itself (autonomous vehicle equipped with camera) notices the onset (or the presence) of the yellow light.

(iii) Based on its current acceleration, speed, road conditions, and maximum applicable braking force, the on-board computer of the car estimates the vehicles stopping distance XS, and computes Δ_S (see equation (10)).

(iv) The computer finds that $\Delta_S > 1$, meaning the car cannot be safely immobilized before the stop line. It then determines the normal travel time θ_B to go through the intersection, i.e., to cover the distance XB, should it decides to go at speed s.

(v) The car requests and obtains from the traffic controller the values of α_i, β_i, i=1,2 and the length of the reduced cycle C_{YL}. It then computes the light-car crossing metric Δ_{LC} using equation (11).

(vi) The on-board computer finds that $\alpha_1 < \Delta_{LC} < \beta_1$. At this point, the only way for the car to avoid violating the safety requirement (i.e., never cross the stop line when the light is red) is to hope that while braking, it will cross the stop line when the line is still yellow.

(vii) Using equation (12), the car determines the travel time $\theta_B^{'}$ to cover the distance XB while stopping. Then, it computes the light-car stopping time metric $\Delta_{LC}^{'}$.

(viii) The on-board computer finds that $\alpha_1 < \Delta_{LC}^{'} < \beta_1$, which translates as the light will be already red when the car crosses the stop line while stopping.

Individual values of the metrics Δ_S, Δ_{LC} and $\Delta_{LC}^{'}$ generate a point coordinate somewhere within the dilemma Tube I, as pictured in Figure 3. The physical interpretation of this system state is that the vehicle does not have a good decision option, and will need a change of course of action or help from the light to safely cross the intersection.

2. Simulation Setup and Coverage. The simulation setup relies extensively on Java and its advanced graphics and media packages JavaFX as supportive technologies to create, test, debug, and deploy a client application. Simulation coverage consists of four cars c_i, $i \in \{1, 2, 3, 4\}$ of different size (sedan, SUV, bus, cargo truck) and a stop light. Vehicles will be distinguished by their weight (m). Vehicle velocity (v), braking force (F_b) and distance to stop light line (XB) are discrete parameters that can be selected within a predefined range by the modeler/analyst. As for the stop light, the duration of the red light (d_{RL}) and green light (d_{GL}) are treated as constants; the duration of the yellow light (d_{YL}) and the corresponding remaining duration (r_{YL}) are discrete variables within predefined range. The range of each parameter is generally distributed around an average value that is used when a fixed value for a specific parameter is needed. Table I summarizes the case vehicles and parameter values employed in this simulation.

3. Simulation Execution and Dilemma Tubes Visualization. Visualization of the dilemma tubes occurs through a processing pipeline that involves the acquisition, storage, processing, flow and restitution of data between the input file and the visualization platform. For the execution of a scenario involving one car and one stop light, the following steps will be completed:

(1) A user creates an input file containing an execution/simulation program in a Whistle-compliant format. In this application

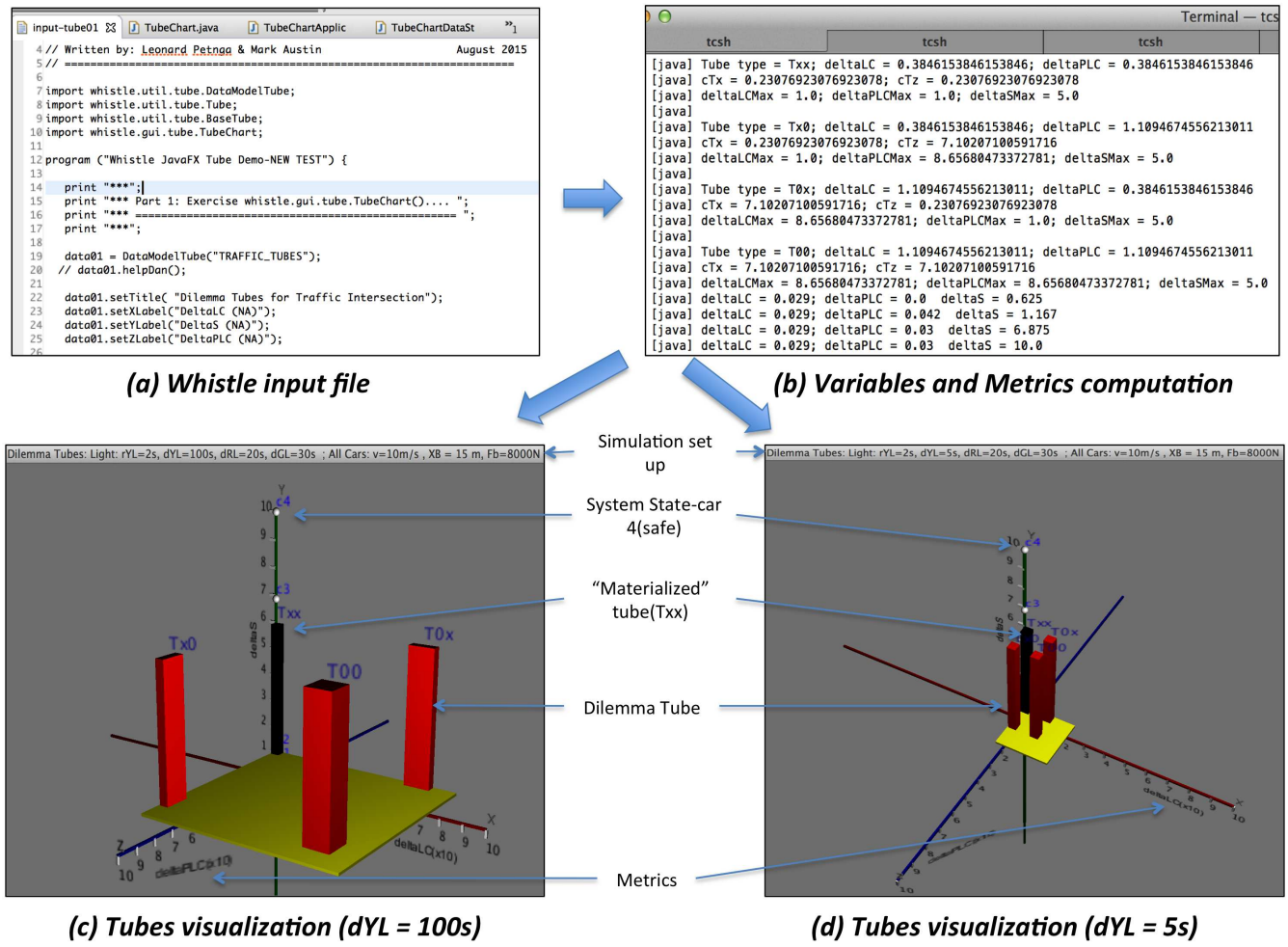


Figure 5. Schematic of system inputs and outputs. The sub-figures are: (a) Whistle input file, (b) variables and metrics computation, (c) tubes visualization for dYL = 100 seconds, and (d) tubes visualization for dYL = 5 seconds.

we use a text file, such as the one shown in Figure 5(a).

(2) The program instantiates a tube DataModel matched to the needs of the simulation. This will later serve as a place holder for the various versions of tubes as they are constructed and displayed.

(3) The system is initialized. This is done by configuring the stop light with predefined values to d_{YL}, d_{RL} and d_{GL}. As for the car, if the engineering simulation module (e.g., racetrack) is hooked to the integration platform, then a car type is selected based upon its weight and its physical parameters (initial velocity, trajectory and position). The corresponding component models are interfaced with the integration module.

Computational requirements during the simulation can be reduced through pre-computation and storage of the dilemma tube parameters, as described in the following steps (4)-(7). This is done for various values of r_{YL} and dimensionless indexes n and n' (see equations (13) and (14)).

(4) The number of dilemma tubes N that need to be visualized at each iteration of r_{YL} is determined as follows:

$$N = \begin{cases} 1 & \text{if } n \text{ and } n' \text{ are undefined} \\ n+2 & \text{if } n \geq 0 \text{ and } n' \text{ undefined} \\ n'+2 & \text{if } n \text{ undefined and } n' \geq 0 \\ (n+2)(n'+2) & \text{if } n' \geq 0 \text{ and } n \geq 0 \end{cases} \quad (19)$$

In equation (19), n is undefined when $\Delta_{LC} < 1$ and n' is undefined when $\Delta'_{LC} < 1$. In this configuration, the only tubes that can be viewed are of Type I, as per Figure 3.

(5) From the input file, a method of the tube DataModel file is called to generate a baseline empty tube as per the initial configuration of the traffic light. This results in the creation and storage of a new BaseTube that acts as a placeholder for the set durations of the three lights. For simulations involving multiple stoplights, the same method can be called repeatedly for each set of stoplights. Each call of this method will result in a TrafficTube model being created and instantiated.

(6) Next, a new method is called to create and update dilemma tubes for the given input baseline tube. This leads to: (a) the calling of the traffic tube instance, the extraction and storage

of the set value for d_{YL}, then, (b) the creation of the dilemma tubes via an update of the traffic tube for the decreasing values of r_{YL} from d_{YL} to 0. Besides the value of r_{YL}, the values of n and n' as well as the input baseline tube are needed. The foundational variables needed to display each dilemma tube are computed, i.e., the tube type, dimensions on axis and coordinates of their location in the dimensionless (delta) space, as shown in Figure 5(b). The total number of dilemma tubes created is determined, as per equation (19). In this case, we have $n = n' = 0$, which leads to four dilemma tubes, Txx, Txo, Tox and Too which are of types I, II, III and IV, respectively.

(7) The dilemma tubes are sorted and grouped by r_{YL}. This information will allow control of the display of tubes in a way that is consistent with the unfolding of r_{YL}

(8) With the computation and storage of dilemma tubes completed, we can now make the move toward their visualization. The first step consists of enabling Whistle access to the visualization tube model in order to create an instance of a JavaFX 3D chart. For those cases where the engineering simulation module is hooked to Whistle, the racetrack and its contents will be uploaded and displayed as per the set up in (3). Otherwise, the simulation can be done with the system state in the dimensionless space computed separately based on the initial set up and targeted configurations.

(9) The 3D scene for the tube charts is created then, the data stream system is configured and the data (flow) channel tube between the input file and the 3D GUI is created and initialized.

(10) The simulation of the engineering module is started. As the car follows the path toward the intersection stop line located at B, its position X is sensed. The remaining duration on the yellow light r_{YL} is measured from the clock. Both quantities are sent back to the computation module for processing. For each pair (XB, r_{YL}), the values of Δ_{LC}, Δ_S and $\Delta_{LC}^{'}$ are computed as per equations (10), (11) and (12). As a group, these values define the state of the system in the Δ space.

(11) The set of dilemma tubes corresponding to the value of r_{YL} is pulled from storage (see step 7) and "pushed" through the channel (see step 9) to the display GUI. We can now visualize an output similar to the ones shown in Figures 5(c) and (d). The yellow plate is the *Plan Tube* for the system in the (Δ_{LC}, $\Delta_{LC}^{'}$) space. It is built from the maximum values of both parameters for the set of dilemma tubes available for display and defines the system boundary at $\Delta_S = 1$ for which the dilemma tubes take shape.

(12) Identification mechanisms are encoded into the channel system to single out *materialized tube(s)* – that is, tubes for which the safety of the system has to be checked. Materialized tubes are within the immediate vicinity of a system state and, as such, depending on how compact the tube system is, there could be many of them. There is always at least one materialized tube at any moment (in black in Figure 5(c) and (d)). When a materialized tube contains a system state, it means that the system is unsafe. Such cases are quantified as "active tubes." We note here that the physical interpretation of an active tube is not that of an actual violation of the system safety constraint (see Section III), but that it will happen in the immediate future, and certainly within the time left on the yellow light (if any).

(13) Configuration of the tube system. The way the tubes appear on the visualization GUI depends on the values of dimensionless indexes n and n'. To identify the formation of the tubes, we look at the tubes from the top view in the plan ($\Delta_{LC}^{'}$,Δ_{LC}) in the computer screen reference system, i.e., with $\Delta_{LC}^{'}$ pointing downward and Δ_{LC} pointing right. As for the value of N in equation (19), four types of formation are possible:

$$TubeFormation = \begin{cases} point & \text{if } n \text{ and } n' \text{ are undefined} \\ line & \text{if } n \geq 0 \text{ and } n' \text{ undefined} \\ I & \text{if } n \text{ undefined and } n' \geq 0 \\ rectangle & \text{if } n' \geq 0 \text{ and } n \geq 0 \end{cases} \quad (20)$$

In the *point formation* the only tube that can be displayed is of Type I. In the *line formation*, realized tubes appear aligned horizontally on an axis parallel to the Δ_{LC} axis. A similar formation is observed in the *I formation* with the tubes being aligned vertically following the $\Delta_{LC}^{'}$ in the dimensionless space. The boundary of the last type of formation has the shape of a rectangle. When $n = n'$, it becomes a square as for the four-tube formation in Figure 5 (c).

VI. Safety Analyses

The purposes of this section are two-fold. First, we employ the simulation platform described in Section V to identify and analyze the key factors that affect the system level safety of the traffic system. In the second part of this section, single and set-pair factor safety analyses are performed to investigate how system safety depends on systematic adjustments to single factors (e.g., vehicle braking force) and combined sets of parameters.

1. Safety Factors Identification. Under the set of assumptions (A1) to (A3), and from Table I, the following six factors are single out for further consideration: weigh of the car(m), car velocity(v), car braking force(Fb), distance to stoplight (XB), remaining duration of the yellow light (r_{YL}), and configured duration (d_{YL}). For these studies we pick $n = n' = 0$ which leads to a four-tube square formation.

2. Single Factor Safety Analysis.

a/ Effect of Car Weight and Velocity. For this analysis, we use the set of four cars and assign for each simulation run a velocity within the range in Table I with a step of $5m/s$. The remaining four parameters are fixed to their set value. For each run, we observe and record the presence and name of any active tube (synonym of unsafe system) as well as the identity of the car whose state has been materialized in the active tube. The absence of any active tube means the system is safe for all vehicles. The results are summarized in a *parameter-based safety profile* as shown in Figure 6(a).

For this particular configuration of the traffic system, the active tube for all runs is the tube Txx, which is of Type I. The

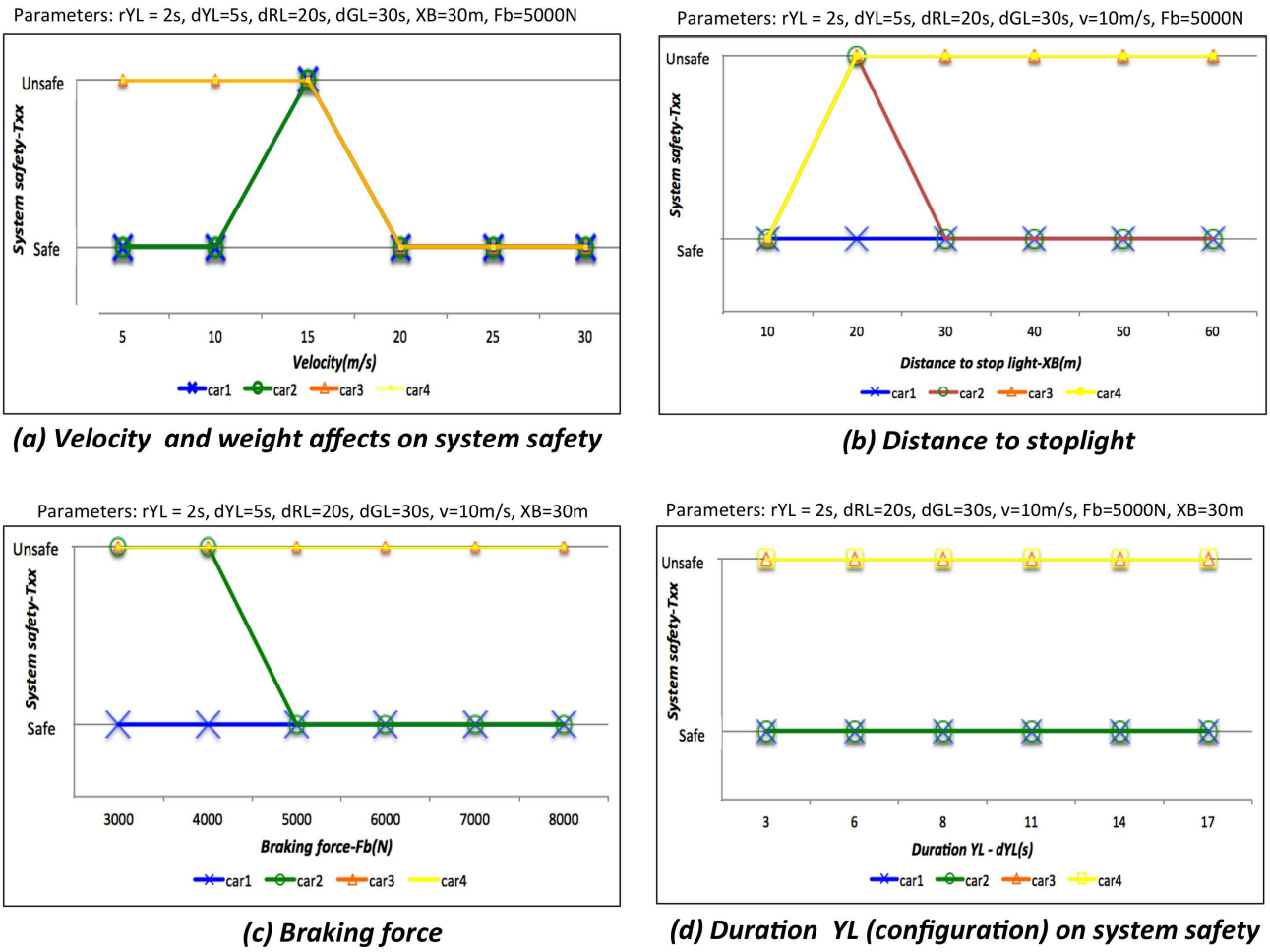


Figure 6. Parameters-based single factor safety profiles.

heavier cars (#3 and #4) violate the safety constraint at lower speed ($v \leq 15m/s$), while small and mid-size vehicles (#1 and #2) would not violate the safety constraint if they operate on both sides of velocity $v = 15m/s$. The combined effects of inertia and velocity play against safety (i.e., heavier cars lack agility – at velocity $v \leq 15m/s$, they can neither stop before nor clear the intersection within the 2s time interval). We note the troubling "unsafe" state for all cars at $v = 15m/s$. To summarize, operating heavier vehicles within higher velocity range and, small and average size vehicle at lower or higher velocities are the only way to keep the traffic system safe.

A quick evaluation of the sensitivity of the safety profile to changes in any of the fixed parameters shows that the only one for which it doesn't change significantly is d_{YL}. For instance, if we consider changes in r_{YL}, smaller and mid-size vehicles become safer as long as r_{YL} grows beyond $2s$ ($3s$ for heavier vehicles). At lower r_{YL} ($\leq 1s$), all vehicles tend to be unsafe except for smaller ones at low velocity ($v \leq 10m/s$). Given the relatively far distance (XB = 30 m) at which this evaluation is performed, there might still be room for improvement as the car gets closer to the intersection stop line, especially at low velocities.

b/ Effects of the Car Distance to the Intersection. For this study, we use the same set of four cars and keep track of the distance to the stop line, this time with a step of $10m$ which is used to define the location of sensing points for the system. And as with the previous analysis, the remaining four parameters are fixed to their set value. System safety is tracked by observing and recording the presence and name of active tubes along with the identity of the car whose state has been materialized in the active tube. Finally, the distance-to-stop-line safety profile (see Figure 6(b)) is generated.

We observe that as heavier vehicles (#3 and #4) approach the intersection, they are mostly unsafe until the last checkpoint, where their dynamic capabilities allow them to either stop safely before or clear the intersection within the remaining $2s$ on the yellow light. The small vehicle (#1) is safe all the time; with the exception of checkpoint $XB = 20m$ (which corresponds to the last location where heavier vehicles transition to a safe state), the mid-size vehicle (#2) are safe. An examination of the sensitivity of this profile to perturbations in r_{YL} reveals that heavier cars are more sensitive than mid-size and small cars. Away from the light ($XB \geq 50m$), heavier cars are unsafe and they will require $5s$, $4s$ and $3s$ on r_{YL},respectively at $40m$, $30m$ and $20m$ to avoid violating the

intersection safety requirement. Mid-size vehicles, in contrast, only require $3s$ at $20m$ to stop.

c/ Effects of the Car Braking Force. The same protocol is followed to study how car braking force affects system safety. To that end, we systematically vary the parameter F_b within the defined range in Table I using a $1000N$ step. This results in the braking force safety profile shown in Figure 6(c).

For this configuration of the system, the effect of the braking force is well perceived for the mid-size car ($\#2$) as it leaves the unsafe state when F_b increases and passes the $5,000N$ threshold. Under the same circumstances, heavier cars ($\#3$ and $\#4$) certainly need a braking force outside the current simulation range – in fact, our set value for the maximum force of $8,000N$ does not help switch the system back into safety. In other words, even a $8,000N$ braking force is insufficient to counter the kinetic energy of the vehicles and immobilize them within $XB = 30m$ and $r_{YL} = 2s$. Small cars are much more agile, and the minimum braking force of $3,000N$ is good enough to keep the smallest car ($\#1$) safe.

As the value of r_{YL} decreases, the safety profile for car $\#1$ is not affected as all for all values of F_b. However, below $5,000N$, the mid-size and heavier cars would require $r_{YL} \leq 4s$ to remain safe. Above that threshold force, only heavier car will need the same amount of time to stay safe. Thus, we can conclude that the higher the inertia of the vehicle, the higher breaking force and time on yellow light are needed for the system to remain safe.

d/ Effects of the Initial Configuration of the Yellow Light. As a final step in this experiment, we would like to understand how the configuration of the stoplight by the traffic engineer and, in particular, the duration of the yellow light d_{YL}, affects the system safety. To that end, we consider a fixed stoplight cycle duration $C = 55s$ and assign a progressively increasingly high percentage of that duration to the yellow light from 5% to 30% with a step of 5%; thus, the data range shown in Table I. The simulation is run for the various values of d_{YL} and results of the safety profile are shown in Figure 6(d).

We see from the safety profile that, for a given value of $r_{YL} = 2s$, increasing the actual configuration of the yellow light does not affect the outcome of system safety. However, a look at the corresponding tube formation shows that, as the value of d_{YL} increases, so is the spacing between the tubes. This translates into more room for safety, should the system manage to get out of unsafe situations, i.e., the volume occupied by the tubes. The contrast between the tube formations in Figures 5(c) and (d) illustrates this phenomenon. When $d_{YL} = 5s$, a low value, the rectangle formation is compact, and the tubes are closed to each other (see Figure 5 (d)). Should they realize all, there will be little to no room to avoid a violation of the safety constraint. Conversely, at higher $d_{YL} = 100s$ (for illustration only) there is plenty of room between the tubes. This means that, should there exist a mechanism to take advantage of the availability of this safety space to adjust r_{YL} to higher values, the safety of the system will be improved. These observations make the case for reconfigurable traffic lights that are capable of adjusting the remaining duration of the yellow light to resolve safety issues. Also, we note the variation in tube sizes in Figures 5(d) and 5(c), with Txx being the smaller and Too the bigger. This observation can be traced back to index k, as per equation (4), and its further propagation into the parameters that define the tubes as shown in Figure 3, especially those defined by equations (15) to (18). Finally, we note that $0 \leq r_{YL} \leq d_{YL}$ thus, the two variables are dependent. Setting d_{YL} from an initial position d_{YL1} to $d_{YL2} \geq d_{YL1}$ allows r_{YL} to add $d_{YL2} - d_{YL1}$ to its range which, as we have seen so far, adds more safe room for the overall system.

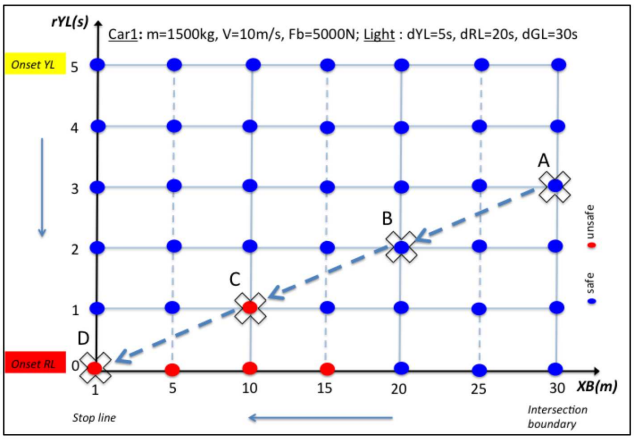
3. Set (pair) Factor Safety Analysis. Despite the valuable insight provided by single factor analyses in understanding system level safety, they provide just a "snapshot" view of the system through the perspective of the parameter considered for the analysis. The sensitivity of most safety profiles to changes in the values of r_{YL} clearly shows that even though most factors are set or controlled independently, their interaction is the key driver behind system level safety. Thus, there is a need to look at changes to system safety caused by adjustments to combined sets of parameters.

a/ Parameter-based Safety Template for Pair (r_{YL}, XB). Pairing the six parameters leads to fifteen possible sets. However, given that parameters such as r_{YL} and d_{YL} are dependent and others such as m and XB are constrained by the vehicle physics, not two sets of parameters are equally important or relevant for this study. Thus, we won't be analyzing the system safety for all pairs, but we will be looking at the pair (r_{YL}, XB), which illustrates the cyber-physicality of the traffic system as introduced in Section II. The protocol of the study described here can be repeated and applied to other pairs as well.

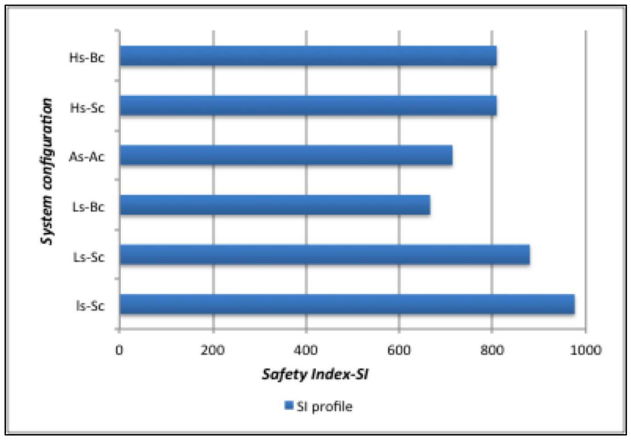
For set factor studies, all the parameters considered vary within their individual, predefined range. The other parameters are configured to their set values as presented in Table I. Running the simulation and recording the safety state of the system results in the creation of a parameter-based *safety template*, such as the one seen on Figure 7(a). This particular template is created with the configuration: $K \equiv (m = 1,500kg, v = 10m/s, Fb = 5,000N, d_{YL} = 5s, d_{RL} = 20s, d_{GL} = 30s)$. The template shows the safety state of each system operational point. A red dot signifies that under K, the system state is in an active tube (i.e., the system is unsafe). A blue dot means the system is safe. In practical terms, the template is an indicator of safety – for instance, under configuration K, if car $\#1$ crosses the intersection boundary (XB=30m) when there is only $3s$ left on the yellow light, the system will be safe as it will be located at $A(30m, 3s)$, which is a safe operational point on the template. If, however, the configuration K remains unchanged, the system will be unsafe $2s$ later at location $C(10m, 1s)$. Therefore, for the system to remain safe under K, the car has to enter the intersection when there is at least $4s$ left on the yellow light. These examples illustrate the greater insight, we can gain using safety templates, in the interplay between system parameters and their effects on system level safety.

b/ Parameter-based Safety Indexes for Pair (r_{YL}, XB). A subspace Us that contains all unsafe states of the system for the configuration K can be defined as follows:

$$Us^{K}_{(r_{YL},XB)} = \begin{cases} 0s \leq r_{YL} \leq 1s \\ 1m \leq XB \leq 15m. \end{cases} \quad (21)$$



(a) Safety template for the set (XB,rYL)



(b) Safety Index chart for set (XB, rYL)

Figure 7. Parameters-based safety templates and indexes.

Intuitively, one might think that a smaller subspace Us translates to a safer system, but this is only part of the story. Considering that an unsafe subspace might also contain safe states, as observed in this case, we ought to be able to quantitatively assess the safety of a configuration in a clear and simple way. To this end, we introduce the parameter-based *configuration safety index* SI as follows:

$$SI^K_{(r_{YL},XB)} = \left(1 - \frac{n_{U_K}}{n_K}\right) * 1000. \quad (22)$$

Here, n_{U_K} is the number of unsafe states (red dots) in Us and n_K the total number of states in the template for configuration K. For the safety template shown in Figure 7(a), we count $n_{U_K} = 5$ unsafe states and $n_K = 6 * 7 = 42$ total states. This leads to a configuration safety index of $SI^K_{(r_{YL},XB)} = 880$.

By systematically adjusting the vehicle weight (m) and velocity (v) we can generate an ensemble of safety templates, and then for each, compute the safety index. This leads to the safety index chart shown in Figure 7(b). The chart shows that for high speeds, both the smallest vehicle (S_c) and heaviest vehicle (B_c) have similar levels of safety. The smallest vehicle does a better job at lower velocities. In-between, the mid-size vehicle (A_c) cannot do better at average velocity (A_s). These results are consistent with the findings in 1.a/.

We note that this safety index does not capture the topology of unsafe and safe points in the Us subspace for (r_{YL}, XB). As seen in a/ above, that distribution is critical in predicting the future state of the system. Therefore, we cannot use the safety index SI to that same end. However, it can be used for a high level estimate of the parameter-based safety appreciation of the system safety before diving into topological considerations of Us for further investigation. To that extent, the two approaches serve complementary purposes.

4. Beyond Predefined Configurations and Pair Factors. Any change in the value of a parameter in the configuration K in Section 3.a/ automatically forces the switch and use of a different safety template (with the new value for that parameter) to predict the state of the system when the car reaches the stop line. This limits the ability of the Systems Engineer to navigate the design space of the traffic system. A possible solution is to flatten all independent variables in a pentagon-like diagram which will give a partial view of the whole design space. The actual full design space is much more complex (i.e., a five-dimensional shape) and almost impossible to visualize. Any combination of values of the five parameters (m, v, F_b, r_{YL}, XB), each within its respective range, is theoretically a valid point.

VII. Discussion

Our preliminary results are contingent upon assumptions (A1) through (A3) listed in Section V. Neglecting air resistance (A1) certainly simplifies the account of the dynamics of the cars but it comes at a price. With the acceleration null, the velocity is assumed constant on XB which leads to a constant value of Θ_B in equation (10) for all vehicles at the same velocity for the the same value of XB. This propagates all the way to the tubes visualization where, under such circumstances, points for the various cars will be stuck in the plan (Δ_S,Δ'_{LC}) at a single Δ_{LC} value. One opportunity for future work is to account for the air resistance in the dynamics of the car, through a drag force $f = k_1 * v^2$ for instance. This will lead to a more accurate model of the vehicle dynamic that will ultimately improve the quality of the results. The immediate effect on this tube framework will be the distribution of system states along the axis Δ_{LC} as well.

Task execution of the scenario introduced in Section V requires intensive computations and communication at multiple steps; this makes it hard for assumptions (A2) and (A3) to survive any physical prototype testing of the system. In fact, as many researchers have pointed out, not only do real-world computations and communication require finite amounts of time to complete [24][25], but delays of unacceptable duration can trigger accidents in traffic scenarios that are safety critical. Given that such considerations are platform-dependent, there should be in a future iteration of this work a mechanism to

account for delay information in the execution model, perhaps along the lines of what has been accomplished with Ptolemy [26].

VIII. Conclusion and Future Work

The purpose of this paper has been to introduce and describe a new and innovative tubular (3D) characterization of the dilemma zone problem. We have discussed the modeling, design and prototype simulation of a tubular framework that supports the study and analysis of the dilemma zone problem using a set of dimensionless metrics.

State-of-the-art approaches to the dilemma zone problem treat the cars and stoplights separately, with the problem formulation being expressed exclusively in either spatial or temporal terms. By taking on a systems perspective that allows for two-way interactions between the cars and stoplights, the proposed method leads to a dilemma tubes formulation that compactly describe sets of conditions for which the vehicle-light system will be in an unsafe state.

The essential elements of the two-way interaction are formally captured by three metrics: (1) the car stopping distance metric Δ_S, (2) the light-car crossing time metric Δ_{LC}, and (3) the light-car stopping time metric $\Delta_{LC}^{'}$ in dimensionless space (Δ). These three metrics work together to define a simple and precise way safety of the system in a manner that is consistent with the system decision tree. To support this formulation we have developed a flexible software architecture for the computation of metrics and implementation of the tubes. Simulations were performed and tubes were visualized under sets of physical and cyber parameters for the car and the light extracted from the system design space.

The single safety factor analysis indicates that system-level safety is strongly influenced by the combined effect of car weight m and velocity v, its distance to the stop light XB, and the configuration of the yellow light d_{YL}. Parameter-based safety templates, which are effective in predicting the future state of the system at the stop line, were created by pairing the remaining duration of the yellow light r_{YL} and XB. We have defined a parameter-based safety index SI as a first-order estimate of system level safety. This new metric enables the characterization and comparison of safety templates. All of these analyses work together to provide a deeper understanding of the dilemma zone problem and strategies for resolving unsafe scenarios. The proposed approach and preliminary results are consistent with research that has investigated the critical role of component interactions on the safety of complex systems [27].

Future versions of this work need to fully embrace the cyber-physicality of next generation traffic systems as described in Section II. Key characteristics of these developments would include semantically-enabled and efficient platform structures that can support the modeling, emulation and simulation of the behavior of real-world autonomous cars and intelligent traffic control systems as agent of cyber-physical transportation systems. To that end, an ontological architecture supporting the formal description of the relevant sub-domains involved is needed. Spatio-temporal reasoning supported by appropriately implemented semantic extensions (such as Jscience or Joda time) will enhance traffic agents decision-making capabilities. For the traffic system, the architectural framework will support reasoning in the dimensionless space and enable light reconfiguration, should a car be heading into a dilemma tube. The dilemma metrics introduced in this paper will be implemented in the Integrator rules engine. This entity (physically a smart traffic controller) will be the ultimate responsible of system-level decisions. Further details on the underlying semantic platform infrastructure supporting this architecture can be found in Petnga and Austin [28].

References

[1] L. Petnga and M. A. Austin, "Tubes and Metrics for Solving the Dilemma-Zone Problem," The Tenth International Conference on Systems(ICONS 2015), Barcelona, Spain, April 19 - 24, 2015, pp. 119–124.

[2] J. Sztipanovits, J. A. Stankovic, and D. E. Cornan, "Industry-Academy Collaboration in Cyber-Physical Systems(CPS) Research," *White Paper August 31*, White Paper, August 31, 2009.

[3] D. Winter, "Cyber-Physical Systems in Aerospace - Challenges and Opportunities," *The Boeing Company*, Safe & Secure Systems & Software Symposium (S5), Beavercreek, Ohio USA, June 14-16, 2011.

[4] Google inc., "The Latest Chapter for the Self-Driving Car: Mastering City Street Driving," Official Google Blog. N.p., n.d. Web. April, 28 2014, http://googleblog.blogspot.com/2014/04/the-latest-chapter-for-self-driving-car.html; Accessed : November 15, 2015.

[5] General Motors Corporation (GMC), "GM Unveils EN-V Concept: A Vision for Future Urban Mobility," http://media.gm.com; Accessed : November 15, 2015.

[6] Energetics Incorporated, "Cyber-physical Systems Situation Analysis of Current Trends, Technologies, and Challenges," Energetics Incorporated for the National Institute of Standards and Technology (NIST), 2012.

[7] D. Hurwitz, "The "Twilight Zone" of Traffic costs lives at Stoplight Intersections," Oregon State University, Corvallis, Oregon, USA, March 03, 2012.

[8] C. V. Zeeger and R. C. Deen, "Green-Extension Systems at High-Speed Intersections," Division of Research, Bureau of Highways, Department of Transportation, Commonwealth of Kentucky, April, 1978.

[9] D. S. Hurwitz, B. H. Wang, M. A. Knodler, D. Ni, and D. Moore, "Fuzzy Sets to Describe Driver Behavior in the Dilemma Zone of High-Speed Signalized Intersections," School of Civil and Construction Engineering, Oregon State University, USA and Department of Civil and Environmental Engineering, University of Massachusetts Amherst, USA, March, 2012.

[10] P. Li, "Stochastic Methods for Dilemma Zone Protection at Signalized Intersections," Doctor of Philosophy Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University, VA, USA, August, 2009.

[11] M.S. Chang, C.J. Messer, and A. J. Santiago "Timing Traffic Signal Change Intervals based on Driver Behavior," Transportation Research Record, 1027, pp. 20-30, 1985.

[12] P.D. Pant and Y. Cheng, "Dilemma Zone Protection and Signal Coordination at Closely-Spaced High-Speed Intersections," Report FHWA/OH-2001/12, Ohio Department of Transportation, Columbus, OH, USA, November, 2001.

[13] D. Maas, "Dilemma Zone Elimination," Sacramento Department of Transportation, Sacramento, CA, USA, 2008.

[14] K. Zimmerman and J.A. Bonneson, "Number of vehicles in the dilemma zone as a potential measure of intersection safety at high-speed signalized intersections," 83^{rd} Annual Meeting of the Transportation Research Board Washington, D.C., USA, January, 2004.

[15] N.G. Wassim, J. Koopmann, J.D. Smith, and J. Brewer, "Frequency of Target Crashes for IntelliDrive Safety Systems," US Department of Transportation – National Highway Transportation Safety Administration, DOT HS 811 381, October, 2010.

[16] J. Chu, "At a Crossroads: New Research Predicts which cars are likeliest to Run Lights at Intersections," http://newsoffice.mit.edu/2011/driving-algorithm-1130; Accessed: November 15, 2015.

[17] Siemens Corporation, "Intelligent traffic solutions," http://www.siemens.com; Accessed : November 15, 2015.

[18] International Business Machine (IBM) Corporation, "IBM Intelligent Transportation Solution for Active Traffic Management," Systems and Technology Group, IBM Corporation, November, 2013.

[19] Carnegie Mellon University (CMU), "Smart Traffic Signals," http://www.cmu.edu/homepage/computing/2012/fall/smart-traffic-signals.shtml; Accessed : November 15, 2015.

[20] L. Petnga and M. A. Austin, "Ontologies of Time and Time-based Reasoning for MBSE of Cyber-Physical Systems," 11th Annual Conference on Systems Engineering Research (CSER 2013), Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

[21] L. Petnga and M. A. Austin, "Cyber-Physical Architecture for Modeling and Enhanced Operations of Connected-Vehicle Systems", 2nd International Conference on Connected Vehicles (ICCVE 2013). Las Vegas, NV, December 2-6, 2013.

[22] P. Delgoshaei, M.A. Austin, and D. A. Veronica, "A Semantic Platform Infrastructure for Requirements Traceability and System Assessment," The Ninth International Conference on Systems (ICONS2014), Nice, France, February 23 - 27, 2014.

[23] P. Delgoshaei, M.A. Austin, and A. Pertzborn, "A Semantic Framework for Modeling and Simulation of Cyber-Physical Systems," International Journal On Advances in Systems and Measurements, Vol. 7, No. 3-4, December, 2014, pp. 223–238.

[24] E. Lee, "Computing Needs Time," Technical Report No. UCB/EECS-2009-30, Electrical Engineering and Computer Sciences University of California at Berkeley, February 18, 2009.

[25] R. Wilhelm and D. Grund, "Computing takes time, but how much?," Communications of the ACM, Vol. 57, Issue. 2, February, 2014, pp. 94–103.

[26] C. Ptolemaeus, Editor, "System Design, Modeling, and Simulation using Ptolemy II", Ptolemy.org, 2014.

[27] N. C. Leveson, "A Systems-Theoretic Approach to Safety in Software-Intensive Systems," IEEE Transactions on Dependable and Secure Computing, Vol. 1, Issue. 1, January-March 2004, pp. 66-86.

[28] L. Petnga and M. A. Austin, "Semantic Platforms for Cyber-Physical Systems," 24th Annual International Council on Systems Engineering International Symposium (INCOSE 2014). Las Vegas, USA, June 30 July 3, 2014.

Multi-Scheme Smartphone Localization with Auto-Adaptive Dead Reckoning

Michael Jäger, Sebastian Süß, Nils Becker
Institute of Software Architecture
Technische Hochschule Mittelhessen - University of Applied Sciences
Gießen, Germany
Email: {michael.jaeger, sebastian.suess, nils.becker}@mni.thm.de

***Abstract*—Most indoor localization approaches for mobile devices depend on some building infrastructure to provide sufficient accuracy. A commonly used method is the fusion of absolute position measurements with relative motion information from sensor units. This paper examines the requirements for smartphone localization in areas consisting of several buildings and open space, where a single positioning method might deliver good results at one location but might also fail at another. It is shown that, for several disparate reasons, a localization system combining alternative positioning techniques and going beyond the scope of a single hybrid method, is desirable. The paper proposes such a multi-scheme system with a three-layer architecture consisting of base methods, hybrid methods, and scheme selection. Automatic selection of an appropriate scheme is described for heterogeneous infrastructure within multi-story buildings and for indoor-outdoor transitions. Support of several distinct hybrid methods can be based on the same generic fusion algorithm. The paper proposes a novel lightweight fusion algorithm, called "auto-adaptive dead reckoning". It can be used in indoor and outdoor environments to combine an absolute localization method, e.g., Wi-Fi-based signal strength fingerprinting, in an adaptive way with inertial pedestrian navigation. Based on an accuracy factor reflecting the current context conditions of a location measurement the influence of each of the involved estimates is weighted accordingly. In a case study using Wi-Fi fingerprinting, accuracy has been improved by 43% in an indoor environment. Hence, more genericity can be obtained without loss of accuracy.**

Keywords–Indoor Positioning; Pedestrian Activity Classification; Dead Reckoning; Wi-Fi Fingerprinting.

I. Introduction

This article is based on [1], where the concept of indoor smartphone localization with auto-adaptive dead reckoning has been introduced. This work is extended by a multi-scheme concept combining alternative positioning techniques and providing multi-floor and multi-building support even in the case of heterogeneous infrastructure, as well as seamless indoor-outdoor transitions.

Location awareness has become a key feature of many mobile applications. A common problem in the context of navigation and tracking applications is the accurate localization of a mobile device within a well-known area comprising several buildings and also open space, e.g., a company premises, an airport, an exhibition center, or a university campus. Such sites are typically heterogeneous in the sense that a single localization method delivers good results in one sub-area but fails in another. Popular indoor solutions use hybrid methods comprising a suitable combination of an absolute positioning method with sensor-based relative positioning.

A. Absolute Positioning

With respect to mobile devices like smartphones an absolute positioning method estimates the device location in terms of latitude and longitude. Relative positioning determines the distance and heading of the movement, when a device is moved to a new position. Elevation might also be of interest, especially in order to determine the floor-level in a multi-story building. As far as outdoor environments are concerned absolute positioning is commonly based on global navigation satellite systems (GNSS) [2], like the well-known Global Positioning System (GPS) [3], the Russian GLObal NAvigation Satellite System (GLONASS), the Chinese BeiDou, or the european Galileo system. While deviation of second generation GNNS will be in a magnitude of some centimeters in outdoor use [4], satellite systems are not expected to provide sufficient accuracy inside of buildings without being supported by expensive complementary ground component (aka "pseudolite") technology [5].

Thus, the quest for accurate and inexpensive indoor localization techniques has fostered intensive research over the last decade and resulted in a number of different promising approaches. While solutions based on cellular signals have not successfully solved the problem of insufficient accuracy, the use of IEEE 802.11 wireless networks, e.g., Wi-Fi, has been widely adopted for real-time indoor localization purposes [6–10]. The rapidly growing usage of Wi-Fi access points as navigation beacons is, among other reasons, due to the ubiquitous availability of Wi-Fi networks and to the fact that a smartphone can easily measure Wi-Fi signal strength values. "Received Signal Strength Indication" (RSSI) values of several Wi-Fi access points are used to determine the current position of a Wi-Fi receiver. The advent of cheap bluetooth low energy (BLE) beacons [11], e.g., iBeacons [12], might foster their use for the same purpose within the next few years.

Ultra-wideband (UWB) radio has the potential to become the most successful base technology for indoor smartphone positioning. It can be used similar to Bluetooth for inter-device communication. But, most important, UWB has been designed specifically to enable precise distance measurements even through walls. Localization of an UWB-equipped smartphone is based on distance measurements between the device and UWB tags. As of summer 2015, the first UWB-enabled smartphones became available on the market [13].

Regardless of the beacon types and localization algorithms, absolute indoor localization methods currently rely on a dense beacon mesh to allow for accurate localization. In a heterogeneous area, thus, a practically important issue is the device localization at spots that lack a sufficiently good beacon signal coverage.

B. Pedestrian Dead Reckoning

A substantially different approach to localization is dead reckoning, a well-established relative positioning method. Starting from a known position, inertial and other sensors, e.g., accelerometers, gyroscopes, magnetic field sensors, or barometers, are used to track relative position changes. For example, distance estimation in pedestrian dead reckoning (PDR) systems [14] is typically based on step detection with motion sensors and step length estimation. This is combined with direction information from an electronic compass. Modern smartphones are crammed with all kinds of sensors and, thus, are well-suited for inertial navigation. Sensor-based localization is, however, subject to unbound accumulating errors, and therefore needs frequent recalibration.

An additional challenge for indoor PDR systems is the floor level determination within multi-story buildings. Measuring vertical displacement is straightforward for barometer-equipped smartphones. Vertical movements are usually associated with floor changes using elevators, stairs, or escalators. From the athmospheric pressure measurements the vertical displacement can be inferred sufficiently accurate to determine the final floor level unambiguously [15]. The exact height of this level, taken from the building model, can in turn be used to recalibrate the pressure altimeter. In the absence of a barometer, sensor-based pedestrian activity classification can be used to detect floor changes and to determine the final level.

C. Fusion of Absolute Positioning with PDR

Accuracy requirements for localization systems depend on their intended use. A pedestrian indoor navigation system frequently needs to distinguish if a user is in a corridor or in an adjacent room, or has to lead the user unambiguously to one of two doors placed side by side. Whereas errors of more than one meter are undesirable in such a setting, an accuracy of a few centimeters is barely ever needed. GPS or Wi-Fi-based fingerprinting typically have average errors of several meters. According to [16], the average errors of inertial positioning systems range between 60 centimeters and "corridor width". The aim of combining inertial and less accurate absolute methods is to provide an average accuracy significantly below one meter while limiting the accumulation of inertial measurement errors.

A hybrid method is a fusion of an absolute positioning method with sensor-based navigation. For example, in a GPS-based automotive navigation system sensor-based speed and direction measurements are used to track the current position whenever GPS signals are degraded or unavailable, e.g., in a tunnel. Similarly, a PDR system can be combined with GPS into a hybrid solution for outdoor areas or, together with any absolute indoor position method, e.g., Wi-Fi-based, for use within a building.

An interesting aspect of hybrid systems is the distribution of roles. The absolute positioning could be seen as a minor subsystem of the sensor-based system supplying the start position and, occasionally, intermediate positions for recalibration. However, existing systems typically use the absolute positioning method as a primary method, whereas sensor-based location measurements are only used in case of degraded beacon signals. The absolute base-method is used to compute position estimates ("fixes") at regular intervals. Each fix is considered a new known start position for inertial navigation. Whenever a fix is not available due to poor signal coverage, the relative movement from the last fix location is used to determine the current device location. A car navigation system, e.g., will use inertial navigation in a tunnel. After leaving the tunnel, it will return to the primary method GPS. This commonly used combination pattern does not take into account that, depending on the current beacon reception conditions and despite the accumulating sensor measurement errors, the dead-reckoned position will often be more accurate than the base method fix.

D. Auto-Adaptive Dead Reckoning

This paper proposes a hybrid localization solution, called "auto-adaptive dead reckoning", incorporating a more sophisticated way of combining absolute and relative positioning. Considering that the accuracy of each of the involved methods might fluctuate extremely between measurement locations, the fusing algorithm evaluates context conditions, that are critical for the accuracy, with every measurement. A measurement value that is considered accurate has a stronger impact on the result. The term "adaptive" is used for a fusion algorithm that associates a weighting factor with each fused method in order to adapt the algorithm to site-specific measurement conditions, e.g., Wi-Fi signal coverage within a building. Static adaptation refers to a configuration time weighting, whereas auto-adaptive (or dynamic) fusion refers to a dynamic weighting for each individual measurement. This advanced fusing technique has been implemented as a component of a mobile application for the Android platform, called SmartLocator [17]. It is explained in more detail in Section IV.

E. Supporting Different Localization Schemes

Going beyond the auto-adaptive dead reckoning approach, this paper introduces a novel concept for the integration of distinct localization methods in the same system. A "Localization Scheme" is a realization of a localization approach, comprising a selection of base technologies, a system model, and appropriate algorithms. A multi-scheme localization system supports several alternative schemes with automatic scheme selection. In fact, there are several distinct motivations for envisioning a multi-scheme approach:

Localization Infrastructure Dependencies: Regarding a user entering and leaving several buildings while roaming through a complex area, it is quite obvious that a single localization

technique is not sufficient. In the outdoor environment GPS can be used, either stand-alone or fused with PDR into a hybrid scheme. Regarding the indoor case, though infrastructure-less positioning is possible, most solutions build on specific infrastructure, e.g., Wi-Fi radio maps, to obtain more accurate measurements. Ideally, all the buildings in the area of interest are equipped with a homogeneous localization infrastructure. Even in this case, a localization system has to switch between indoor and outdoor localization schemes. However, most areas with several buildings will not be homogeneous in this sense, but will rather require different techniques for indoor positioning. For example, some buildings might have an infrastructure for Wi-Fi absolute positioning, some might be equipped with iBeacons, and others will have no appropriate infrastructure at all. Even within a single building, different localization approaches could be required, e.g., because a specific infrastructure is not available in all floors.

Different Pedestrian Activities: Another aspect influencing the localization approach is the way of pedestrian movement. A positioning system using PDR must in some manner deal with different movement patterns, e.g., vertical movements due to stairs or elevator usage.

Device Hardware Capabilities: Novel smartphone hardware features often offer new opportunities for positioning, e.g., NFC- or BLE-support, barometers, hardware step detectors, or UWB. Whenever new hardware facilitates an advance in localization, it will be exploited for that purpose. However, not all devices are equipped with all kinds of available sensors and radios. For example, only a few smartphone models have built-in barometers. However, a barometer could be used to determine the current floor after an elevator trip in a relatively simple and reliable way, whereas other methods, e.g., based on radio beacons or inertial sensors, have several disadvantages. Another example are hardware step detectors, which might be preferable to software solutions due to lower power consumption. A localization system might keep up with the ever-increasing device diversity in several ways:

1) The system uses only commonly available hardware components, e.g., Wi-Fi and common inertial sensors, thereby forgoing the new opportunities.
2) The system requires a high-end smartphone with several non-standard hardware features in order to exploit these for positioning.
3) A system uses different positioning techniques for devices with distinct positioning capabilities.

Hence, a multi-scheme approach does not only address the diversity of contextual conditions, or rather the availability of some positioning infrastructure. It is also a suitable concept for considering different pedestrian moving patterns and different positioning-related hardware capabilities.

F. Requirements for Localization in large and complex Areas

Subsuming and extending the discussion above, the following requirements should be met by a complex area positioning system:

Requirement 1. *A positioning scheme should provide sufficient accuracy.*

Though accuracy requirements for pedestrian localization depend to some degree on the intended application domain, most of currently available absolute methods, e.g., GPS, Wi-Fi-based, BLE-based, are considered too inaccurate to be used stand-alone. Thus, the important consequence of this requirement is that fusion with PDR or some other enhancement is necessary.

Requirement 2. *Several positioning schemes have to be supported.*

In addition to a GPS-based outdoor scheme, the system has to support at least one, but typically more than one indoor scheme. Several PDR-schemes in order to address different pedestrian moving patterns and detect floor-level changes, as well as scheme selection depending on device capabilities are not considered as requirements, but rather as desirable features.

Requirement 3. *The system should automatically select the most appropriate scheme.*

While roaming in the area, repeated manual selection of a new positioning technique is not acceptable for a user. There must be a mechanism for detecting transitions between subareas requiring different localization approaches and for selecting an appropriate scheme.

Requirement 4. *Unnecessary power consuming measurements have to be avoided.*

With respect to localization, power consumption issues arise with high processor load due to probabilistic fusion algorithms and with the use of sensor and radio equipment. As a consequence, it is not acceptable to use several techniques simultaneously, when a single one is sufficient. For example, continuously searching for a GPS fix, while the user stays in a building for hours, will drain the battery unnecessarily. The same holds for dispensable Wi-Fi scans etc.

Several camera-based localization schemes have been proposed. Since these are inherently power-consuming they have been out of consideration in the multi-scheme approach presented in this paper.

G. Overview

This paper presents a smartphone localization system satisfying the requirements listed above. It is based on a three layer architecture. At the bottom layer, the system comprises a PDR subsystem and several basic absolute positioning methods, namely, Wi-Fi-based fingerprinting, BLE-based fingerprinting, Near Field Communication (NFC) [18], and GPS. The intermediate layer consists of hybrid positioning schemes. Each scheme fuses PDR with an absolute positioning method using a generic and efficient auto-adaptive dead reckoning algorithm. At the top layer, a multi-scheme mechanism is used to detect necessary scheme switches and select the most appropriate scheme automatically.

After presenting related work in Section II, the proposed positioning system is described in the succeeding two sections.

The focus of Section III is on the overall architecture and the automatic selection of an appropriate localization scheme, while Section IV explains the fusion of PDR and absolute positioning using the auto-adaptive dead reckoning approach.

Section V discusses experimental results showing the achieved accuracy improvements over non-hybrid as well as hybrid methods with non-dynamic method fusion. Section VI reviews some benefits and shortcomings of the presented approach and future research plans.

II. Related Work

A large number of solutions to the problem of real-time indoor localization have been proposed, and several efficient algorithms for absolute and relative positioning have been published. Auto-adaptive dead reckoning, as presented in this paper, is based upon Wi-Fi fingerprinting, BLE-fingerprinting, GPS, NFC, and PDR.

A. Wi-Fi-based Fingerprinting

Using an existing Wi-Fi infrastructure for indoor localization is an obvious and well-investigated approach. While RSSI-based distance calculations have proven to be too inaccurate to be used for trilateration-based indoor localization, RSSI-fingerprinting methods are particularly useful in the context of real-time smartphone positioning [6–10].

Fingerprinting is based on probability distributions of signal strengths for a set of access points at a given location. A map of these distributions is used to predict a location from RSSI samples. This radio map is created in an offline learning phase for a number of known locations called calibration points. In order to determine the device position, RSSI values are collected from all visible access points and the radio map is searched for locations with similar signal strengths.

A major advantage of Wi-Fi fingerprinting is that it does not require specialized hardware [7][19][20]. Nevertheless, a non-dynamical Wi-Fi infrastructure with good coverage is needed to achieve reasonable positioning results.

However, the most important disadvantage is the elaborate fingerprint database creation and maintenance. Since the accuracy of position estimates highly depends on the density of the radio map [7], the construction of a high-density map is inevitable for Wi-Fi-only positioning solutions. The auto-adaptive algorithm, in contrast, allows for a significant reduction of the number of calibration points without loosing too much overall accuracy.

In order to avoid the map creation overhead completely, zero-effort solutions based on crowdsourcing have been proposed [21][22]. Although efficient map creation is outside the scope of this paper, it should be noted that map creation and map usage algorithms are typically loosely coupled. Thus, any successful approach to automate map creation could possibly be generalized for usage with existing fingerprinting systems.

B. Sensor-based Positioning

According to [16], PDR systems can be classified as Inertial Navigation Systems (INSs) or Step-and-Heading Systems (SHSs). While the INSs typically require specialized hardware, the SHSs are well-suited for PDR with smartphones.

The SmartLocator solution presented in this paper implements an SHS, which builds upon efficient algorithms for step detection and heading estimation. The heading is determined by a sensor fusion method described in [23]. Step detection exploits the smartphone's accelerometer signals. Whenever a peak with a certain amplitude at the z-axis is noticed, a step can be assumed [24]. A modified Pan-Tompkins algorithm is used for signal preparation. Pan-Tompkins, in the context of step detection, has been used by Ying [25] before.

For larger areas different pedestrian moving patterns have to be considered, as elevators, stairs, or escalators will be used temporarily. Promising approaches to sensor-based pedestrian activity classification have been presented in [26] and [27].

C. Method Fusion

An interesting approach combining Wi-Fi-based fingerprinting with PDR was proposed in [28]. Their fusing algorithm uses a limited history of location measurements for both methods to achieve accurate position estimations. Another promising solution is described in [29]. The algorithm builds on a statistical model for Wi-Fi-localization avoiding the effort of fingerprinting map creation, deliberately taking into account the resulting poor accuracy of the obtained position information. Both fusing methods comprise the use of floor plans and particle filters in order to obtain more accurate position information [30].

Particle filters are probabilistic approximation models based on Bayesian filters [31], which can be used for fusing PDR measurements with the results of absolute positioning methods. They also provide a means for incorporating movement constraints obtained from a floor map of a building or a footpath or road map. In the context of smartphone localization, a particle consists of an estimation for position and heading together with a weight value representing the probability that the estimation is correct. The current state of a smartphone is not represented by a single location and heading, but rather by many particles. State changes have to be handled according to the underlying motion model by a recursive algorithm whose computational cost depends to a considerable degree on the number of particles. Basically, when a step is detected, each particle is propagated to a new position by exploiting Bayes theorem, and new weight values are computed. Next, a resampling filter is applied to replicate particles with large weights and to remove those with negligible weights.

When used in a laboratory environment with a high-quality PDR system, i.e., a special purpose, firmly attached, foot-mounted sensor system, particle filters have shown to produce very accurate position estimates. With a stock smartphone sensor system, however, movements of the device are more loosely coupled with the movements of its user, which induces more uncertainty into the measurements. In a probabilistic model this additional uncertainty leads to a considerable increase in the measurement error variances, which in turn has to be accounted by an increased number of particles. As

a consequence, particle filters induce high processor load and have considerable impact on power consumption [16]. Moreover, suitable floor maps have to be supplied and maintained.

III. Multi-Scheme Positioning

This section describes the proposed multi-scheme approach and the implementation of scheme selection in the Smart-Locator positioning system. After introducing the system architecture in Section III-A, Section III-B presents the coarse localization subsystem used to determine the current building and floor-level. Section III-C describes the implementation of the high-level scheme switching.

A. System Models and Architecture

Since multi-scheme support is one of the outstanding features of the proposed system, the term ”localization scheme”, the motivation for a multi-scheme approach, and the relation between scheme selection and method fusion deserve some further explanation.

It is a common property of many advanced fusion-based localization approaches that they are based on a specific dynamic state-space system model with a hidden state. Partial information about this state can be obtained by observations or measurements. The model contains a set of assumptions about the measurement methods used and the context in which the measurements are taken. A common approach is to use a system model for pedestrian movements based on heading and stride size. The positions obtained from an absolute positioning method are the observations, which can be used to infer information about the system state. The fusion of PDR and absolute method is used to recalibrate the PDR state and can be implemented, e.g., by a particle filter.

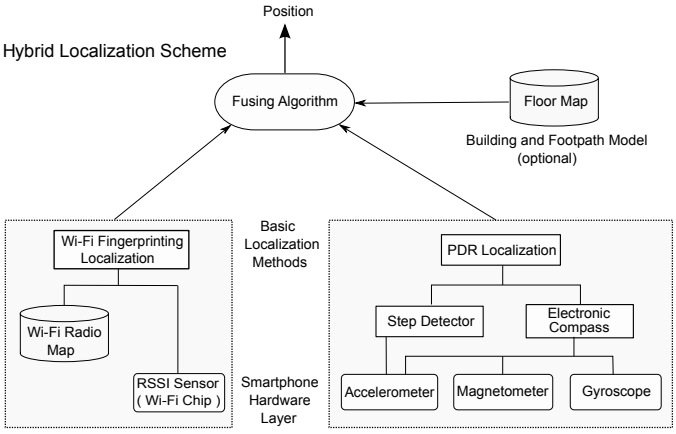


Figure 1. Layers of a typical Hybrid Localization Scheme.

The architectural layers of such hybrid localization systems are shown in Figure 1. The illustration abstracts from the fact that the PDR subsystem could also use internal fusing algorithms for sensor measurements, e.g., a Kalman filter as part of the electronic compass implementation [32]. To avoid confusion, the term ”localization method” is used for the lower level subsystems of the basic localization method layer, e.g., PDR, GPS, Wi-Fi- or Bluetooth-fingerprinting. Instead, the term ”localization scheme” denotes the uppermost architectural layer of a hybrid system, which is characterized by the high-level fusion algorithm.

A complex area with heterogeneous positioning infrastructure requires the combination of different localization techniques. The relation between the model-based view and the requirements for complex areas can be clarified by some examples:

- If the same PDR system is combined with GPS outdoors and with Wi-Fi indoors, a switch between these schemes corresponds to a replacement of the observation model.
- Changing the way of moving around, e.g., taking an elevator to change the floor, or using a shuttle bus between two buildings of the area, corresponds to the replacement of underlying pedestrian movement model.
- Supposing that a building map is used in a particle filter to detect and eliminate through-the-wall movements, this map usage should perhaps be switched off in order to avoid unnecessary processor load in a wall-free environment or in an unmapped subarea. This would correspond to a change in the fusion algorithm.

Figure 2 depicts the components of a hybrid localization scheme from a model-based view. Supporting different schemes, i.e., different localization approaches, in a single system makes sense for a variety of reasons already presented above. However, the figure illustrates that a scheme change can be related to

1) a change in the pedestrian activity,
2) a change of the PDR recalibration mechanism,
3) or a change in some location-depending algorithmic aspect of the fusion algorithm.

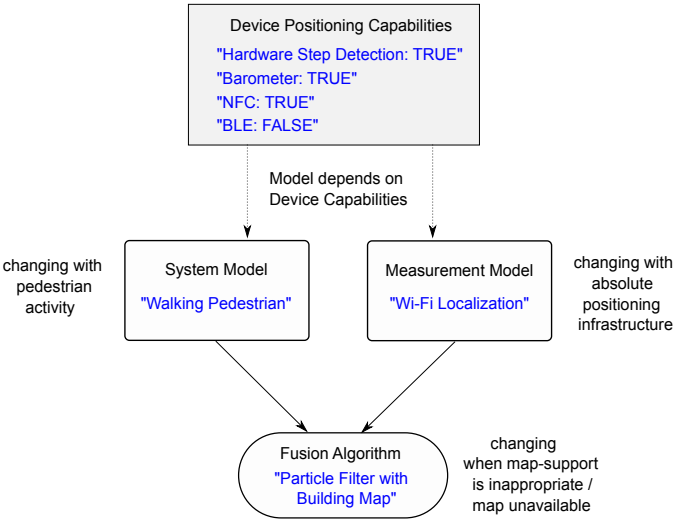


Figure 2. System-theoretic View of a Localization Scheme and its Components.

Whereas the scheme changes listed above are dynamic, i.e., imposed by location changes, the dependencies between schemes and hardware features are location-independent and static. Multi-scheme support corresponds to the alternative

usage of several distinct models. Hence, scheme switching is not a replacement for method fusion, but rather a higher level concept for automatically selecting an appropriate (possibly hybrid) localization scheme. Support for several localization schemes with automatic scheme selection introduces a new top-level layer into the system architecture as shown in Figure 3.

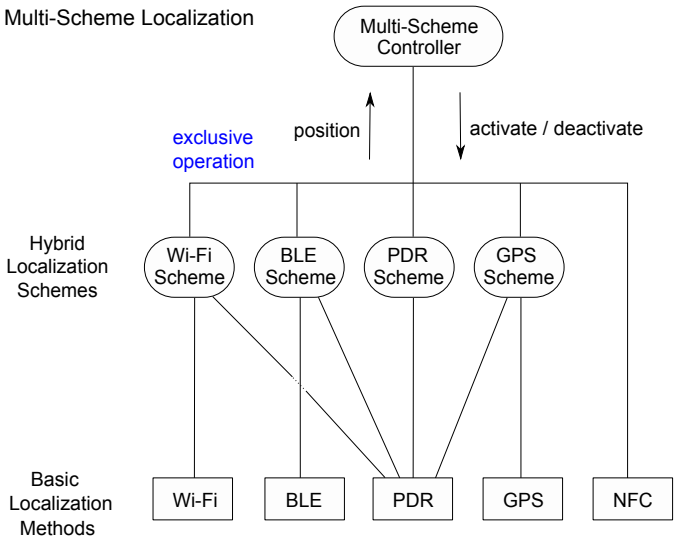


Figure 3. Layers of a Localization System with Multi-Scheme Support.

The multi-scheme controller has to determine dynamically which localization scheme is the most appropriate one at the current device location. There are some noteworthy relations between the components of the bottom and intermediate-level layers:

1) PDR is used in several schemes to improve accuracy: Fingerprinting-based methods using Wi-Fi or BLE-beacons can be fused with PDR the same way as GPS.
2) Methods that provide sufficient accuracy on their own are not fused with PDR. A currently supported accurate method is the reading of NFC tags with known position. Future methods based on the upcoming UWB and next generation GNNS technologies will also fall into this category.
3) The PDR scheme in the model is not necessarily identical to the basic PDR method. It is rather a sensor-based algorithm to be used in the absence of radio beacons, which might make use of a floor map or path model to recalibrate the measurements.

B. Coarse Localization Subsystem (CLS)

In order to recognize the current floor-level after a vertical movement, as well as for the detection of indoor/outdoor transitions, a coarse localization subsystem consisting of a set of special localization methods is used. The CLS architecture is illustrated in Figure 4.

The CLS can be considered as a simple, special-purpose, secondary localization system. It does not use radio maps or method fusion. Instead, the CLS base algorithms, e.g., the

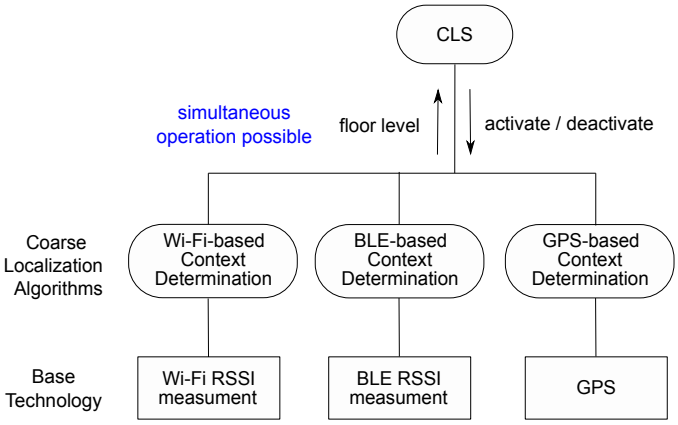


Figure 4. The Coarse Localization Subsystem.

Wi-Fi-based context determination, are lightweight algorithms which essentially check, whether a floor-specific beacon constellation is detected, or whether a GPS-fix is available. The outdoor environment is treated as a special floor-level in this context. An important difference to the primary multi-scheme localization is the possibility of simultaneously searching for usable Wi-Fi, BLE, and GPS-signals. The CLS supports two operating modes.

- The *limited mode* is used to check the current floor-level against a small set of possible levels. A typical use is the floor-level determination after elevator usage. Since the former position is available, the set of reachable floors can be determined in advance as will be explained in Section III-C.
- The *unlimited mode* is used for an initial estimation of building and floor whenever there is no former location available, e.g., when the system is started. Compared to the limited mode, the set of possible floors to be checked is the set of all floors of all buildings.

The CLS reports successful context determinations immediately, but stays turned on until it is explicitly deactivated by the multi-scheme controller.

C. Transition Detection and Scheme Selection

When a roaming user moves to an area requiring a different positioning technique, the multi-scheme controller has to detect this situation, to determine the new positioning scheme, and to switch to the selected scheme.

Positioning Context: A positioning context is a spatial area with exactly one associated localization scheme. Although arbitrary contexts could be defined, only three types are considered here, i.e., the outdoor context, a specific building, or a floor of a building. Figure 5 gives an example of positioning contexts with their associated localization schemes.

Positioning Context Map: The association of a scheme with a context is done via a *positioning context map*. With each positioning context this map associates

- A localization scheme descriptor (LSD) identifying the algorithm to be used and containing links to radio map and building model.

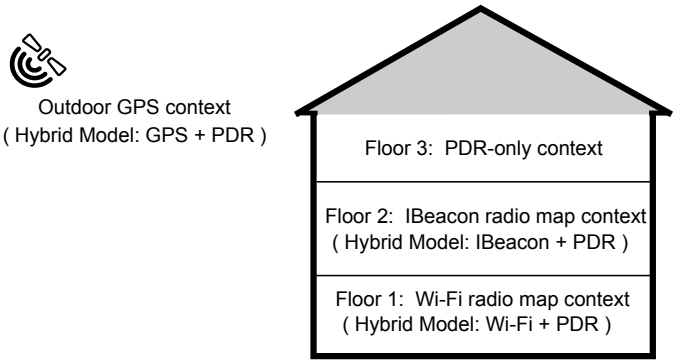


Figure 5. Positioning Contexts Example.

- A context determination descriptor (CDD) identifying the method to be used for recognizing the context, e.g., Wi-Fi or BLE, and a context-specific beacon constellation.

The construction and maintenance of this map is, in fact, very lightweight. SmartLocator is based on an Openstreetmap (OSM) ([33]) model with extensions for multi-story buildings. In the extended OSM model, any building and any floor is represented by a relation. Those relations can be tagged with additional information, e.g., a localization scheme descriptor. All relations are associated with the outdoor context per default. Thus, if a new localization infrastructure is established in a floor of a building, that is supported by a localization scheme S, in order to update the map one only needs to tag the floor relation with the descriptor for S.

Locations and Port Objects: In the building model, the location of an object consists of a latitude-longitude pair, a building ID, and a floor-level. Doors, stairs, or elevators are special port objects, which represent links between buildings, between two or more floors of the same building, or between the building and the outdoor environment. An elevator, e.g., is represented by several port objects, all sharing the same latitude and longitude, but with different floor-levels. A door leading from one building directly into another one is represented by a pair of port objects belonging to two distinct buildings but having the same geographic position attributes.

The neighbourhood of a port object consists of the object itself and all other port objects that are linked to it as "reachable" objects. Essentially, these are the possible endpoints of elevators, escalators or stairs, together with the objects of adjacent contexts (other building, outdoor environment) which share the same geographic position. Each port object is linked to its neighbours, such that the neighbourhood can be computed efficiently. A port object is member of its own neighbourhood to model situations like an elevator trip ending in the original floor.

Transitions between Floors and other Positioning Contexts: When a user is roaming through a building, the positioning system always knows the current building and floor-level and checks against the building model, whether the user is near one of the floor's port objects. If this is the case, the system enters a transition detection state providing continuous attempts to detect movements across the borders between two adjacent positioning contexts. Entering this state, those adjacent positioning contexts are computed as follows. First, the port object's neighbourhood is extracted from the building model. For each location in this neighbourhood, the corresponding positioning context is determined and the context-associated descriptors LSD and CDD are looked up in the positioning context map.

The list of CDDs for the adjacent contexts is subsequently used for an activation of the CLS in limited mode. The CLS will repeatedly check for BLE-, Wi-Fi-, or GPS signals according to the CDDs of the adjacent contexts and report the context identification results until transition detection is eventually deactivated by the multi-scheme controller. The deactivation criteria depend on the type of port object. In case of an elevator, the transition is assumed to be completed, when the user is walking again. At this time, a unique floor-level identification is required from the CLS. For staircases, the end of transition is assumed after counting as many steps as the staircase has according to the building model and a unique floor identification is available from the CLS. Only if the floor has changed, the position is recalibrated to the staircase endpoint location. Admittedly, this is to some extend error-prone with respect to the horizontal location, e.g., when a sportive person takes two steps at once, or if someone turns back after nearly having reached the end of the stairs. In the case of a door, transition detection ends when five steps have been counted from the begin of transition. Obviously, these five steps need not necessarily be steps away from the door. In those rare scenarios, where the transition detection is switched off but the user is still at or near the original port object location, the system will immediately return to transition detection state again.

Leaving transition detection state means deactivation of the CLS. If a transition was detected, the scheme is switched according to the LSD of the new context. In transition detection state, several ambiguous situations are possible, reflecting either the uncertainty about the current floor-level during a vertical movement, or about the exact progress of passing through a door. For example, a user could walk towards a building entrance door, stay there for a while, turn around and walk back. Though a context transition does not happen in this case, the system is in transition detection state as long as the users smartphone position is estimated to be near the door. Two adjacent contexts might be identified by the CLS at the same time, e.g., if at a building entrance a GPS position is available but also the floor-identifying Wi-Fi access point RSSI value ranges have been verified.

The context transition for border locations is depicted in Figure 6, the numbers denoting the order in information flow. The intermediate states have been omitted for clarity.

It should be noted that an appropriate infrastructure is mandatory in this context. In practice, existing Wi-Fi infrastructure will commonly be usable for this purpose without much additional effort.

A common scenario is that a user leaves a building after

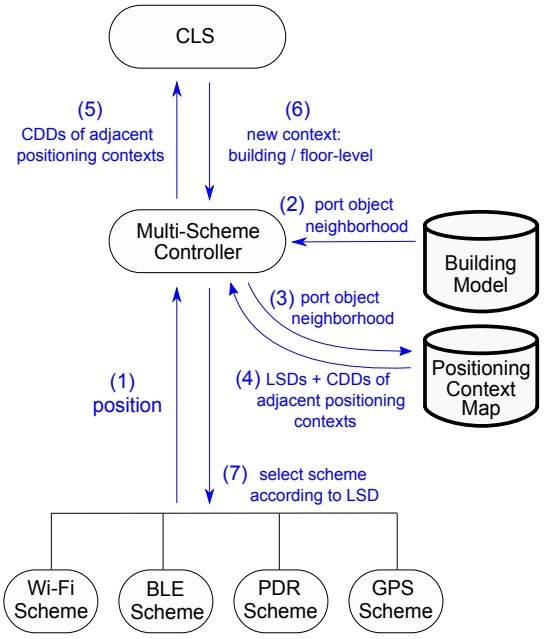


Figure 6. Context Detection near a Context Border Location.

working there for hours. As long as the current location is not directly adjacent to a door or elevator leading to the outside of the building, GPS is turned off and the device tracks itself using the context's indoor scheme. Only when the user moves to a door leading to the outside, i.e., a port object with an outside context neighbour, GPS will be turned on again.

IV. Intermediate-Level and Basic Positioning

This section describes auto-adaptive dead reckoning and its implementation in the SmartLocator positioning system. As already described above, positions determined with GPS, Wi-Fi, or BLE are considered inaccurate, whereas NFC-based positioning is treated as accurate. Using small low-cost NFC paper tags the maximal reading distance for a smartphone will be below 20mm. Thus, the accuracy of an NFC-based device location estimation is essentially determined by the accuracy of the position information associated with the NFC tag, while the distance between tag and device is typically insignificant. The problem of determining accurate positions of fixed objects is out of the scope of this paper. However, the position of an NFC tag can often be determined by attaching it to an object appearing in a floor plan, e.g., a door or a stair railing, and measure the tags location relative to this object.

Whenever an accurate location measurement can be obtained, it overrides all other measurements.

In addition to the absolute positioning capabilities, SmartLocator incorporates a PDR subsystem with step detection and heading estimation. The stride size is simply set to a user-specific fixed value. However, using the absolute localization methods, it could straightforwardly be augmented with automatic stride size recalibration.

The emphasis of this section is to present the way of fusing PDR with an absolute positioning method. The term "auto-adaptive dead-reckoning" refers to this fusing approach. From the perspective of PDR, absolute localization is needed to obtain an initial position and for recalibration. In contrast to a full recalibration, we propose a partial recalibration determined by a dynamic weight, which reflects the accuracy of the absolute location estimation.

It is a particular strength of the approach to be generically usable with any absolute positioning method. Figure 7 illustrates how absolute location sources are combined with relative positioning information. A deep discussion of all of the supported absolute methods is out of the scope of this paper. Therefore, only the Wi-Fi fingerprinting scheme is considered as a typical example.

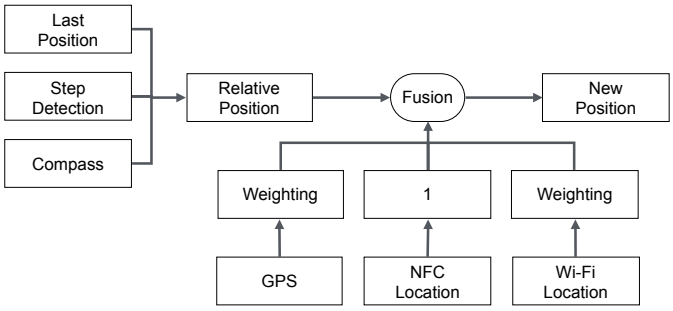


Figure 7. SmartLocator Positioning Concept.

The following subsections describe the Wi-Fi fingerprinting approach (Section IV-A), the step detection algorithm (Section IV-B) and the auto-adaptive fusion (Section IV-C).

A. Fingerprinting

During the training phase a radio map is created, containing Wi-Fi samples for a set of calibration locations. Each entry consists of a location and a set of signal strength values $\{s_1, \ldots, s_n\}$ obtained at this location. If k is the number of accesspoints, each sample s_i is a vector of k integral measurements (in dBm).

A position estimation is a result of the process chain shown in Figure 8. After scanning the RSSIs from all visible access points a Wi-Fi fingerprint is created. To select only well-known beacons, a SSID and a BSSID filter are applied. Subsequently, the MinRSSI filter eliminates access points which are unusable for localization due to their low RSSI level. Before the cleared fingerprint is matched against the radio map a BSSID filter is applied to reduce the number of calibration points and thus the expense of distance determination.

The distance between the current fingerprint and the calibration point fingerprints in the radio map is computed using the naïve Bayes classifier [7][19][20], which is more accurate than algorithms comparing distances between RSSIs [34–37]. This advantage has been confirmed during the evaluation of this positioning system.

If $s = (s_1, \ldots, s_k)$ is the vector of RSSI values obtained at the current location and $P(x|s)$ denotes the probability that s is obtained at an arbitrary location x, the problem is to

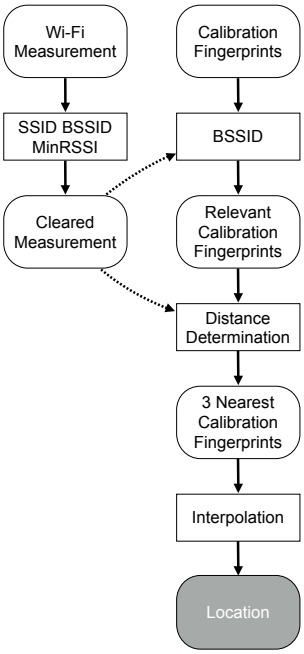


Figure 8. Fingerprinting Process Chain.

determine the location that maximises this probability. Using Bayes theorem,

$$P(x|s) = \frac{P(s|x)P(x)}{P(s)}, \tag{1}$$

assuming that $P(x)$ is identical for all locations, and considering that $P(s)$ is location-independent, the remaining problem is the determination of $P(s|x)$, since

$$\underset{x}{\operatorname{argmax}}\, P(x|s) = \underset{x}{\operatorname{argmax}}\, P(s|x). \tag{2}$$

Assuming that RSSI values of all access points are independent of each other,

$$P(s|x) = \prod_{i=1}^{k} P(s_i|x). \tag{3}$$

A common approach to determine $P(s_i|x)$ from the radio map for a single access point AP_i is to assume a Gaussian distribution of the RSSI values obtained at a location x [34][19, p. 36]. The probability density function of this distribution is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \tag{4}$$

where μ and σ denote the mean and the standard deviation, respectively, which are estimated from the samples in the radio map using the maximum likelihood method.

Since the radio map contains only fingerprints of calibration locations, an interpolation scheme is used to determine the actual position. Using the k-nearest neighbors algorithm for $k = 3$, the position estimation $\hat{x}$ is obtained by interpolating the locations l_1, l_2, l_3 of the three best fitting fingerprints using $P(l_i|s)$ as a weighting factor:

$$\hat{x} = \frac{\sum_{i=1}^{3} l_i P(l_i|s)}{\sum_{i=1}^{3} P(l_i|s)}. \tag{5}$$

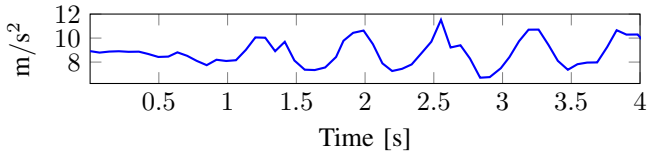
B. Step Detection

The step detection algorithm recognizes pedestrian movements based on a simple peak detection algorithm described by Link et al. [24]. To improve the amount of detected steps and decrease the appearance of false positive detections, the signal is prepared by applying a slightly modified version of the *Pan-Tompkins* method.

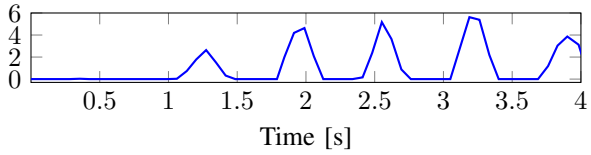
$$y(n) = \begin{cases} \frac{1}{4}[2x(n) + x(n-1) - x(n-3) - 2x(n-4)] & \text{if } y(n) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$y(n) = (1 + y(n))^2 - 1 \tag{7}$$

A derivative operator uses low-pass filtered acceleration values in order to suppress low-frequency components and enlarge the high frequency components from the high slopes (6). Negative values are discarded, as they are not needed for the peak detection. Figure 9 shows the incoming acceleration signal before (a) and after (b) this preparation.



(a) Raw Acceleration at Z-Axis.



(b) Squared Derivative Signal.

Figure 9. Acceleration Measurements Before and After Preparation.

The step detection algorithm examines the signal for peaks by comparing the last three values, represented by the red squares in Figure 10. A step is assumed whenever the signal changes by a certain threshold. After a step has been detected, the algorithm pauses for 300ms to prevent a step from being detected twice.

C. Auto-Adaptive Dead Reckoning

The major innovation of SmartLocator's intermediate-level hybrid localization is the accuracy-dependent fusion of absolute and relative positions. Traditional dead reckoning systems overwrite past position determinations whenever a new absolute position is available. This is not reasonable whenever absolute positions' accuracy is bad or varying. Therefore, every

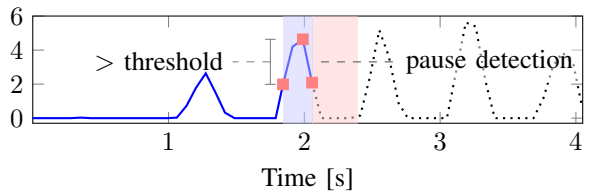


Figure 10. Step Detection Example. Red Squares Represent Analyzed Values.

absolute position is reckoned with past position estimations. The weighting of the new absolute position depends on an estimation of its accuracy. As a consequence, accurate absolute positions have a greater influence on the final position than less reliable position estimates.

E.g., Wi-Fi positions determined in an area with poor Wi-Fi coverage just have little influence on the final position estimation and the position determined by detecting the pedestrian's steps and heading is weighted strongly. On the other hand, Wi-Fi positions which are determined in an area with lots of access points and good signal quality are used to correct the drift which may occur due to inaccuracies in step detection and heading estimation.

Let $AbsPos$ be a location coordinate estimate obtained by an absolute positioning method at time t_{AbsPos}, e.g., a Wi-Fi or GPS position. The contribution of $AbsPos$ to the resulting location information $FusedPos$ depends on the method-specific accuracy factor $accuracy(AbsPos)$. This factor, which is obtained by context evaluation, reflects the measurement's context-dependent accuracy.

In addition, a time-dependent factor $drift(t_{AbsPos})$ is added to the accuracy factor. In this way, sensor drifts in the relative position will be taken into account and absolute positions have a stronger influence if the last position determination was long ago. The linear $drift(t_{AbsPos})$ used in SmartLocator is represented by Figure 11.

New calculations of $AbsPos$, $PDRPos$ or $FusedPos$ are triggered by time, NFC read, signal loss or user movement events.

$$FusedPos = AbsPos * \alpha + PDRPos * (1 - \alpha) \quad (8)$$

$$\alpha = max(accuracy(AbsPos) + drift(t_{AbsPos}), 1) \quad (9)$$

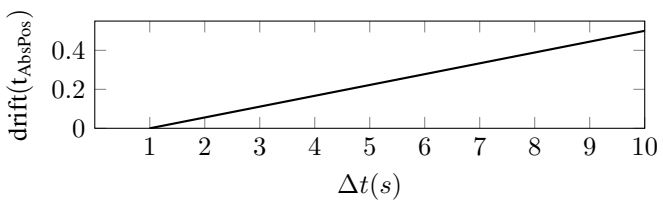


Figure 11. Time-Dependent Factor.

Accuracy Factors: The accuracy factor $accuracy(AbsPos)$ depends on the currently used positioning method. The following methods are used for Wi-Fi fingerprinting, GPS and NFC.

Wi-Fi: Several evaluations with an existing Wi-Fi infrastructure yielded an average error of 2.94 meters for pure Wi-Fi positioning. However, the error varied from 0.07 to 7.99 meters. Figure 12 shows the analysis of the gathered test data, revealing a relation between the average error and the amount of access points, which have been available for position determination. Even in case of good Wi-Fi coverage, error varies from 0.3 to 7.3 meters.

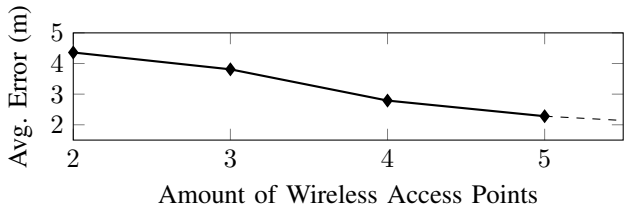


Figure 12. Accuracy Factor for Wi-Fi Positioning.

The accuracy factor of the Wi-Fi positioning method, illustrated in Figure 13, takes this relation into account to reduce the influence of unreliable position measurements.

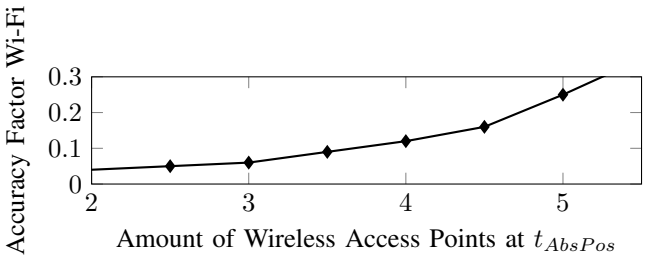


Figure 13. Wi-Fi Accuracy Factor Depending on Amount of Access Points.

GPS: The GPS position is determined by the smartphone through the operating system API. This API associates with each GPS position an accuracy property, which represents an estimated average error in meters. The accuracy factor, shown in Figure 14, is based on this accuracy property.

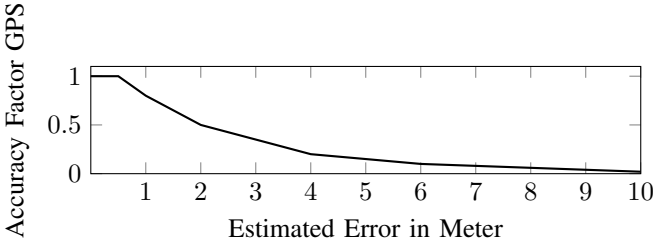


Figure 14. Accuracy Factor for GPS Positioning.

NFC: Near Field Communication (NFC) is used for positioning by placing passive NFC tags at points of interest. In order to scan an NFC tag, the smart phone needs to get in touch with it. Therefore, the location of the smart phone can be expected to be the location of the NFC tag. As a consequence, the accuracy factor of NFC always returns the maximum value of 1, which means that an NFC position overwrites prior location determinations completely.

V. Evaluation

SmartLocator has been tested under realistic circumstances in a university campus. However, it is a system undergoing continuous further development. The heuristic rules for scheme switching, as described in Section III, are still under evaluation. Also, experiments addressing the use of short-range, low signal level BLE-beacons for accurate positioning at port locations are not finalized.

Therefore, the focus of this evaluation is a detailed discussion of the Wi-Fi fingerprinting approach. Using eight Wi-Fi access points for positioning, fingerprints at 67 different locations have been recorded. The fingerprint locations are distributed uniformly with a distance of two meters. Hence, an area of about 280 m^2 is covered. Four orientations have been measured for any location. Three fingerprints for each orientation, resulting in an overall amount of 804 fingerprints.

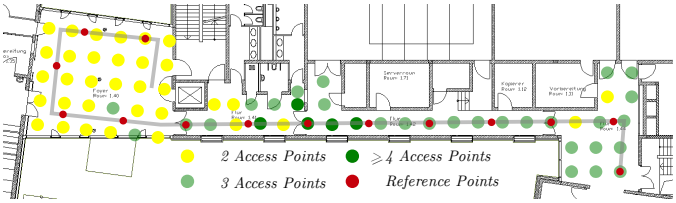


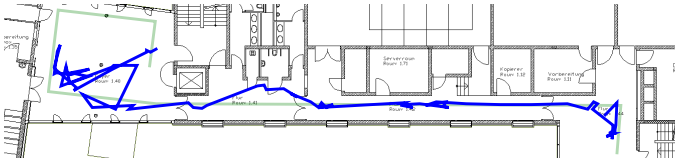
Figure 15. Wi-Fi Positioning Test Area with Fingerprints.

A track of 70 meters has been walked in various speeds, with different devices and in different directions to get a representative evaluation. 14 reference positions have been marked at the track. Those known reference positions are compared to the estimated positions, to determine the accuracy of the different approaches. Figure 15 shows the test environment, including the test track, which is illustrated by a grey line.

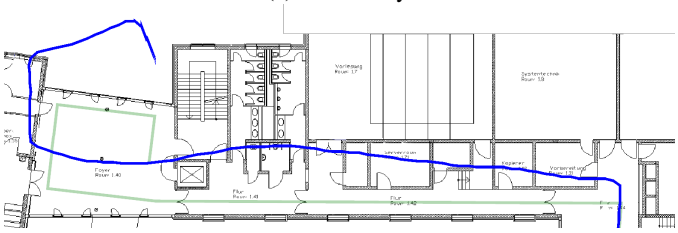
Figure 16 shows a visualization of one test run. The test started in the bottom right corner and followed the light green path. The blue line represents the actual positioning result. Figure 16b shows the results gathered with traditional dead reckoning, which means that absolute positioning results overwrite prior positioning estimations. Figure 16c presents a static weighting of 0.5, i.e., new absolute positions are just reckoned up by half. Figure 16d visualizes the positioning results achieved with a dynamic, auto-adaptive combination.

Remarkably, all figures reveal a clearly visible deviation from the real path at the same location (in front of the restrooms, left of the middle). This results from a coincidence of two local environment conditions. The first factor is the poor Wi-Fi-coverage in this area. Furthermore, a heavy metal fire door impacts the magnetometer of the electronic compass. Obviously, if neither of the involved measurement methods obtains an accurate location, the method fusion cannot compensate the resulting drift completely.

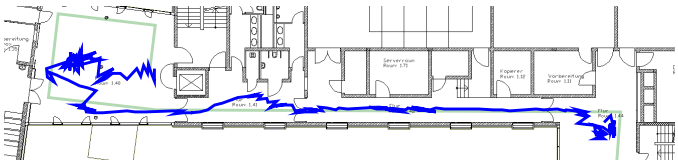
The evaluation revealed that the traditional dead reckoning (Trad. D.R.) approach performed even a little bit worse than the pure Wi-Fi positioning. A static combination of relative and absolute positions was able to slightly improve the positioning accuracy, especially in the foyer at the left side of the



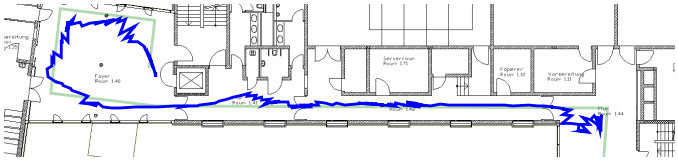
(a) Wi-Fi only.



(b) Traditional Dead Reckoning.



(c) Static Weighting of ($\alpha = 0.5$).



(d) Auto-Adaptive Weighting.

Figure 16. Comparison of Different Weightings.

floor plan. Auto-adaptive combination of Wi-Fi and relative positioning is able to reduce the average positioning error significantly. The average error has been improved from 2.94m (Wi-Fi only) to 1.67 meters, the upper quartile from 3.54m to 2.29m. Figure 17 shows the error ranges for the evaluated approaches.

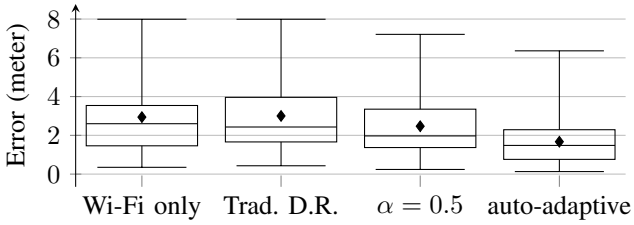


Figure 17. Error Ranges for Different Weightings.

VI. Conclusion

The presented approach has two important aspects. First, it introduces a general concept for integrating different localization methods into a single layered system. This is exploited for dynamically selecting a positioning scheme that is most appropriate for the current location with respect to supporting infrastructure. However, the multi-scheme approach is just as

well suitable for pedestrian activity classification and selective support of algorithms that build on non-standard device hardware features. The main benefit of the multi-scheme approach is that a user can rely on seamless localization in larger areas, typically exhibiting heterogeneous positioning conditions. A disadvantage is the need for a building map containing information about building entrance locations, stairs, elevators, and positioning context information as described in Section III-C. However, compared to a detailed building model containing all rooms, corridors, and doors, or even a fingerprinting database, this map is leightweight, and its construction and maintainance does not require much additional effort.

The second remarkable characteristic is the auto-adaptive dead reckoning algorithm for fusing PDR and an absolute positioning method into a hybrid scheme. Due to the genericity with respect to the absolute positioning method, this fusion approach is well-suited as a central building block within a multi-scheme architecture. Nevertheless, auto-adaptive dead reckoning provides accurate measurements even in areas with low radio beacon coverage. A comparison of Wi-Fi-based auto-adaptive dead reckoning with other advanced indoor localization systems shows that errors are in the same order of magnitude. For example, the Zee localization system [22] combines crowdsourcing of Wi-Fi fingerprints with a sophisticated map-based particle filter algorithm, which records a user's path through a building and uses map-matching to obtain additional information about PDR parameters and absolute locations. Zee can be combined with Horus [19] or EZ [38] and performs very well in a building with narrow corridors and obstacles that restrict the set of possible paths. The 50%ile and 80%ile errors are reported as 1.2m and 2.3m, respectively, which is slightly better than the results of the auto-adaptive dead reckoning evaluation. Furthermore, if largely unrestricted roaming is possible, e.g., in spacious halls, a map-based approach like Zee cannot exploit its strengths. The auto-adaptive dead reckoning approach seems to be quite promising, although additional evaluations with different environment conditions are necessary to gain more confidence in the statistical evaluation. More sophisticated accuracy estimation methods [39] and the additional use of floor map information [29] could probably improve this result further.

The evaluation shows that areas with bad Wi-Fi coverage and large rooms benefit the most. As a result, this positioning system can be used in areas which do not meet the requirements for Wi-Fi-only positioning approaches.

An unsolved problem is the determination of an initial position at starting locations with poor Wi-Fi coverage. Considering the enormous effort needed to construct a fingerprinting database, it obviously makes sense to also consider the selective deployment of NFC tags in such areas. These tags are cheap, permit exact localization, and will be supported by the vast majority of future smartphones. Moreover, the implementation of NFC-based localization has shown to be rather uncomplicated.

Compared to the more elaborate particle filters, auto-adaptive dead reckoning is a lightweight algorithm imposing only modest CPU load. The low-complexity fusion method and the avoidance of elaborate probabilistic algorithms result in a good real-time behaviour. Several test runs with different smartphones have shown that even on low-end hardware the SmartLocator runs without any visible performance problems. However, a more detailed analysis of algorithmic performance factors would be interesting, since time-consuming computations have negative effects on response times and power consumption.

TABLE I. Operation Modes and Power Consumption.

Operation mode	Active Components	Power Consumption
No motion	Motion sensors	Very low
Indoor Wi-Fi	PDR and Wi-Fi	Low
Indoor BLE	PDR and BLE	Low
Outdoor GPS	PDR and GPS	High
Indoor/indoor border	PDR/Wi-Fi/BLE	Moderate
Indoor/outdoor border	PDR/GPS/Wi-Fi or PDR/GPS/BLE	High High
Initialisation	PDR/GPS/Wi-Fi/BLE	High

Moreover, the approach carefully avoids unnecessary sensor usage. Investigations of the influence of sensor scanning on power consumption with different smartphones [40, 41] reveal that the GPS antenna is a major power consumer reducing battery life up to 50%, whereas the impact of inertial sensors, magnetometers, or barometers is negligible unless high sampling rates inhibit the monitoring processor from staying in a low-power idle mode. The proposed multi-scheme algorithm generally activates only one localization scheme at a given location, using only the scheme-related device components. Simultaneous activation of several absolute localization methods is restricted to a few special scenarios, i.e., system initialisation and detection of border-crossing movements, like indoor-outdoor transitions, near port locations. Furthermore, if no motion is detected, the localization system changes into a power-saving mode, reducing its activities to motion sensor scanning every two seconds. The properties of the multi-scheme algorithm imply several power consumption scenarios as shown in Table I.

References

[1] N. Becker, M. Jäger, and S. Süß, "Indoor smartphone localization with auto-adaptive dead reckoning," in Proceedings of the 2015 Tenth International Conference on Systems, Barcelona, Spain, 5 2015, pp. 125–131.

[2] C. J. Hegarty and E. Chatre, "Evolution of the global navigation satellitesystem (gnss)," Proceedings of the IEEE, vol. 96, no. 12, 2008, pp. 1902–1917.

[3] US Government, "Official U.S. government information about the global positioning system (GPS) and related topics," http://www.gps.gov, 2015, accessed: March, 3rd 2015.

[4] P. Misra and P. Enge, Global Positioning System: Signals, Measurements and Performance Second Edition. Lincoln, MA: Ganga-Jamuna Press, 2006.

[5] J. Wang et al., "Pseudolite applications in positioning and navigation: Progress and problems," Positioning, vol. 1, no. 03, 2002.

[6] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, ser. MobiSys '05, 2005, pp. 205–218.

[7] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 2. IEEE, 2000, pp. 775–784.

[8] M. Weber, U. Birkel, R. Collmann, and J. Engelbrecht, "Wireless indoor positioning: Localization improvements with a leaky coaxial cable prototype," in 2011 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Guimaraes, Portugal, pp. 21–23.

[9] T.-N. Lin, S.-H. Fang, W.-H. Tseng, C.-W. Lee, and J.-W. Hsieh, "A group-discrimination-based access point selection for WLAN fingerprinting localization," Vehicular Technology, IEEE Transactions on, vol. 63, no. 8, 2014, pp. 3967–3976.

[10] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," Computer Networks, vol. 47, no. 6, 2005, pp. 825–845.

[11] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," Sensors, vol. 12, no. 9, 2012, pp. 11 734–11 753.

[12] N. Newman, "Apple iBeacon technology briefing," Journal of Direct, Data and Digital Marketing Practice, vol. 15, no. 3, 2014, pp. 222–225.

[13] Spoonphone.com. Bespoon. [Online]. Available: http://spoonphone.com/en/ [retrieved: Jul., 2015]

[14] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in Proceedings of the 3rd Workshop on Positioning, Navigation and Communication, 2006, pp. 27–35.

[15] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, and S. Agarwal, "Barometric phone sensors: more hype than hope!" in Proceedings of the 15th Workshop on Mobile Computing Systems and Applications. ACM, 2014, p. 12.

[16] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," IEEE Communications Surveys & Tutorials, no. 15, 2013, pp. 1281–1293.

[17] N. Becker, "Development of a location-based information and navigation system for indoor and outdoor areas," Master's thesis, Technische Hochschule Mittelhessen, Giessen, Germany, 2014.

[18] R. Want, "Near field communication," IEEE Pervasive Computing, vol. 10, no. 3, 2011, pp. 4–7.

[19] M. A. Rehim, "Horus: A WLAN-based indoor location determination system," Ph.D. dissertation, University of Maryland, 2004.

[20] Y. Chen and H. Kobayashi, "Signal strength based indoor geolocation," Princeton University, Tech. Rep., 2002.

[21] P. Bolliger, "Redpin-adaptive, zero-configuration indoor localization through user collaboration," in Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments. ACM, 2008, pp. 55–60.

[22] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in Proceedings of the 18th annual international conference on Mobile computing and networking. ACM, 2012, pp. 293–304.

[23] S. Ayub, A. Bahraminisaab, and B. Honary, "A sensor fusion method for smart phone orientation estimation," in 13th Annual Post Graduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting, 2012.

[24] J. B. Link, P. Smith, N. Viol, and K. Wehrle, "Footpath: Accurate map-based indoor navigation using smartphones," in Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on. IEEE, 2011, pp. 1–8.

[25] H. Ying, C. Silex, A. Schnitzer, S. Leonhardt, and M. Schiek, "Automatic step detection in the accelerometer signal," in 4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007). Springer, 2007, pp. 80–85.

[26] X. Chen, S. Hu, Z. Shao, and J. Tan, "Pedestrian positioning with physical activity classification for indoors," in Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011, pp. 1311–1316.

[27] S. Khalifa, M. Hassan, and A. Seneviratne, "Adaptive pedestrian activity classification for indoor dead reckoning systems," in Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on. IEEE, 2013, pp. 1–7.

[28] L.-H. Chen, E.-K. Wu, M.-H. Jin, and G.-H. Chen, "Intelligent fusion of wi-fi and inertial sensor-based positioning systems for indoor pedestrian navigation," 2014.

[29] F. Ebner, F. Deinzer, L. Köping, and M. Grzegorzek, "Robust self-localization using Wi-Fi, step/turn-detection and recursive density estimation," in International Conference on Indoor Positioning and Indoor Navigation, vol. 27, 2014, p. 30th.

[30] S. Thrun, W. Burgard, and D. Fox, Probabilistic robotics. MIT press, 2005.

[31] J. Hightower and G. Borriello, "Particle filters for location estimation in ubiquitous computing: A case study," in UbiComp 2004: Ubiquitous Computing. Springer, 2004, pp. 88–106.

[32] D. Roetenberg, H. J. Luinge, C. T. M. Baten, and P. H. Veltink, "Compensation of magnetic disturbances improves inertial and magnetic sensing of human body segment orientation," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 13, 2005, pp. 395–405.

[33] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," Pervasive Computing, IEEE, vol. 7, no. 4, 2008, pp. 12–18.

[34] V. Honkavirta, T. Perala, S. Ali-Loytty, and R. Piché, "A comparative survey of WLAN location fingerprinting methods," in Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on. IEEE, 2009, pp. 243–251.

[35] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg, "Compass: A probabilistic indoor positioning system based on 802.11 and digital compasses," in Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization. ACM, 2006, pp. 34–40.

[36] J. Letchner, D. Fox, and A. LaMarca, "Large-scale localization from wireless signal strength," in Proceedings of the national conference on artificial intelligence, vol. 20, no. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 15.

[37] A. Bekkelien, M. Deriaz, and S. Marchand-Maillet, "Bluetooth indoor positioning," Master's thesis, University of Geneva, 2012.

[38] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in Proceedings of the sixteenth annual international conference on Mobile computing and networking. ACM, 2010, pp. 173–184.

[39] H. Lemelson, M. B. Kjærgaard, R. Hansen, and T. King, "Error estimation for indoor 802.11 location fingerprinting," in Location and Context Awareness. Springer, 2009, pp. 138–155.

[40] B. Priyantha, D. Lymberopoulos, and J. Liu, "Littlerock: Enabling energy-efficient continuous sensing on mobile phones," Pervasive Computing, IEEE, vol. 10, no. 2, 2011, pp. 12–15.

[41] P. Zhou, Y. Zheng, Z. Li, M. Li, and G. Shen, "Iodetector: A generic service for indoor outdoor detection," in Proceedings of the 10th ACM conference on embedded network sensor systems. ACM, 2012, pp. 113–126.

Meta-Theorizing and Machine-Intelligent Modeling a Complex Adaptive System that is Poised for Resilience Using Architectural and Empirical Indicators

Roberto Legaspi
Transdisciplinary Research Integration Center
Research Organization of Information and Systems
Tokyo, Japan
E-mail: legaspi.roberto@ism.co.jp

Hiroshi Maruyama
Department of Statistical Modeling
The Institute of Statistical Mathematics
Tokyo, Japan
E-mail: hm2@ism.co.jp

***Abstract*—With our world witnessing critical systemic changes, we argue for a deeper understanding of what fundamentally constitutes and leads to critical system changes, and how the system can be resilient, i.e., persist in, adapt to, or transform from dramatically changing circumstances. We position our argument with long-standing theories on complexity, self-organization, criticality, chaos, and transformation, which are emergent properties shared by natural and physical complex systems for evolution and collapse. We further argue that there are system regimes that, although normally denote impending peril or eventual collapse, could actually push the system positively to be poised for resilience. In light of resilient systems, criticality and chaos can actually be leveraged by the system to promote adaptation or transformation that can lead to sustainability. Furthermore, our extensive simulation of complex adaptive system behaviors suggests that advantageous and deleterious system regimes can be predicted through architectural and empirical indicators. We framed our arguments in a two-fold complex systems resilience framework, i.e., with a meta-theory that integrates theories on complex system changes, and a machine-intelligent modeling task to infer from data the contextual behaviors of a resilient system.**

Keywords- complex adaptive systems; intelligent systems; systems resilience.

I. INTRODUCTION

We argued in [1] that we need to deepen our analysis and understanding of what makes a system resilient through a deeper understanding of what constitutes and leads to systemic changes, and how the system can be resilient through changes that are undesirable. Our world has been experiencing both slow and fast critical systemic changes on multiple levels and scales. On a global scale, for instance, Rockström and his colleagues [2] have argued that there are significant shifts happening in extremely important earth biophysical processes (e.g., climate change, freshwater and land uses, ozone depletion, and biodiversity loss), but all towards criticality. Some environmental scientists have been pointing at human activities as expediting what are supposedly naturally slow processes; hence, the debate on whether we have arrived at the Anthropocene [3]. Systemic changes also happen in the social realm when the existence of communities is significantly altered due to unprecedented massive devastations in terms of human lives, livelihoods and infrastructures brought about by natural hazards, such as Katrina of 2005, the Haiti earthquake of 2010, the triple disaster of 2011 in Tōhoku, and Haiyan of 2013, all of which brought significant human and economic losses [1]. Communities are not only evacuated, even worse, uprooted permanently as the natural environment and physical infrastructures that once supported their existence are completely destroyed. But make no mistake; it is not that our reality consists mostly of forgettable events marked by only a handful of massive devastations, but rather we hear daily the occurrence of accidents in land, air or sea, oil, chemical and radiation spills and leaks, terrorist attacks, spread of viruses, and most recently, the migration of millions of escapees and refugees crossing international borders to avoid wars, but only to find themselves enclosed by humanitarian crises. Such events can only compel our systems to carry out dramatic and novel adaptations in order for humanity to survive and sustain its existence. In the midst of critical systemic changes, our world and life systems should be *resilient*, i.e., they are able to withstand even large perturbations and dramatically changing circumstances and preserve their core purpose and integrity [4], and embrace change once transformation due to extreme perturbation is inevitable [5]. Otherwise, our systems would fail to provide the expected conditions for life to persist.

In [1], we positioned our arguments to further understand systemic changes with long-standing theories on complexity, self-organization, criticality, chaos, and transformation, all of which are interesting emergent properties shared by complex systems, and have been used to explain the evolution of complex adaptive systems (e.g., biological, natural, and socio-ecological [6]-[11]), computation by physical systems [12]-[14], and the collapse of social systems [15]-[18]. We discussed in detail our framework, supporting concepts, simulation results, and analyses. What is significantly missing, however, is further elucidation as to which aspects of systemic changes can actually push the system positively to be poised for resilience. We also need to explain further how these advantageous systemic changes can manifest themselves through architectural and empirical indicators. We extend our discussions in [1] to address these two issues.

Our paper is structured as follows. We discuss in Section II the Campbellian realistic basis, as well as the real-world application, of our complex systems resilience framework. Our framework is two-fold, i.e., with a meta-theory that integrates long-standing foundational theories of systemic change, and a two-part machine-intelligent computational

modeling, specifically, using network analysis and machine learning algorithms, to realize our meta-theory. We detail our meta-theory in Section III and highlight in Section IV how the different aspects of the meta-theory relate to how the system is poised for resilience. We then discuss in length our machine-intelligent modeling approach and simulation results in Section V, and end that section with the seemingly insurmountable challenges that our approach would face in the future. We then conclude in Section VI.

II. Our Complex Systems Resilience Framework

A. Campbellian Realism – Theoretical Basis

Donald Campbell, together with scientific realists, allied with the semantic conception theorists to replace the syntactic or axiomatic basis of theory, in Figure 1a, with its semantic conception using a *theory-model* link, in Figure 1b, wherein the axiom may or may not be necessary (indicated by the broken arrow from the axiomatic base in Figure 1b) [19]. This important aspect of Campbellian realism (other aspects are elucidated in [20]) urges scientists to coevolve the development of theory and model (as indicated by the blue bidirectional arrows in Figure 1b). Figure 1c shows our framework that conforms to Campbellian realism and the thrust of the semantic conception.

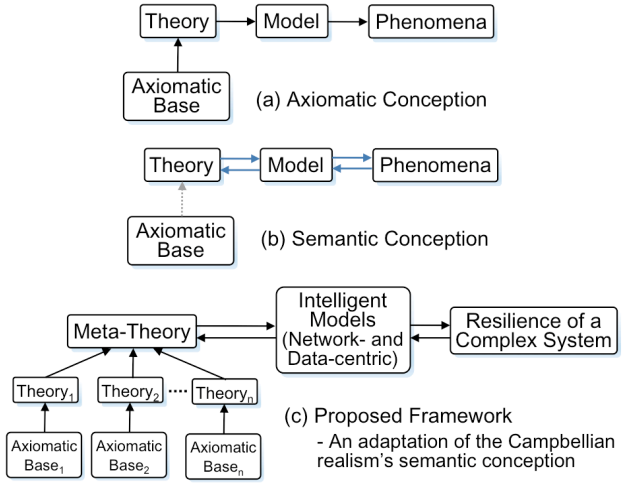


Figure 1. Conceptions of Axiom-Theory-Model-Phenomena relationship from the (a) axiomatic and (b) semantic bases [19], and (c) our linking that conforms to the thrust of the semantic conception.

We started with the Campbellian realism concept to point out that, following Campbellian realism, although the component theories of our meta-theory may have axiomatic bases, our meta-theory has no accompanying axiom. What will propel our meta-theory to becoming realistically grounded, however, is the intelligent modeling constantly updating it. Secondly, we also point out that our elucidation in Section III of the meta-theory, and the references that accompany our elucidation, would attest to the fact that the individual theories that comprise our meta-theory are neither from a vacuum nor just mere speculations as they are evident in physics, ecology, biology, or system dynamics. What we are putting for consideration, however, is a theory of how these theories are related that characterizes the resilience of a complex system. We integrated essential concepts of these theories in varying grains of analyses and view this integration as a *meta-theory*. Lastly, while the idea of Campbellian realism is to derive models manually, our modeling hinges on automatic and incremental knowledge inference using machine learning, hence the *machine-intelligent modeling*.

By starting with a meta-theory as background knowledge to guide our modeling, we avoid scattered and loosely knitted paradigms. Complementary, any truth present in the inferred models that is not accommodated in the meta-theory shall correct the flaw in the meta-theory. Our meta-theory and machine-intelligent models can therefore evolve together with increasing "predictive isomorphism" [19, p.7] to accurately represent the phenomena that are endogenous and exogenous to the system. We believe that this mutual reinforcing of meta-theory and intelligent modeling to automatically characterize the contextual interaction behaviors of a resilient system is novel and is not found in the more established frameworks, such as the Adaptive Cycle [9], Self-organized Criticality [8], and Dual-Phase Evolution [11].

Our machine-intelligent modeling consists of two parts, namely, network analysis and machine learning. Network theory concerns itself with the order and patterns that emerge from the self-organization of complex systems than with elucidating the underlying mechanisms by finding simplified mathematical engines [21]. The intractable nature of complex adaptive system behavior significantly prohibits the application of mathematical formulation since it would only result to futility, e.g., several researchers have addressed the fact that the formal models used to study the resilience of socio-ecological systems do not explicitly include the internal structural characteristics of these systems that are in constant interaction [22]-[24]. The tendency of formal models is to abstract many of the system's internal workings [25]. Furthermore, network theory concerns itself with system phase transitions wherein the processes of adaptation and transformation are possible [21].

We employ machine learning to automatically discover hidden relational rules that can describe the emergent system behaviors that are indicative of resilience. Machine learning algorithms can detect hidden behavior patterns in the data, which the system can use to understand its resilience capability and adjust its behavior accordingly. Our framework is data-centric as opposed to using formal verifications. Again, we can argue that formal or mathematical verification does not always guarantee reality and is not absolutely reliable. It can even fall short given the computational intractability of complex systems. The intractability of a complex system state space leads to issues of big data, which is where machine-learning inference becomes viable. Furthermore, and again as above, formal models tend to abstract much of the realistic nonlinear and stochastic intricacies of the system's internal workings [25].

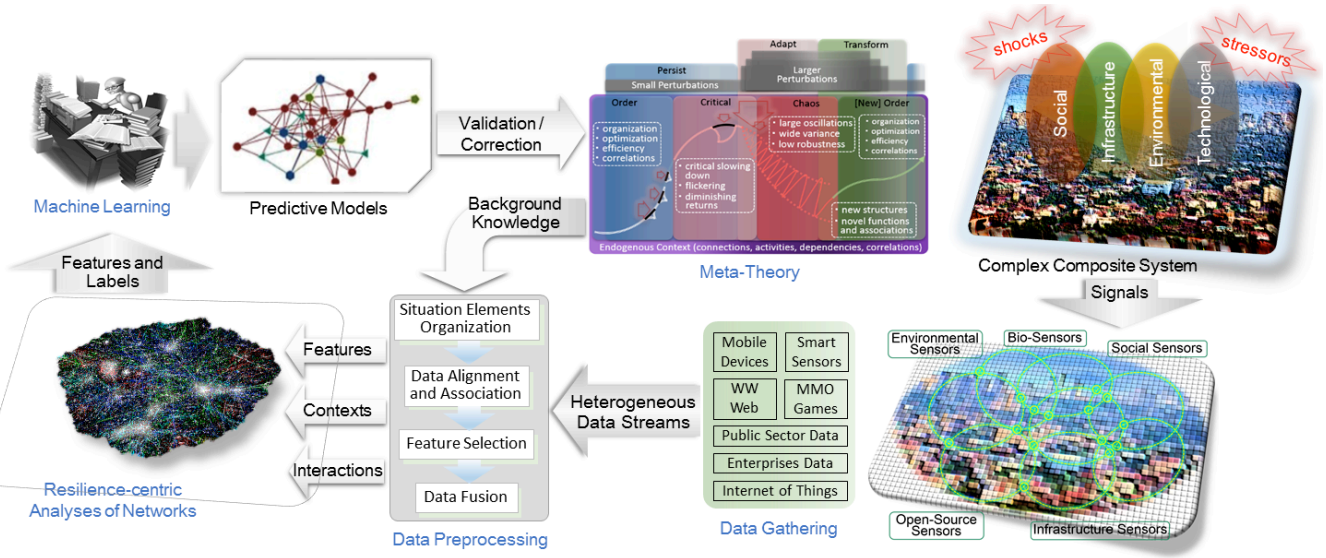


Figure 2. Our entire complex systems resilience modeling architecture, which includes our two-fold framework.

B. *Application of the Framework*

When we speak of complex system properties, we speak of system-wide behaviors emerging from the interaction and interdependencies of diverse system components. To be more concrete, our long-term objective is to model the complex hyper-connections of our social, infrastructure, environmental and technological systems, as shown at the right side of Figure 2, where system components, which can be composite systems in themselves [26][27], are intricately connected and may display extreme dependencies. This complex system-of-systems contains continuous flow of information, energy, capital, and people, among other resources. The resilience of any component will critically depend on its place in the system and how it, and the entire complex system, can withstand perturbations.

The meta-theory can be viewed as by-product of integrated transdisciplinary perceptions of what characterizes complex systems resilience. Carpenter et al. [28] suggest that to account for uncertainties in complex systems, we must consider a wide variety of sources of knowledge and stimulate a diversity of models. They also suggest that the tendency to ignore the non-computable aspects of complex systems can be countered by considering a wide range of viewpoints and encouraging transparency with regard to conflicting perspectives. They emphasized that there are instances where expert knowledge may not suffice since they can demonstrate narrow and domain-dependent practices. They went on to provide evidences where the perceptions of local people, who are experience-filled individuals, led to breakthroughs. Knowledge engineering approaches can be used to build and maintain knowledge-based systems that capture relevant contributions based on expertise and experience. We can also develop knowledge representation, extraction, inference and integration technologies that can infer relationships that exist among knowledge from largely varying domains and can synthesize individualized, micro-level, and domain-dependent knowledge towards contextual systemic knowledge that can lead to actionable information for resilience. Such actionable information, for example, can be in the form of a repository of evidences of what works (predictive) and may work (innovative) in a given situation (e.g., disaster management).

To gather large amount of data to model the complex system-of-systems, ubiquitous smart and interacting daily-living objects can offer a wide range of possibilities [26][29]. The World Wide Web is an open world and quintessential platform for us to share and receive information of various kinds. Web contents are created and duplicated rapidly and continuously. Crawlers or scrapers can be written to extract data stored deep in the Web. Our mobile devices have become ubiquitous in our lives that we rely on them for communication and information, keeping them within reach so that we can check them, at times unconsciously, every few minutes. But our mobile devices also have powerful sensing, computing and communication capabilities that allow us to log our daily activities, do web searches and online transactions, and interact on social media platforms and micro-blogging sites, among others. Ubiquitous and interacting ambient sensors [26][29] can gather large volumes of human- (e.g., individual mobility, physiology and emotion signals, crowd or mass movements, traffic patterns) and environment-related (e.g., climate and weather changes, changing landscapes and topographies, light and CO_2 emissions) data. Tiny interacting embedded systems could also play a valuable role in protecting the environment from hazards, e.g., sensors so minute, as the size of dust particles, but can detect the dispersion of oil spills or forest fires [26]. There are also the massively multiplayer online games (MMOGs) that have become unprecedented tools to create theories and models of individual and group social and behavioral dynamics [30], which might shed some light on human resilience behavior. There are data that the public sector produces, which include geographical information, statistics, environmental data, power and energy grids, health and education, water and sanitation, and transport. There are

the systematically acquired and recorded census data about households and the services made available to them (e.g., health and medical, education, water, garbage or waste disposal, electricity, evacuation, and daily living-related programs). Enterprises (corporations, small businesses, non-profit institutions, government bodies, and possibly all kinds of organizations) may collect billions of real-time data points about products, resources, services, and their stakeholders, which can provide insights on collective perceptions and behaviors, as well as resource and service utilizations. And lastly, there is the Internet of Things (IoT) that extends the reach of the Internet beyond our desktops, mobile phones and tablets to a plethora of devices and everyday things (e.g., wearable and ambient sensors, CCTVs, thermostats, electric power and water usage monitors, etc.). Data can be made available online and publicly through the IoT, and therefore democratized, i.e., accessed freely for the common good. Hence, our digital universe is ever expanding as millions of data points are continuously created by and acquired from heterogeneous sources. Machine intelligence can be used to infer from these massive data points accurate informative models for situation analysis and awareness, decision-making and response, and component feedback. All these to aid the complex system sense and shape the contexts in which it is embedded.

Heterogeneous data related to humans, infrastructures, environments, and technologies, and their interactions will often be reported or obtained from a multiplicity of sources, each varying in representation, granularity, objective, and scope. Preprocessing techniques can be used to organize, align, and associate input data with context elements. With feature selection, it can also reveal which features can improve concept recognition, generalization and analysis. Lastly, data fusion can address the challenges that arise when heterogeneous data from independent sources are combined.

All the pertinent features, contexts and interactions inferred in the preprocessing stage will be used in our two-part machine-intelligent modeling. First, information will be organized, represented and analyzed as a network. Paperin et al. [11] provide an excellent survey of previous works that demonstrated how complex systems are isomorphic to networks and how many complex properties emerge from network structure rather than from individual constituents. One may think of the human body and brain, local community, virtual community of socially related digital natives, banking systems, electric power grid, and cyber-physical systems as networks. Furthermore, the science of complexity is concerned with the dynamical properties of composite, nonlinear and network feedback systems (citations in [31]). Second, using as inputs the network and resilience properties of the system, machine learning will be used to infer the relational rules of system contextual interaction behaviors that define its adaptive and transformative walks and therefore define its resilience. Our modeling will capture how the complex system's ability to vary, adjust or modify the connectivity, dynamism, topology, and linkage of its components (endogenous features), and its capacity to withstand perturbations (exogenous feature), can dictate its resilience.

III. OUR META-THEORY

Figure 3 shows our meta-theory that cohesively puts together complexity, self-organization, critical transition, chaos, resilience, and network theories. While we adopt the terms order, critical, and chaos from dynamical systems theory [32], to persist, adapt, and transform is resilience thinking [33]-[35]. Scheffer et al. [36] proposed integrating the architectural, i.e., the underlying network configuration, and empirical indicators of system phase transition. They suggested that since these two approaches have been largely segregated, a framework that can smartly unify them could greatly enhance the capacity to anticipate critical transitions. Although Scheffer's primary concern in [36] is the critical transition, we adapt these two approaches to observe complex system behaviors. While the top layer of our meta-theory specifies the empirical indicators, the bottom layer specifies the network configuration-based indicators.

A complex system can be highly composite, i.e., it can consist of very large numbers of diverse components, which can also be composites in themselves, and these components are mutually interacting with each other. Their repeated interactions over time eventually leads to a rich, collective behavior, which in turn, becomes a feedback to the individual components [37]. Self-organization holds that structures, functions, and associations emerge from the interactions between system components and their contexts. For Levy [21], the most appealing and persuading aspect of complexity theory is its promise to elucidate how a system can learn more effectively and spontaneously to self-organize into well structured, sophisticated forms to better fit the constraints of its environment.

The complex system evolution cycle in our meta-theory involves three regimes, namely, order, critical, and chaos. The second ordered regime, however, could be novel in the sense that it required the system to transform when adaptation back to the previous state was no longer attainable. The moving line at the top layer indicates system "fitness", i.e., the changing state of the system in terms of its capacity to satisfy constraints, efficiency and effectiveness in performing tasks, response rate (time to respond after experiencing the stimuli), returns on its invested resources or capital, and/or its level of control. The fitness curve may indicate growth (e.g., exponential, i.e., an initial quantity of something starts to grow and the rate of growth increases, or *s*-shaped, i.e., an initial exponential growth is followed by a leveling off), degression (gradual) or quick descent, or oscillation where the fitness fluctuates around some level.

This section discusses in detail the various aspects of our meta-theory. We want to believe that through our meta-theory we can view a complex system as *open*, i.e., always in the process of change and actively integrating from, and disseminating new information to, changing contexts, as well as *open-ended*, i.e., it has the potential to continuously evolve, and evolve ways of understanding and manipulating the contexts (endogenous and exogenous) that embed it [38]. Both characteristics are vital for the resilience of the complex system.

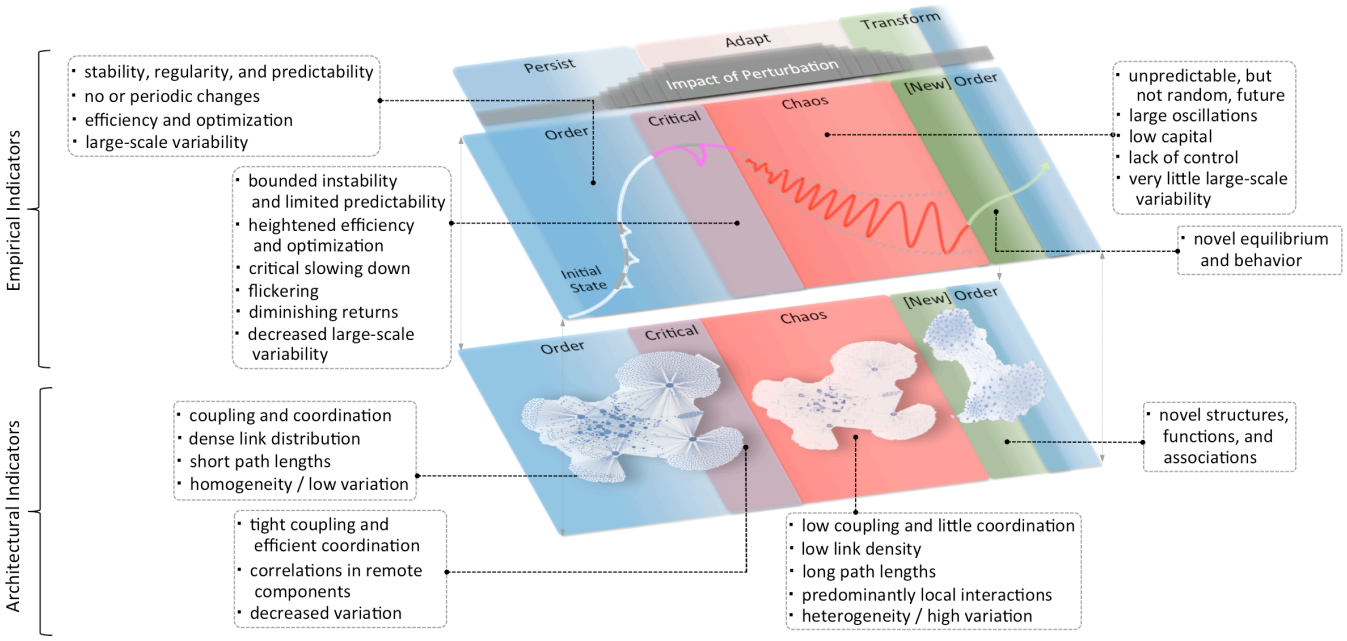


Figure 3. We integrate in varying grains of analyses how the different theories are plausibly related – hence, a *meta-theory*.

A. Order

It is in the ordered regime that dependencies and correlations begin to emerge in the structural and logical connections of the system components. The components become coupled and coordinated. Eventually, the system will settle into a regular behavior, i.e., a state of equilibrium. It is also possible for a system to have multiple feasible equilibriums wherein it shifts between equilibriums. Its components have high degrees of freedom to interact with each other that it can have many possible trajectories [39].

The system will always attempt to establish equilibrium each time it is perturbed (as illustrated by the dents along the fitness line at the top layer) in order to persist in its ordered state. Perturbations are assumed to be largely, albeit not totally, identifiable and unambiguous. When it encounters a perturbation, it will aim to resume normal operations as soon as possible. The system will always control and manage change, with its agent components acting in accordance to an accepted set of rules. The system will act in predictable ways, either executing once again previous behaviors or selecting from its known limited range of behaviors with anticipated or foreseeable results [31]. The system will operate in a negative feedback manner, with the appropriate rules, to reduce fluctuations and maintain its regular predictable behaviors [31]. Hence, its success is measured in terms of stability, regularity and predictability [40].

The ordered regime is also characterized by increasing system efficiency and optimization of processes. The system will carry out its tasks efficiently as possible according to the well-defined structural and logical connections of its components and policies and procedures it strictly adheres to. The system's self-regulation becomes optimized specifically to the set of perturbations and responses it already became familiar with.

B. Critical

In highly coupled systems, the iterative recovery from small-scale perturbations give the illusion of resilience when in fact the system is transitioning to a critical change and setting itself up for an unwanted collapse [36]. The coupling among components has become tight to the point where the order of the system becomes highly dependent to the strong coordination of its parts. All this build-up, however, is like an accident in the wings waiting to happen. This rigid tight coupling makes the impact of any perturbation to also increase, regardless of whether its magnitude is small or large. One situation, even though stirred by a small perturbation, can easily become critical and can trigger other events in a cascading fashion such that the different situations within the propagation enhance themselves to criticality. As one of the bedrocks of complexity science, complex adaptive systems have the tendency to move towards criticality when provoked with complexity [40].

As a real world example, Lewis [41] cited several reasons why the electrical power grid can self-organize to criticality due to heightened complexity. These include the increase of components' reliability that consequently increases the cascade of failures and its consequences, optimization of the grid by power stations and centralizing substations, tight coupling of hubs in telecommunications networks, and the simultaneous occurrence of stable load increase as more people consume more electricity, electricity providers maximize profit, and maintenance procedures become more efficient. Also, when Levinthal [42] applied random Boolean networks to simulate the adaptation of

business organizations to their environment, he found that tightly coupled firms find it hard to adjust to changes.

It is also the case, however, that one of the profound insights from the science of complexity is that this regime – that is poised between stasis, where there is no or regular changes, and chaos, where changes are irregular – holds significant paradoxes. It is neither stable nor unstable, but both at the same time [31]. It is both optimal and fragile [39]. It may herald an unwanted collapse and become a harbinger of positive change [36]. Furthermore, while it may signal hidden fragilities [43], it is also theorized to facilitate complex computations, maximize information storage and flow, and be a natural target for selection because of its hidden characteristics to adapt [13][6][14].

C. Chaos

Comes a point when complexity can no longer be sustained, persistence is no longer possible, and predictive adaptations are not anymore sufficient. Eventually, the system converges to a state that makes itself less adaptive to perturbations and moves to chaos. The building up of complexity becomes a constraint to adaptation and eventually leads to chaos.

Chaos denotes a state of non-equilibrium, thus, instability, and turbulent, aperiodic changes that lead to crisis, disorder, unpredictable outcomes, or, if on a large scale, to collapse. The notion of chaos has been used interchangeably, or in association with, several other concepts, such as non-linear systems models and theories on disorder, dynamical complexity, catastrophe, bifurcation, discontinuity, and dialectical dynamic, among others (refer to [44]).

When in chaos, the system would need larger adaptations if only to survive. The system must learn how to minimize the negative effect of chaos and maximize its positive properties, which is a compelling and yet to be fully addressed key problem of social and natural scientists [44].

D. New Ordered Regime through Transformation

When chaos happens, the once tight connections and rigid coordination are broken. This then becomes an opportunity for the system to try other, perhaps novel, connections that can lead to positive transformation. Systems may undergo a transformational process, as it is provoked by instabilities, potentially leading to an emergent order that is different from its previous ordered state [31]. Systems that demonstrate a transformative capacity can generate novel ways of operating or novel systemic associations and can recover from extreme perturbations [5]. Such systems learn to embrace change [5], and instead of bouncing back to specification that has been proved vulnerable and led to chaos, they bounce *forward* to a new form [45].

IV. META-THEORIZING A COMPLEX SYSTEM THAT IS POISED FOR RESILIENCE

We posit in this section that in the critical and chaotic regimes the system can be poised for resilience. We also discuss here the different architectural and empirical indicators of system changes.

A. Critical Transition that is Poised for Resilience

1) Intituition Behind the Concept

The intuition is both appealing and intriguing: systems that are highly stable are static and those that are chaotic are too unstable to coalesce, and thus it is only at the border between these two behaviors that the system can perform productive activities [46][40]. Another bedrock principle of complexity science is that complex adaptive systems are at risk when in equilibrium, and that this stasis is a precursor to the system's death [40]. In cybernetics lingua, competing pressures must perturb the system far away from its normal arrangements before it can significantly evolve to a new form. This state of being far away from equilibrium but not in chaos has been called by several names, including the edge of chaos [46] or instability [31]. This edge is not sharp and unambiguous, but rather, it is like overlapped coatings with bidirectional gradation between order and chaos.

According to Miller and Page, "In its most grand incarnation, the edge of chaos captures the essence of all interesting adaptive systems as they evolve to this boundary between stable order and unstable chaos." [46, p. 129] The proponents of this condition think of it as holding "the secret of everything from learning in the brain to the evolution of life" [21, p. 73]. Similarly, Pascale stated that as systems continue to self-organize, they "all flourish in a boundary between rigidity and randomness and all occasionally forming large structures through the clash of natural accommodation and competition." [40, p. 3] For instance, Krotov et al. [47] hold evidence to suggest morphogenesis at criticality in the genetic network of early Drosophila embryo.

Stacey proposed that at the critical transition the outcomes can be indeterminate, or what he calls bounded instability [31]. His notion is that although the system behavior cannot be predicted over the long-term, hence the presence of instability, there is qualitative structure in the system's behavior that is recognizable and that short-term outcomes can be predicted, hence bounded. He stated that it is in the bounded instability that the complex system becomes changeable and its behavior patterns are in unpredictable variety. Stacey also stated that the agents are not constrained by their rules, schemas and scripts, but by the freedom they have to choose their actions within these constraints that will have major consequences for the system. Similar to the edge of chaos, in bounded instability, a system is far easier to adapt because small actions by any of the agents can escalate into major system outcomes [31].

2) Indicators of Critical Transition

A broad range of research has looked at connectivity and variation (from homogeneous to heterogeneous) of network components as what constitute the architecture of fragility [36]. Variation refers to the actually existing differences among individual system components in terms of type, structure or function [48]. According to Rickels et al. [37], a system is at critical point when the degree of connectivity and dependence among the components is extremely high. For instance, in the investigation of Krotov and his colleagues, criticality manifested itself as patterns of correlations in gene activity in remote locations [47][49].

The system enters criticality as its components become more and more coupled. As a consequence, a small perturbation in the system can lead to massive systemic changes. Scheffer et al. [36] stated that a network with low connectivity and heterogeneous components has greater adaptive capacity that enables it to change gradually, as opposed to abruptly, in response to perturbations. At the same time, a network with tightening couplings and heightening homogeneity of its components can only manage to resist change up until a certain threshold where critical transition is reached. According to Page [10], the amount of variation is low at stasis and high when the system is in flux. Page pointed out both the obvious and deep insight to this: it is obvious that there is more variation when the system has yet to settle down; however, it also means that there is more variation in a system that is about to transition since the used to be stable configurations found difficulty holding together.

Scheffer et al. [36] explained the various empirical indicators of critical transition. One is critical slowing down, i.e., the rate at which the system bounces back from small perturbations becomes very slow, which makes it more vulnerable to be tipped more easily to another state. Critical slowing down can be inferred indirectly from rising variance and correlation (e.g., higher lag-1 autocorrelation). Another is flickering wherein a highly stochastic system flips to an alternative basin of attraction when exposed to strong perturbations. Rising variance is also indicative of such a change, as well as the multimodality of system state frequency distribution over a parameter range. Scheffer et al. also stated that while critical slowing down may point to an increased probability of an abrupt transition to a new unknown state, flickering suggests an opposite regime to which the system may transition into if conditions change.

Page [10] elaborated in his book why diminishing return is an empirical indicator of criticality. He described diminishing returns as the decrease in some system performance measure such as efficiency, accuracy or robustness. For example, as Lewis pointed out, lessening of reactive power, transmission capacity, and information in the grid indicates that the grid is in a critical phase [41]. Furthermore, according to Dixit, the tension between increasing and diminishing returns would likely result to the self-organized criticality of economic systems [78].

Lastly, there is also variability as empirical indicator. Variability describes the potential or propensity to change (e.g., variability of a phenotypic trait in response to environmental and genetic influences) [48]. It implies rules that can lead to periodic or aperiodic dynamics. While connectivity and variation can be directly observed, according to Wagner and Altenberg, variability is harder to measure due to its "dispositional nature" [48]. They used the concept of solubility as akin to variability being dispositional, i.e., it does not describe the current state the substance is in, but rather the behavior that results when a substance gets in contact with enough solvent. In Paperin et al.'s Dual-Phase Evolution [11], order is described as a well-connected phase that is characterized by highly dense link distributions, short path lengths and wide-scale interactions between most system components – hence tight coupling, with little local variation and high large-scale variability. The poorly connected phase that is akin to chaos is quite the opposite, i.e., the link density is low, path lengths are long, with mostly within sub-network (local) interactions, and strong local variation but little large-scale variability.

3) Critical Transition To Resilience

Although there is still a lot to be done in terms of anticipating critical transitions, even though methods and approaches are already emerging, there are works that suggest leveraging critical transitions for the resilient walks of the system. For one, critical systems by nature will move toward, rather than away from, the precipice of collapse [41]. Bak et al. [8] asserted that a system collapse is inevitable mainly because of the internal dynamics rather than from any exerted exogenous force. This is in line with the classical approaches within evolutionary biology that view organisms as simply passive objects that can be controlled by internal or external forces, and that these forces are beyond the ability of the organisms to influence, let alone surmount [50]. Dialectics argue, however, that organisms can also be subjects of their own evolution [50]. Page [10] posited that this critical transition is not a system state, but rather, this border is in the space of system behaviors, i.e., the set of the agent's decision rules and interaction scripts [46]. Hence, at the critical transition, a complex adaptive system can have the ability to tune its rules and scripts towards resilience (or vulnerability). The notion of "edge" of chaos or instability can be that narrow but sufficient space where the adaptive system has the ability to see qualitative structures and relations, and can predict sufficiently, even if for a bounded distance, and change its course for the better. The system can utilize both amplifying and dampening feedbacks to flip itself autonomously from one equilibrium to the next and not be pinned down to just one.

In [36], Scheffer et al. discussed how the different architectural features and empirical indicators that enhance criticality could actually offer provisions for diagnosis and potential intervention. For instance, since it can be predicted that a network with low diversity and high connectivity is positioned for critical transition, the potential response could be to redesign the system for more gradual adaptive response or further strengthen the preferred state. Furthermore, as it can be predicted that critical slowing down can elevate chances of critical transition and that flickering can increase the probability of tipping to alternative states, in both cases, the system can get ready for the anticipated change, lessen the risk of unwanted transition, or leverage the opportunity to promote the desired transition since the system is more open to change. Again, on the upside, a tightly coupled system makes it possible for tiny interventions, perhaps undetectable and hard to quantify, to escalate into major qualitative interventions that can alter the course of the system's life [31]. For instance, in a biological perspective, this is akin to a protein or a neuron firing so as not to self-organize to criticality [49].

Lewis [41] provided real-world suggestions on how the system can "un-SOC" itself – since self-organizing criticality (SOC) will eventually lead to collapse, the system can extend its life by *un*doing the SOC process. In other words,

allow the system to loosen the connections and modularized or decentralized its components, and make its processes less efficient and suboptimal. He suggested the policy of link depercolation to keep SOC under control, i.e., prevent a node from having too many connections or thin out current links, or reduce the links of hubs. As Casti [18] also pointed out, the only realistic alternative is to loosen up the tight coupling of the components. Since the decoupled dynamics constrain stimuli locally, the entire system becomes more robust in the face of perturbations [11]. In the wake of a pandemic, for example, if a vaccine has been proved to kill the virus, high connectivity and interdependencies among agents will greatly aid the inoculation process. However, if the population is too massive to inoculate, or no vaccine has yet been discovered to cure the population, depercolation, e.g., by quarantine or limiting the contact of people, will be the next best solution [41]. Lewis also suggested reducing operational efficiency in different sectors, e.g., energy, transportation, and telecommunication, since these work better when they are less efficient and more decentralized.

To end this section, we echo what Casti stated in his book, specifically, "sustainability is a delicate balancing act calling upon us to remain on the narrow path between organization and chaos" [18, p. 46]. The idea, therefore, is for the system to stay away from the steady state to remain flexible [21] and be close to chaos while retaining some degree of order.

B. Creative Chaos that is Poised for Resilience

1) Intituition Behind the Concept

In an article written by M. Fisher for The Atlantic [51] is an articulate description of how the pre-war Japanese ideology transited from rigidity, to collapse, and to the emergence of a totally new ideology that brought about the reorganization of a country – a transformation that has affected the world even today: For many years prior to the end of the World War II, the then Japanese citizenry had been embedded in an ideology of imperialism, ultra-nationalism, radical militarism, and international primacy. Such rigid ideology drove the country to a quest of imperial expansion, which at the beginning marked Japan's military strength and dominance in the region. Towards the end, however, when Japan's defeat in the international scene became inevitable and the devastation brought upon it was immeasurable, the people feared that with the rigid ideology, where surrender was not an option, they would be forced to choose death over imperial ideology. What the Emperor spoke in those critical moments when their survival as a nation hanged in the balance, however, was different. They were asked instead to choose the radical alternative – embrace the surrender towards a noble change, i.e., one of moral integrity, nobility of spirit, peace and international progress. This marked the collapse of the ideology that was once held unbreakable. The accompanying suffering was indeed enormous (in [79]), but the result after a generation was a nation of renewed identity that emerged to become one of the great leaders of our modern economic and technological progress.

When allowed to progress in complexity and rigidity, a system would eventually collapse. However, in an inevitable collapse, the system can open itself up to possibilities to become a new and better system, if only adaptive and not to maladapt in the midst of chaos. Holling describes the chaotic phase in ecological systems, which he refers to as the backloop of his Adaptive Cycle [52][9], as the sudden release of complexity, characterized by significant decrease in capital and loss of connection among parts. When this happens, however, the system begins to open itself up to novel forms, functionalities, and systemic associations [56]. For example, an essential part of the forest ecosystem is the occurrence of natural fire since it replenishes soil nutrients, allows new plant species to grow, and reduce pathogens and infestations, among others [56]. In evolution, although deleterious mutation is assumed to inject harm and impede adaptive evolution, it also has the potential to evolve complex new functions (refer to [53]). One example in the technology sector, albeit not with a happy ending, is Kodak, as recounted in [54]: From being a worldwide market leader in film photography in the 80s, Kodak collapsed to bankruptcy in 2012. The notion that Kodak collapsed because it missed the rise of digital technology is untrue – the engineers at Kodak have already developed the technology in the 70s. Rather, the management was deceived by its very comfortable position and large profit margins in the film market that it did not want to take the risk of investing in the new products despite the various internal red flags. When Kodak eventually turned to digital photography, it was not anymore an open space for innovation since competitors had already filled the space. Hence, when chaos ensued, there was no room for Kodak to transform; consequently, its total collapse. Kodak failed to rise beyond the innovator's dilemma [41][80] – it stubbornly followed its star technology to its peak (and eventual demise) instead of risking everything for the next big thing.

2) From Chaos to Resilience

Schumpeter's creative destruction theory [55] states that a continuous uninterrupted unpleasant transformation from within that destroys the old one also incessantly creates a new one. If progress means turmoil, then why not accelerate the turmoil if only to also accelerate getting to the new and better progress. Similarly, in [56] we suggested the strategy of deliberately injecting or inducing regulated and controlled shocks into the system as complexity and rigidity among its components begin to build up: This is in a way forcing the system to transition itself to chaos in order for novelty to emerge. The motivating principle behind this strategy is to let the system embrace the fact of an inevitable failure and learn how to deal with it swiftly once it happens. It is more effective to create situations that can force latent systemic problems to surface and become visible, rather than design the system not to fail, which, paradoxically, only makes it less resilient. This strategy will certainly cause disorder and crisis in the system, to say the least, but such will last relatively shorter than if chaos was actually not staged.

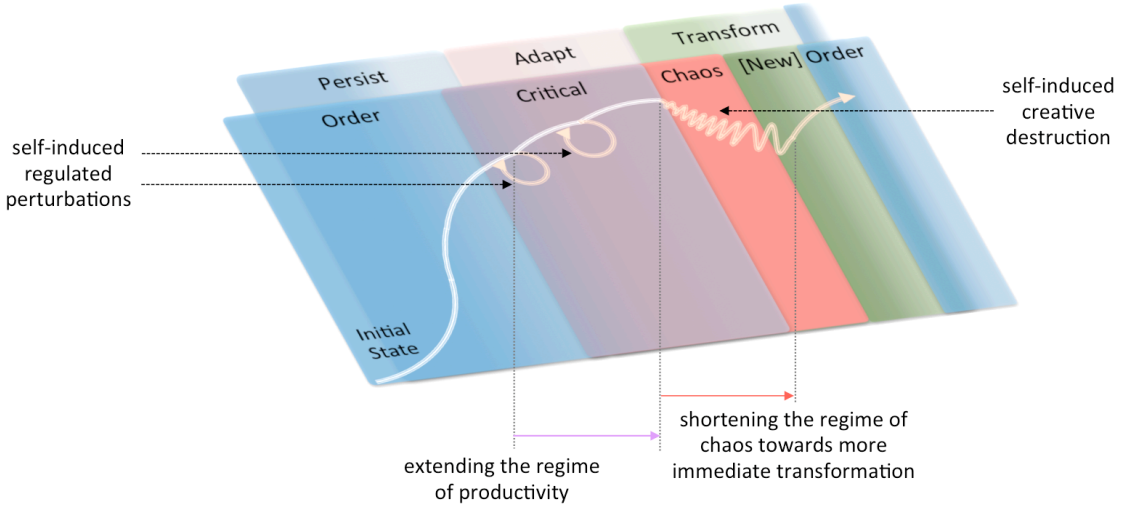


Figure 4. For a complex system to be poised for resilience, it must be able to promote its own desired transition.

C. *System Being Self-Poised for Resilience*

Merriam-Webster dictionary defines "poised" as "in a state, place, or situation that is between two different or opposite things", as well as being "ready or prepared for something". The critical regime is poised between order and chaos marked with complexity and decreased fitness. And yet, this complexity does not destroy the ability of the system to self-organize. With optimal connectivity and coordination they have enough stability to store and propagate information, as well as the fluidity to productively adapt based on the received information. At the same time, a system in the state of chaos has lost most much of its stored information and connectivity, and yet, has become ready to create new associations and transmit improvised information.

Given the above, a system that has the ability to detect through architectural and empirical indicators the critical state can promote the desired transition by (*a*) extending the edge of productivity through necessary adaptations that involve regulated self-induced perturbations (e.g., manageable reorganization or reconfigurations), or (*b*) reducing the risk of unwanted transition by imposing upon itself the self-induced creative destruction that can lead to shorter chaos and groundbreaking innovation. Figure 4 shows our meta-theory for a complex adaptive system's self-imposed transitions to achieve resilience. Notice how criticality is extended while chaos is shortened.

V. MACHINE-INTELLIGENT MODELING

A. *Simulation of a Complex System and its Properties*

Although our aim is to model a real-world complex adaptive system and its intricate properties, as per Figure 2, our major concern at this time, however, is that we have yet to embark on this endeavor. In order to demonstrate our concepts, we used random Boolean networks (RBNs) to simulate the properties of a complex adaptive system. RBNs have been used as models of large-scale complex systems [57][58]. These are idealizations of complex systems where systemic elements evolve [59]. RBNs are general models that can be used to explore theories of evolution or even alter rugged adaptive landscapes [60]. Furthermore, although RBNs were originally introduced as simplified models of gene regulation networks [61][6][7], they gained multidisciplinary interests since they contribute to the understanding of underlying mechanisms of complex systems even though their dynamic rules are simple [62], and because their generality surpassed the purpose for which they were originally designed [20][62]-[64]. By using RBNs, we were able to analyze complex system behaviors and describe the viability of our framework.

A RBN consists of N Boolean (1 being on/active and 0 as off/inactive) nodes, each linked randomly by K connections. A RBN can be viewed as consisting of N automata with only two states available per automaton [65]. N represents the number of significant components comprising an adapting entity, generally, the number of agents attempting to achieve higher fitness [20]. We can view K conceptually as affecting the mutual influence among nodes in an information network [62] since a directed edge $<x, y>$ means that agent y can obtain information from, and can be influenced by, agent x. In this way, K is proportional to the quantity of information available to the agent [62]. The Boolean values may represent, for example, contrasting views, beliefs and opinions, or alternatives in decision-making (e.g., buying or selling a stock [7], cooperating with the community or not). The state of any node at time t+1 depends on the states of its K inputs at time t by means of a Boolean function that maps each of the 2^K possible input combinations to binary output states. The randomly generated Boolean functions can be represented as lookup tables that represent all possible 2^K combinations of input states.

Given N and K, there can be 2^N network states, $(N!/(N-K)!)^N$ possible connectivity arrangements, $(2^{2^K})^N$ possible N Boolean function combinations, and $((2^{2^K}N!)/(N-K)!)^N$ RBNs [66]. This is not counting the many possible updating schemes [60], and possibly extending to have nodes with multiple states [67]. With this huge number of possibilities, it

is therefore possible to explore with RBNs the various properties of even large-scale complex systems and their many possible contexts [58].

As RBNs are systems with information flowing across parts, network theory can be used to define the properties that characterize the configuration of a RBN. These properties can be viewed as the controlling variables that the system can modify or adjust to demonstrate its resilient capabilities. In a plausible sense, these can also be viewed as the *simulated* outputs of the pre-processing stage of our framework (as per Figure 2) that led to the configuration of the network. The parameters are as follows:

- *Connectivity* (*K*). This refers to the maximum or average number of nodes in the input transition function of a network component. As we increase *K*, nodes in the network become more connected or tightly coupled, and more inputs affect the transition of a node.
- *Dynamism* (*p*). A Boolean function computes the next state of a node depending on the current state of its *K* inputs subject to a probability *p* of producing 1, and a probability of 1-*p* of producing 0, in the last column of the lookup table [81][60]. If *p*=1 or *p*=0, then there is no actual dynamics, hence low activity, in the network. However, *p* close to 0.5 gives a high dynamical activity since there is no bias as to how the outputs should be [60].
- *Topology* (or *link distribution*). A RBN may have a fixed topology, i.e., all transition functions of the network depend on exactly *K* inputs, or a homogeneous topology, i.e., there is an average *K* inputs per node. Another type of topology is scale-free, where the probability distribution of node degree obeys a power law. In an information network, a scale-free property means that there is a huge heterogeneity of information existing [62], hence, there is more variation in the network. Following [68], the number of inputs for the scale-free topology is drawn from a Zeta distribution where most nodes will have few inputs, while few nodes will have high number of inputs. The shape of the distribution can be adjusted using the parameter γ (we set initially to 2.5) – when γ is small/large, the number of inputs potentially increases/decreases.
- *Linkage* (or *link regularity*). The linkage of a RBN can be uniform or lattice. If the linkage is uniform, then the actual input nodes are drawn uniformly at random from the total input nodes. Following [38], if the linkage is lattice, only input nodes from the neighborhood (i-$lattice_i$*k_i):(i+$lattice_i$*k_i) are taken, where i is the position of the node in the RBN and $lattice_i$ is its lattice dimension whereby nodes are dependent to those in the direct neighborhood. A wider lattice dimension can lead to a RBN with highly interdependent nodes.

While the above properties indicate the architecture that underlies system regimes, we add another property that serves as empirical indicator of upcoming transition. We measured the *robustness* of the system when faced with perturbations. Note as well that the difference in the number of Boolean functions not only contributes to the variation in variable transitions, but also influences the variability of information flow in the network. Hence, we *combined both architectural and empirical indicators of system regimes*.

Using the BoolNet package [82] for the R programming language for statistical computing, we applied the program of Müssel et al. as outlined in their BoolNet vignette [68] as follows. A perturbation is achieved through random permutation of the output values of the transition functions, which although preserved the numbers of 0s and 1s, might have completely altered the transition functions. For each simulation, a total of 1,000 perturbed copies of the network were created, and the occurrences of the original attractors in the perturbed copies were counted. Attractors are the stable states to which transitions from all states in a RBN eventually lead. The robustness, *R*, is then computed as the percentage of occurrences of the original attractors.

It is very important to realize that robustness here is *not* resilience per se, since resilience refers to *what* enables a system, i.e., change in connectivity, dynamism, topology, and linkage, to preserve its core identity when faced with perturbations [4]. We used *R to quantify the amount of RBN core identity that was preserved*. Hence, *R* is an indicator or measure of systems resilience.

B. Identifying the System Regimes

We started by identifying the system regimes given the RBN properties we specified above. To achieve this, we began our simulations with a base case. Our *base case* is a "conventional" RBN, i.e., the topology is fixed and the nodes are updated at the same time by the individual transition functions assigned to each, i.e., synchronous update. With several conditions to check, we used for now a single value for *N*, which is 20. We computed for the robustness of various RBNs in a dynamism-connectivity plane, i.e., how a RBN with specific dynamism and connectivity values is robust after 1,000 different perturbations. Figure 5 shows our base case *R*-matrix in a dynamism-connectivity space where each component is *R*-value.

R (robustness) matrix

dynamism (*p*) \ connectivity (*K*)	1	2	3	4	5	6	7	8	9	10
0.9	48.700	34.067	31.300	55.300	2.050	0.000	1.900	1.680	0.000	6.267
0.85	49.000	29.075	15.560	4.033	0.000	0.000	1.289	0.000	0.000	6.150
0.8	48.900	26.500	17.180	3.567	0.000	0.000	1.800	0.000	0.000	6.138
0.75	48.200	26.400	14.460	5.033	0.000	0.000	1.444	0.000	0.000	6.288
0.7	49.600	26.875	15.360	3.433	0.000	0.033	1.389	0.000	0.000	6.113
0.65	49.600	24.425	12.340	4.033	0.100	0.000	1.767	0.000	0.000	5.913
0.6	53.800	27.050	13.900	3.933	0.000	0.033	1.611	0.000	0.000	5.813
0.55	49.100	26.100	15.780	4.533	0.000	0.000	1.456	0.000	0.000	5.913
0.5	48.000	26.050	13.280	4.667	0.100	0.000	1.367	0.000	0.000	6.088
0.45	49.400	30.100	16.160	4.267	0.000	0.033	1.689	0.000	0.000	6.275
0.4	49.100	22.575	15.380	3.467	0.100	0.033	1.289	0.000	0.000	5.913
0.35	50.900	26.775	15.820	4.600	0.000	0.000	1.489	0.000	0.000	6.288
0.3	50.700	26.625	15.720	3.900	0.000	0.033	1.478	0.000	0.000	6.013
0.25	49.600	26.675	13.920	4.467	0.000	0.033	1.600	0.000	0.000	5.875
0.2	51.300	27.475	14.680	4.333	0.100	0.000	1.533	0.000	0.000	6.338
0.15	49.600	26.925	14.340	3.700	0.000	0.000	1.478	0.000	0.000	6.275
0.1	49.400	26.100	14.720	4.433	0.000	0.000	1.756	0.000	0.000	6.250

Figure 5. *R*-matrix that summarizes the sensitivity of various conventional RBNs in a dynamism-connectivity space to different perturbations.

However, the question is where are the system regimes located? It is fundamental for us to know where and when the system is poised for resilience. To determine the separation of regimes, we applied two methods that are known for this purpose, namely, state space trajectories and

sensitivity to initial conditions. To implement these two approaches, we used the RBNLab software [83].

Figure 6a shows the matrix of trajectories through space of RBNs with different dynamism-connectivity values. Each cell in the matrix represents the state transitions of network nodes, as shown in Figure 6b, with oscillating (enclosed in red rectangle) and stable states. Oscillation indicates change in system behavior with the stable condition yet to be reached. A column in the cell represents the states of the network nodes at time *t*. Initial states are at the left and time (until 60 steps) flows to the right. Some nodes exhibit oscillations that quickly died out after a few steps, at times after only a single step, e.g., in Figure 6c, and were immediately followed by stable states. While other nodes continued to oscillate longer before reaching a stable state, others never reached stability, e.g., in Figure 6d, even as we continued the simulation for 4,500 time steps.

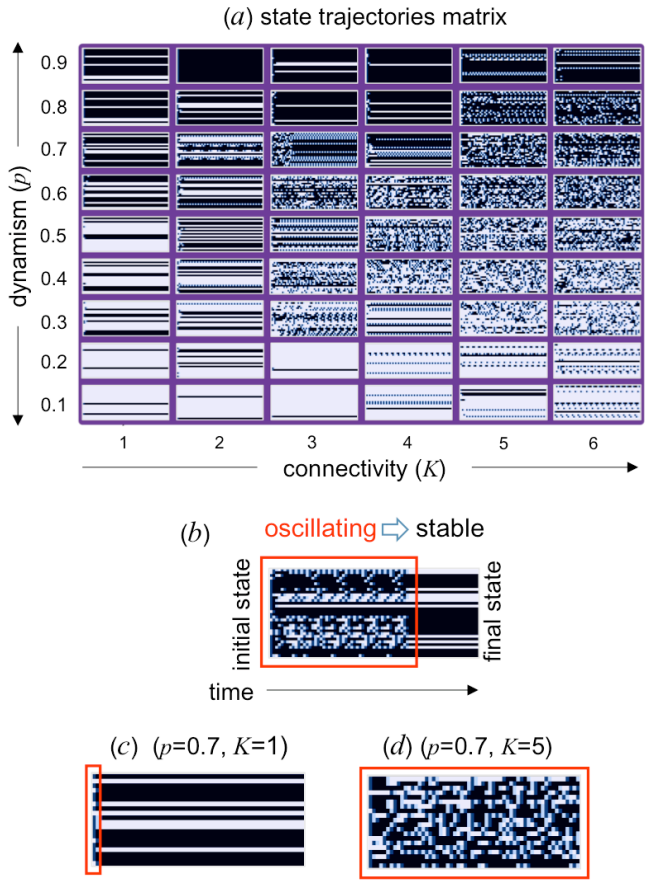


Figure 6. Trajectories of RBNs through the dynamism-connectivity space. Black and white colors indicate active and inactive states, respectively, while light blue colors indicate changing states.

It can be observed from Figure 6a that networks can become overly stable at lower *K* values and *p* close to 0 or 1. These sparsely connected networks had very short state cycles and the system froze up (stable to 0 or 1) very quickly. It can be said that they are rigid and uninteresting [21]. At *K*=2, however, we can see that not all nodes were frozen, unlike those at *K*=1. At (*p*=0.7, *K*=2), for example, the network nodes continued to fluctuate between 0 and 1 and never reached a stable state.

To further define this separation between regimes, we applied sensitivity to initial conditions as a measure of chaos [60][10] – if we change the initial point even by a little bit, the network ends up on a different path. We followed the approach of Gershenson [60] to measure this condition. Using again the RBNLab, we created an initial state I_1, and flipped one node (changed the bit value) to have another state I_2. We ran each initial state in the network for 4,500 time steps to obtain the final states F_1 and F_2, respectively. We then computed separately the normalized Hamming distance of the initial states, as in (1), and the final states to obtain parameter λ, as in (2):

$$H(I_1, I_2) = \frac{1}{N}\sum_{j=1}^{N}\left|i_{1j} - i_{2j}\right| \quad (1)$$

$$\lambda = H(F_1, F_2) - H(I_1, I_2) \quad (2)$$

While a negative λ means that both initial states moved to the same attractor, which is indicative of a stable or ordered state, a positive λ, on the other hand, indicates that the dynamics of similar initial states diverge, which is common to chaotic regimes.

Figure 7 shows the different λ values we obtained for the dynamism-connectivity matrix, and we can observe where order (in blue) and chaos (in red) are. We can also observe from the table of average λ per *K* where the critical regime is – the positive average λ started at *K*=2 (in purple), where there is a balanced mix of order and chaos.

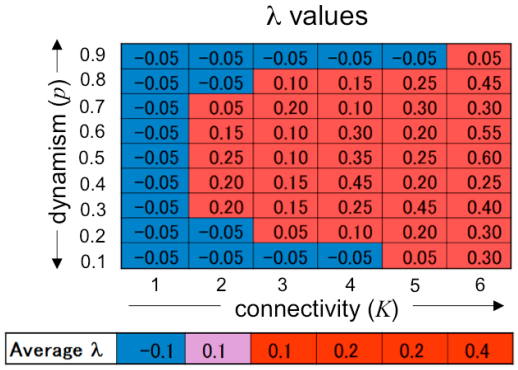


Figure 7. Map of the different regimes based on the sensitivity of conventional RBNs to initial conditions.

After applying the above two methods, we observed that it is at *K*=2 that the networks began to show signs of criticality. In other words, the critical regime lies mostly, although not entirely, at *K*=2. This gave us a hint as to where the regimes could be in our base case *R*-matrix in Figure 5.

We need to emphasize that the fundamental difference of the method by which we derived the *R*-matrix in Figure 5, as compared to the previous two, is that each *R*-value is a

synthesis of 1,000 network perturbations, which statistically tells more than the cells in the previous two matrices (in Figures 6a and 7) that were derived using at most only two network variations and a single-bit perturbation. This implies that the separation between regimes in the *R*-matrix in Figure 5 may be more pronounced. Looking at the *R*-matrix once again, we can observe how the range of *R*-values differed significantly per column, i.e., [48.0, 53.8], [24.4, 34.1], [12.3, 17.2], and [0.0, 6.3] for *K* equals to 1, 2, 3, and >3, respectively. Also, when we computed the average *R* per *K*, as shown in Figure 8, we can see how the average *R* significantly deteriorated by almost half per increase in connectivity starting with *K*=2 until *K*=6, and then stayed low until *K*=10. It is at *K*=1 that the RBNs were most robust. The RBNs losing robustness at *K*=2 may be indicative of critical slowing down or diminishing returns, and the system may therefore be tipped more easily into an alternative state, i.e., from order to chaos, which therefore reflects criticality. With these analyses, we hypothesize that the separation of regimes in our base case matrix is the one shown in Figure 9. We can therefore observe from the *R*-matrix the regimes that are present in our meta-theory.

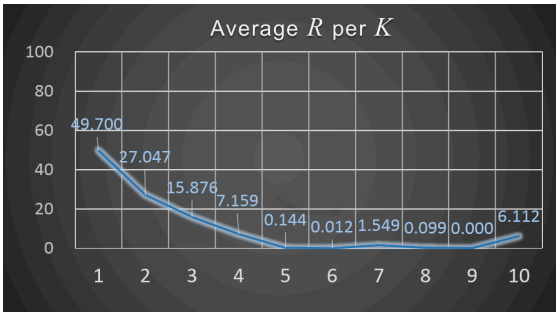


Figure 8. Each value corresponds to the averaged *R*-values across all *p*-values per *K*.

With the base case, we were able to empirically identify the initial range of values that would separate the regimes. After performing and analyzing all our simulations, we further observed that the range of values for each regime could be refined as follows – order: [43,100] (in blue), critical: [22, 43) (in purple), and chaos: [0, 22) (in red).

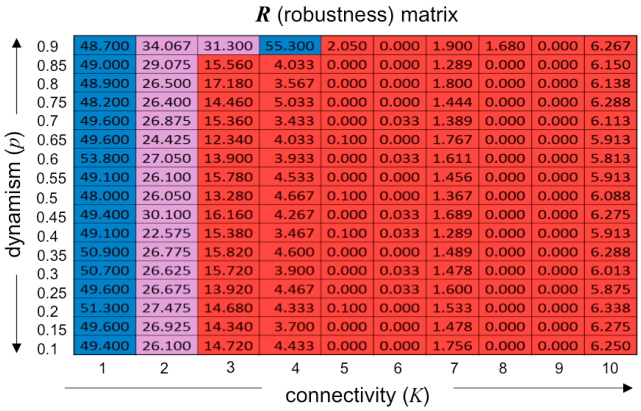


Figure 9. Map of the different regimes based on the sensitivity of conventional RBNs to perturbations. The colors of the cells correspond to that of our meta-theory: blue is order, purple is critical, and red is chaos.

C. *Tracking System Regimes and Transitions*

We now discuss the results of the different simulation models we ran while we varied the network and perturbation configurations. We begin with the one in Figure 10. Each rectangle in the 3×5 *topology-linkage* space is a *R*-matrix with *p*-*K* dimensions. For example, $R_{2,3}$ matrix corresponds to the robustness matrix of the RBNs with homogeneous topology and lattice linkage of size 2.5. The $R_{1,1}$ matrix is the same *R*-matrix in Figure 9.

We can see from the different *R*-matrices the interesting properties that emerged. We can observe the critical regime broadening to *K*=3 (in $R_{1,2}$, $R_{2,2}$, $R_{2,3}$, $R_{2,4}$, and $R_{2,5}$) or re-occurring at *K*>2 (in $R_{1,3}$ and $R_{1,5}$) between chaotic regimes in the fixed and homogeneous RBNs with wider lattice. These extensions and re-occurrences of the critical regime mean alternative opportunities for the system to take advantage of the benefits of the critical regime and the balance of stability and chaos [64]. The wider lattice led to more interdependencies among nearest neighbors, which formed small world networks that brought about such behaviors of the critical regime. This is consistent with the findings of Lizier et al. [69] that a small world topology leads to critical regime dynamics.

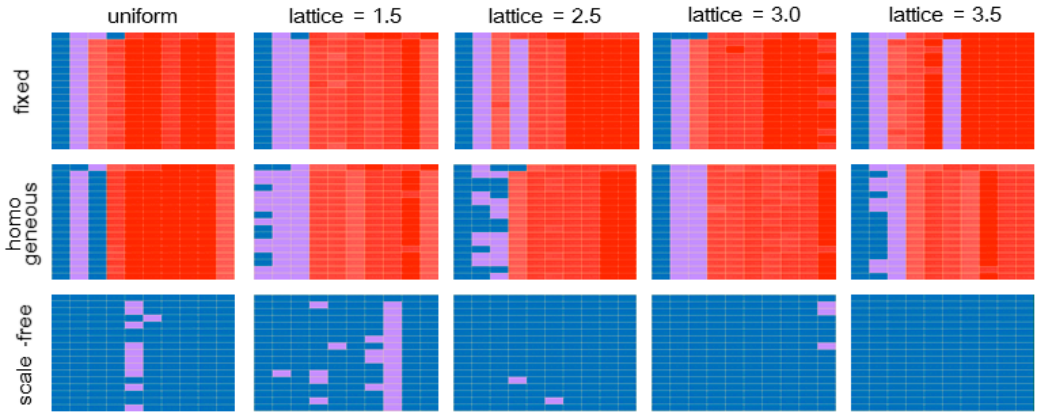
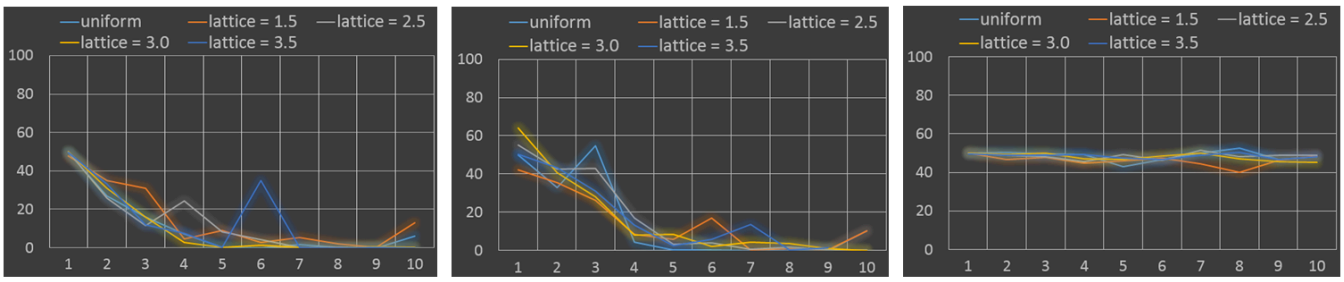


Figure 10. Map of the different regimes based on the sensitivity of RBNs to perturbations when dynamism, connectivity, topology, and linkage were varied



(*a*) fixed (*b*) homogeneous (*c*) scale-free

Figure 11. Mean robustness behaviors of the different RBNs in Figure 10

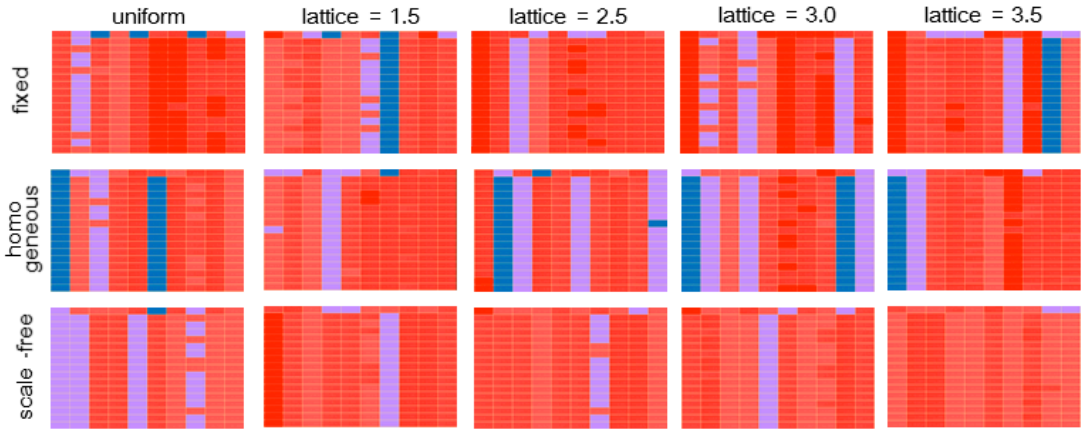


Figure 12. Map of the different regimes based on the sensitivity of RBNs to greater perturbations

Furthermore, the ordered regime expands with homogeneous RBNs. Since the number of input nodes is drawn independently at random, there is more variation in the way components influence each other. This also means that with less tighter connections among components (i.e., as the couplings in the network are loosened), the system becomes less vulnerable to perturbations. $R_{2,1}$, for example, shows how the system could transform to the next ordered state from a critical phase instead of deteriorating to a chaotic regime. With the scale-free topology, however, we can see highly robust RBNs. Since few nodes have more connections, and most nodes have few connections, changes can propagate through the RBN only in a constrained fashion.

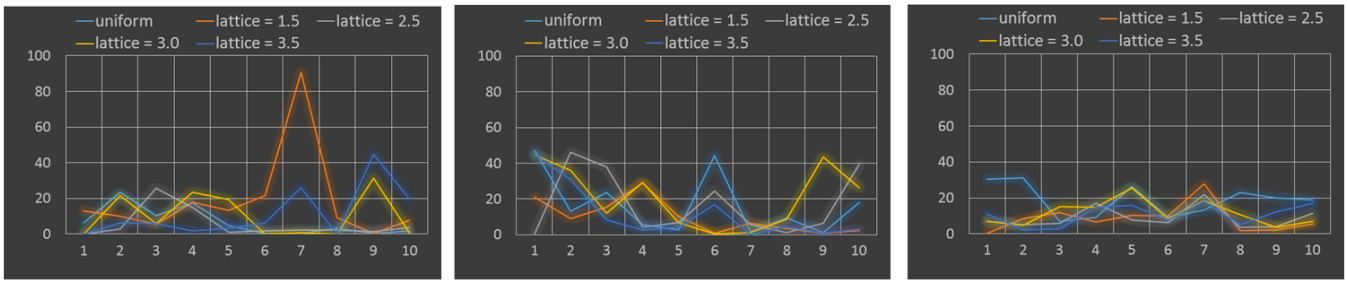
Figure 11 shows the mean (μ) R-values (the colored lines indicate the linkage type), with each value computed as:

$$\mu R_i = \frac{\sum_{p=0.1}^{p=0.9} R_{topology,linkage}[p, K_i]}{p \text{ steps}} \quad (3)$$

We can see the different μR-values continuously decreasing towards zero for the fixed and homogeneous topology. We interpret this as critical slowing down or diminishing returns that began at K=2 before transitioning to the chaotic regime. The μR-values for the scale-free RBNs, however, remained satisfactory throughout. Hence, a complex adaptive system may demonstrate [self-imposed] resilience by *broadening* (*extending*) *the critical regime*, *making the critical regime reoccur*, or *transforming to a scale-free topology*.

Lastly, by applying again the method of Müssel et al. [68], we tested next the sensitivity of the RBNs to greater perturbations. To simulate greater perturbation impacts, for each network transition, the transition function of one of the components is randomly selected, and then five bits of that function is flipped. Figure 12 shows the results we obtained. The first interesting phenomenon is the multiple occurrences of the ordered (in $R_{2,1}$ and $R_{2,4}$) and critical regimes (e.g., in $R_{1,4}$, $R_{2,3}$, $R_{2,4}$, $R_{3,1}$, etc.), even after the chaotic regimes, which are all indicative of resilience. The second is that we can obviously see how the behavior of the scale-free RBNs changed drastically, i.e., we could not find any ordered regime and their μR-values, as shown in Figure 13, dropped significantly. This is consistent with the findings of Barabási and Bonabeau [70] that scale-free networks are very robust against random failures but vulnerable to elaborate attacks. In our case, five flipping bits in every transition of the network was too much perturbation for the scale-free RBN.

This does not mean, however, that the resilience of the scale-free network is entirely lost. When we varied the parameter γ of the Zeta distribution from γ=2.5 to other values, another interesting phenomenon emerged, as shown in Figure 14 – we see more expansions and reoccurrences of the critical regime given other γ values. This means that *varying the scale-free network configuration* is another alternative to *prolong or increase the number of critical regime occurrences*, which is indicative of resilience.



(*a*) fixed (*b*) homogeneous (*c*) scale-free

Figure 13. Mean robustness behaviors of the different RBNs in Figure 12.

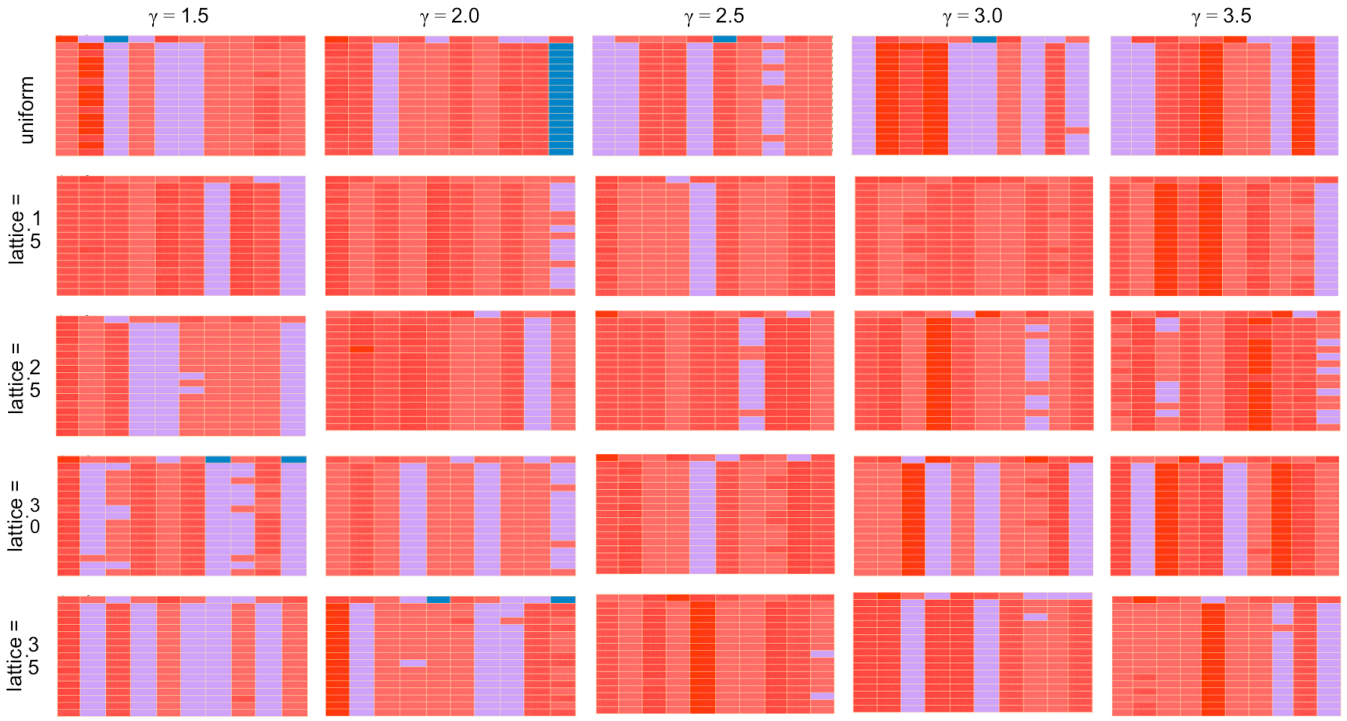


Figure 14. We simulated what will happen with changing γ values. The tables show that with other γ comes more expansions of the critical regime.

D. *Machine-Intelligent Modeling*

It is clear from our simulations that the combinations of the parameter values can characterize system states and regimes. The question now is how to infer these parameter relations as rules of contextual interaction behaviors that can define the complex system's adaptive and transformative walks and therefore define its resilience. Our solution is to use machine learning (ML) to discover the hidden relations.

The ML algorithm should infer a model that is predictive – given the states of the system and the perturbation, which regime in the space of possible regimes is the system in? We illustrate this viability of the predictive model in Figure 15. More importantly, the predictive model should help steer the system to a desirable regime – from the current states of the system and the perturbation, wherein the regime may be undesirable, which system parameters can or should be modified to achieve a desirable regime? *This capacity to modify the system parameters and predict the resulting regime can make the system resilient*.

We represent together the endogenous parameters of the system and the impact of the exogenous perturbation in a feature vector, which is a tuple of attribute values, i.e., <*connectivity*, *dynamism*, *topology*, *linkage*, *lattice*, *gamma*, *perturbation*>, where the possible values are as follows:

- connectivity = [1..10]
- dynamism = (0.10, 0.15, 0.20, ..., 0.80, 0.85, 0.90)
- topology = (fixed, homogenous, scale-free)
- linkage = (uniform, lattice)
- lattice = (1.5, 2.5, 3.0, 3.5)
- gamma = (1.5, 2.0, 2.5, 3.0, 3.5)
- perturbation = (minor, major)

We labeled each feature vector with the corresponding *R*-value that is indicative of the system regime.

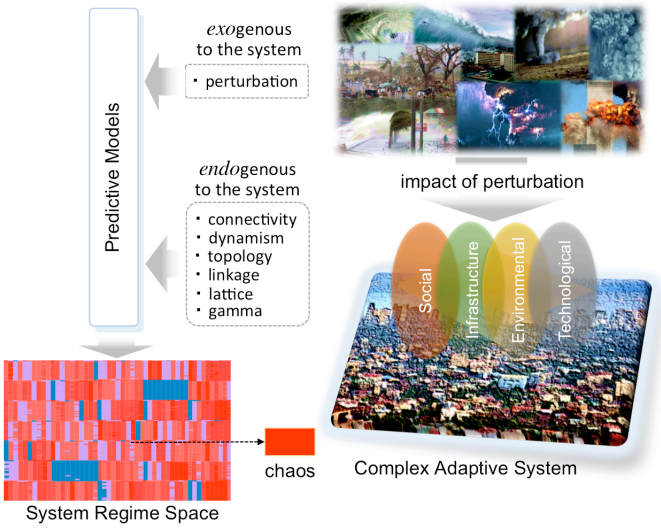


Figure 15. Viability of the predictive model

Our dataset consisted of 7,120 feature vectors, which corresponds to the various simulation scenarios we ran using our different RBN models. It is important to note that even though our data can still be considered minimal (considering for example that we only used one value for N, limited value ranges for the parameters, and only synchronous updates), the advantage of using a data-centric approach is that as the volume and dimensions of the data further increases, ML can be used to automatically handle the growing intricacies and complexities, as well as automatically infer the new relations emerging in the data.

To obtain the model with the best predictive capacity, we ran several well-known ML algorithms using the WEKA open-source software. Due to space constraints, it is best that we refer the reader to the documentation [71] of these algorithms. The ML algorithms are (*a*) function-based: linear regression models (LRM), multi-layer perceptrons (MLP), radial basis function networks (RBFN), and support vector machines for regression (SMOR), (*b*) instance-based or lazy: *K** and *k*-nearest neighbor (Ibk), and (*c*) tree-based: fast decision tree (REPTree) and MP5 model tree (M5PTree). We used *%-split* validation where *x*% of the data was used for training and the rest for testing the accuracy of the model. We measured the performance of the regression analysis in terms of correlation coefficient and root mean squared error to show the strength of prediction or forecast of future outcomes through a model or an estimator on the basis of observed related information. The correlation coefficient is also indicative of how good the approximation function might be constructed from the target model. We constructed several models by increasing the size of the training set from 10% to 90% of the total data, with increments of 10% (horizontal axis of the graphs in Figure 16), which allowed us to see the performance of the inferred models with few or even large amount of data, and also gave us the feel of an incremental learning capacity.

Figure 16 shows the accuracy of the predictive models. We can see that the models inferred by the decision tree-based (REPTree and MP5Tree) and instance-based *k*-nearest neighbor (Ibk) algorithms outperformed the others. These models can accurately predict in more than satisfactory levels the contextual interaction behaviors of the system even with only 10% of the data. We note that our goal at this time is not to improve the algorithms or discover a new one, but to prove the viability of our framework. We anticipate, however, that as the complexity of the system and the data grows, our algorithms would need to significantly improve.

The other advantage of the tree-based models is that the relation rules can be explicitly observed from the tree. Model trees are structured trees that depict graphical if-then-else rules of the hidden or implicit knowledge inferred from the dataset [72][73]. Model trees used for numeric prediction are similar to the conventional decision trees except that at the leaf is a linear regression model that predicts the numeric class value of the instances reaching it [73]. Figure 17 shows the upper portion (we could not show the entire tree of size 807 due to space constraints) of the REPTree we obtained using 10%-split validation with the elliptical nodes representing the features (colored so as to distinguish each feature), the edges specifying the path of the if-then-else rules, and the square leaf nodes specifying the corresponding *R*-values depending on which paths along the tree were selected. We can see how the rules delineated in a fine-grained manner the attribute values that eventually led to satisfactory predictions. We can also see how certain features are more significant to the classification task even early in the tree. The connectivity feature, for example, is prominent in both sides of the tree, and that the dynamism feature is not as significant in the upper levels compared to the lattice.

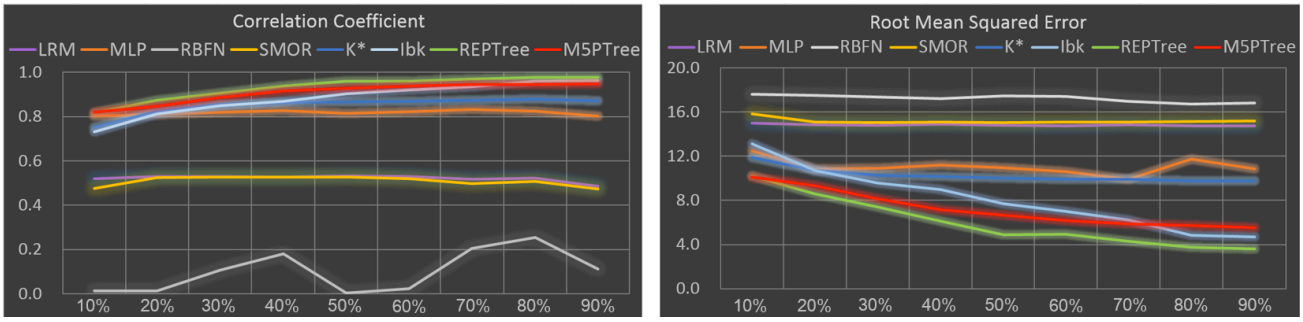


Figure 16. Prediction accuracy of the various models using %-split validation with increasing *x*% values

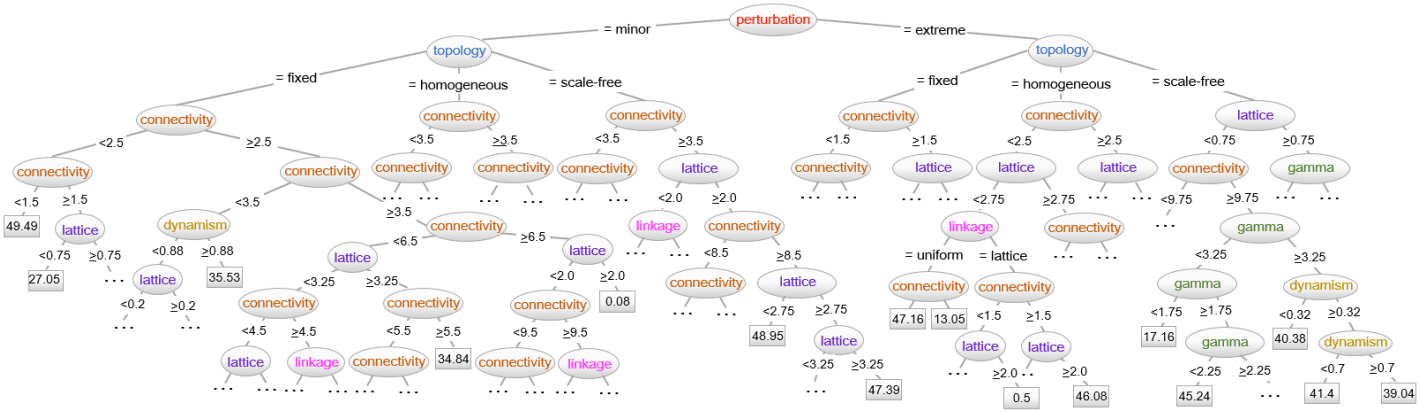


Figure 17. REPTree generated using Weka with a 10%-split validation. The size of the tree is 807, but only parts of it can be shown here due to space constraints. The nodes specify the features (colored so as to distinguish each) with the edges as attribute values, and the leaf nodes as *R*-values.

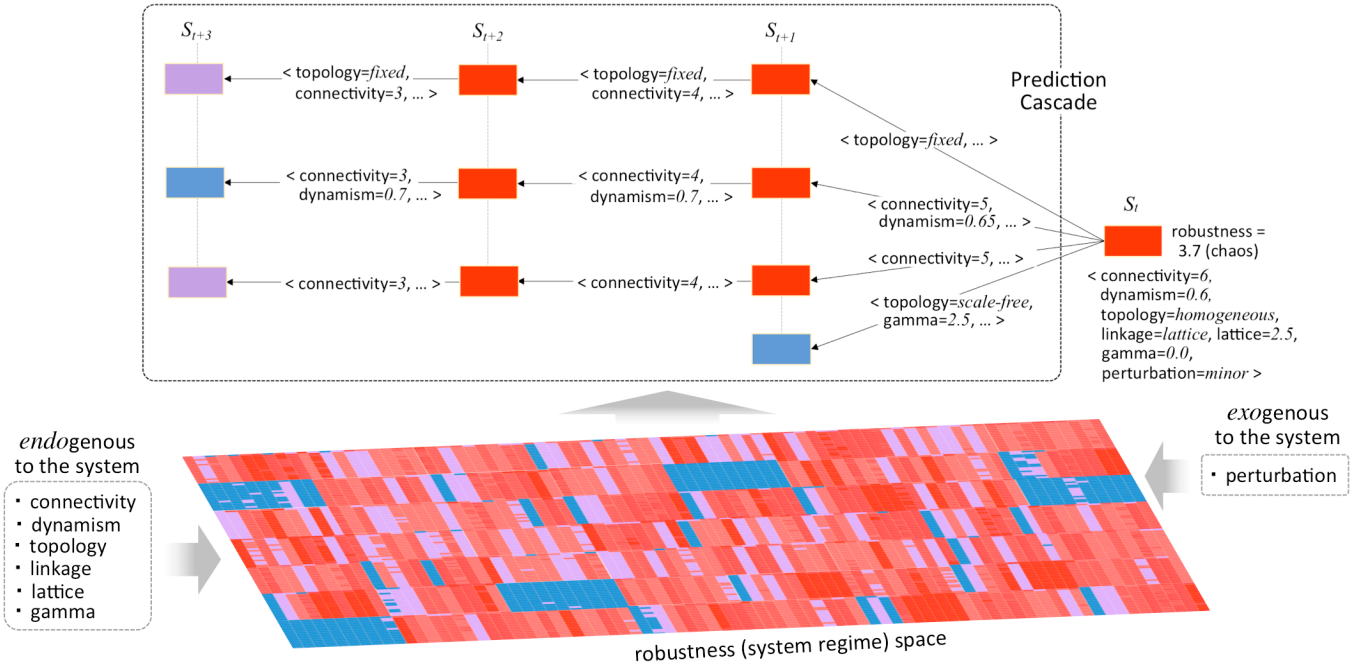


Figure 18. Illustration of how the strength of the predicitve models can be used to find the desirable regime states. For the top illustration, the regime states (colored blocks) and their contextual features (in angle brackets) were taken from the robustness maps, i.e., $R_{2,3}$, $R_{1,3}$, and $R_{3,3}$, in Figure 10.

All these mean that by observing the tree, we can determine which features are significant not only to the classification task, but more importantly to a more relevant sense, which features are actually influential to the resilient (as well as vulnerable) walks of the system.

Lastly, we illustrate in Figure 18 how our predictive model can be used to help steer the system to a desirable regime. The regime states shown in the figure, which were taken from the regime maps in Figure 10 (specifically, from $R_{2,3}$, $R_{1,3}$, and $R_{3,3}$), are obviously only a tiny portion of the possible entire regime space since each cell in every *R*-matrix in Figures 10, 12 and 14 is a regime state. Let us say that the system landed in the chaotic regime S_t, hence undesirable, as a result of the situational context (indicated by the feature vector shown below) it found itself into. The predictive model can be used to predict the resulting regime when one or more of the S_t contextual features are changed. Hence, from the current regime S_t, depending on which features the system change, the system may enter in one of the many possible S_{t+1} regime states. Although it seems elementary for the system to follow the prediction that suggests changing to scale-free topology with γ=2.5 in order to immediately reach a new ordered state, what should be considered is the high cost of changing to a topology that will necessitate breaking many of the current ties (e.g., geophysical, relational, monetary, etc.). Hence, it may be more advantageous for the system for the long haul to seek alternative paths with longer chaos, but less painful and costly. Again, this capacity to modify contextual features and predict the resulting regime demonstrates systems resilience.

E. Challenges Ahead for the Machine-Intelligent Modeling

We touched briefly in [1] the huge challenges we may need to address in our future work for a truly strong machine-intelligent predictive modeling capacity. We see the need to further elaborate here our points.

1) Finding the Optimal Path to the Desirable Regime

One formidable challenge in determining the optimal path to the desirable regime is the possibly huge number of potential paths, each with its own set of multiple candidate divergence. This is depicted in Figure 19, which is only a small portion of what could possibly be a huge set of system trajectories. Without special algorithms to find the correct paths efficiently, the required computing resources might be prohibitive. Equally challenging is the notion that the shortest path is not necessarily the optimal one. A myopic behavior by the system may find the immediate next step as optimal only to realize that the few poor or sub-optimal steps forward can eventually lead to better long-term outcomes. This also begs the question of how we can make our system's foresight to be as far reaching as possible.

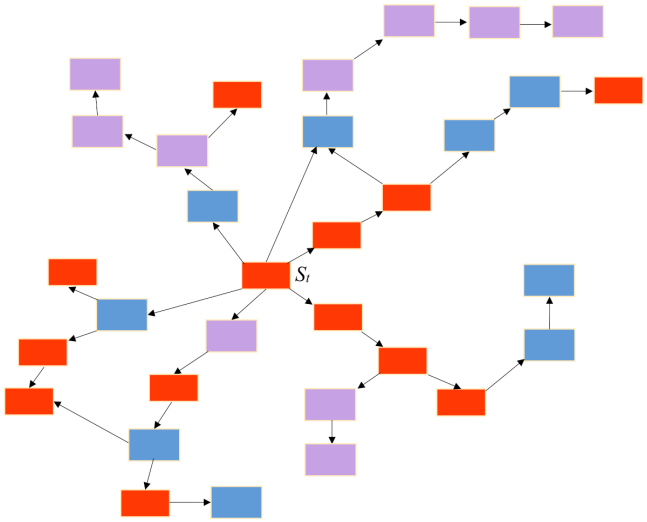


Figure 19. Depiction of possible trajectories of the model's prediction

2) Cost of Being Resilient

What is significantly missing in our modeling is the cost associated to every adaptation and transformation. Although we can account for the actual cost accurately only in retrospect, the challenge is for us to find the function that can meaningfully approximate the cost of system adaptation and transformation. Again is the notion that the shortest path is not necessarily the optimal one. The longer path may in fact possess the more bearable cost compared to that of an immediate, but extreme and radical, change. A similar but real-world insight was drawn after Katrina and Sandy that was shared by Goodman [74], which is "looking not at current losses and rebuilding what was destroyed, but rather at the costs – over time… in the long aftermath of the event." She further stated that it is "looking at any current destruction less as loss but rather as opportunity to create something completely different, perhaps elsewhere, with more wisdom, foresight and technological know-how."

3) Unknown-unknowns in Complexity

As pointed out in the report of the International Risk Governance Council, it is not always that we have knowledge of the multiple plausible alternate futures of our system's behavior [54]. Indeed, the nature of our systems is complex – nonlinear, spanning multiple simultaneous temporal and spatial scales, and with large interrelations and interdependencies among parts. Their evolving nature can affect physical, ecological, economic, and social dimensions simultaneously [28]. Our models can continue to exhibit incomplete and segregated knowledge for several reasons.

First, our predictions will be inaccurate or uncertain since our statistical extrapolations are based on a handful of analogous past experiences or mechanistic models that mislead to dire situations [28]. What we may have is dearth of historical data for predictive analysis [75]. We are therefore made to erroneously believe that certain situations are outside our expected possibilities and will never happen.

Second, our models may not demonstrate the critical links and interdependencies that mesh our systems into a cohesive and coherent whole. Our approaches are intimidated by the task of disentangling and elucidating a messy linked system-of-systems. This leads to a shallow and fragmented understanding of the evolving nature of our complex systems.

Lastly, even if perfect knowledge of costs and probabilities could be assigned to each and every alternative junction in the system phase trajectories, it is still highly possible that our calculations of the aggregate of all costs and probabilities over several junctions are inaccurate.

VI. Conclusion

With our world witnessing critical systemic changes [76], we are concerned with how our systems can be resilient, i.e., able to persist in, adapt to, or transform from dramatically changing circumstances. We believe that a deeper understanding of what fundamentally constitutes and leads to critical system changes sheds light to our understanding of the resilience of our systems. We discussed in length in this paper our contribution towards this understanding of resilience, which is a two-fold complex systems resilience framework that consists of a meta-theory that integrates long-standing theories on system-level changes and a machine-intelligent modeling task to infer from data the contextual behaviors of a resilient system.

Our framework of mutual reinforcing between theoretic and data-centric models allows for less perfect theory and inferred models to begin with, but with both components learning mutually and incrementally towards improved accuracy. Through our meta-theory we are able to have a strong basis of what will constitute our machine intelligent modeling. What the meta-theory can take from the inferred models, however, is to improve its knowledge by incorporating the fine-grained features, e.g., changing lattice and γ values, as well as the magnitude of the perturbations, which can have specific influences towards specific regimes. The knowledge exhibited by the meta-theory has to

incrementally improve based on what has been inferred by the intelligent modeling component. Our theoretic and data-centric models will surely need to co-evolve as we collect more data with increased range of network parameter values, other ways of introducing perturbations, using different transition schemes [60], and with agents having multiple states [67], among others. Furthermore, as nonlinear and unpredictable system intricacies become more detailed and pronounced, our machine-intelligent modeling should account for emerging algorithmic and data complexities.

Due to the absence of our intended real-world complex system data, we simulated the viability of our framework using random Boolean networks (RBNs). If RBNs were in fact sound models of complex systems, then our simulations would have sound basis – which is actually the case. RBNs are models of self-organization in which both structure and function emerge without explicit instructions [77]. Secondly, it is by the random nature of RBNs, albeit the transition functions are fixed, that systemic behaviors that emerge from known individual component behaviors cannot be determined a priori (e.g., exact number and characteristics of possible basins of attractions). All these and that a RBN's "infusion of historical happenstance is to simulate reality" [59, p.88] may attest to the fact that our meta-theory being demonstrated by RBNs is not at all forced. Our network-centric analyses show that the ability by which the system can vary, adjust or modify its controlling variables, specifically those that pertain to the connectivity, dynamism, topology, and sphere of influence of its components (all endogenous), and its capacity to withstand the disturbances (exogenous) that perturb it, will dictate the rules of its adaptation and transformation.

It would be a mistake, however, for us to conclude that since we find evidence of our meta-theory in RBNs, our meta-theory shall hold true for all kinds of complex adaptive systems. First, since we claim that our meta-theory should evolve together with the machine-intelligent modeling task to genuinely represent real phenomena that are endogenous and exogenous to the system means that our meta-theory (as well as our machine-intelligent modeling) is not one size fits all. However, as per the Campbellian realism, the meta-theory may be updated according to the specific contextual realities of the environment in which the system is embedded. Second, although it would also be inaccurate to say that "all" complex system realities can be approximated with RBNs, RBNs can indeed mimic certain complex system behaviors. However, by the fact that we intend to use real word data means that we believe that there are more realities to be discovered beyond what RBNs present. However, our conclusion is that the positive results we obtained with RBNs (which are sound models of complex systems) only demonstrate (proof of concept) the viability of our entire framework. If there is any added knowledge we may have derived regarding RBNs, this is only consequential to our primary objective of further elucidating the concept of complex systems resilience through our framework.

The major addition of this paper to our earlier work [1], which one would be remiss to overlook, is that we expanded our notion of systemic changes to what can actually push the system positively and be poised for resilience. The terms critical and chaos normally denote negative outcomes or impending perils. However, in light of resilient systems, such regimes may even be leveraged by the system to promote novel adaptations that can lead to desired sustainability. We also emphasized in this paper how architectural and empirical indicators of systemic changes, in combination, can help steer the system to desirable regimes. We have expanded our experiment results and analyses to further demonstrate this.

We believe that we have barely scratched the surface of our research problem. Our immediate next concern is to find and collect real world data on hyper-connected composite systems in order for us to further ground our meta-theory and machine-intelligent modeling approaches.

REFERENCES

[1] R. Legaspi and H. Maruyama, "Meta-theory and machine-intelligent modeling of systemic changes for the resilience of a complex system," Proc. Tenth International Conference on Systems (ICONS 2015), IARIA, L. Koszalka and P. Lorenz, Eds. ThinkMind Digital Library, 2015, pp. 102-111.

[2] J. Rockström et al., "A safe operating space for humanity," Nature, vol. 461, Feature, September 2009.

[3] W.F. Ruddiman, "The Anthropocene," Annual Review of Earth and Planetary Sciences, vol. 41, pp. 45-68, May 2013.

[4] A. Zolli and A.M. Healy, Resilience: Why Things Bounce Back. New York, NY: Free Press, July 2012.

[5] P. Martin-Breen and J.M. Anderies, "Resilience: A literature review," The Rockefeller Foundation, September 18, 2011.

[6] S.A. Kauffman, "Antichaos and adaptation," Scientific American, vol. 265, no. 2, pp. 78-84, August 1991.

[7] S.A. Kauffman, The Origins of Order: Self-Organization and Selection in Evolution. New York, NY: Oxford University Press, 1993.

[8] P. Bak, How Nature Works: The Science of Self-Organised Criticality. New York, NY: Copernicus Press, 1996.

[9] C.S. Holling, "Understanding the complexity of economic, ecological, and social systems," Ecosystems, vol. 4, no. 5, pp. 390-405, 2001.

[10] S.E. Page, Diversity and Complexity. Princeton, NJ: Princeton University Press, 2011.

[11] G. Paperin, D.G. Green, and S. Sadedin, "Dual-phase evolution in complex adaptive systems," Journal of The Royal Society Interface, vol. 8, no. 58, pp. 609-629, 2011.

[12] N.H. Packard, "Adaptation toward the edge of chaos," in Dynamic Patterns in Complex Systems, J.A.S. Kelso, A.J. Mandell, and M.F. Shlesinger, Eds. Singapore: World Scientific, pp. 293-301, 1988.

[13] C.G. Langton, "Computation at the edge of chaos: Phase transitions and emergent computation," Physica D, vol. 42, pp. 12-37, 1990.

[14] M. Mitchell, P.T. Hraber, P.T., and J.P. Crutchfield, "Revisiting the edge of chaos: Evolving cellular automata to perform computations," Complex Systems, vol. 7, no. 2, pp. 89-130, 1993.

[15] J.A. Tainter, The Collapse of Complex Societies. Cambridge, UK: Cambridge University Press, 1988.

[16] D. Meadows, J. Randers, and D. Meadows, Limits to Growth: The 30-Year Update. White River Junction, VT: Chelsea Green Publishing Company, 2004.

[17] J. Diamond, Collapse: How Societies Choose to Fail or Survive. England: Penguin Publishing Group, 2011.

[18] J. Casti, X-Events: The Collapse of Everything. New York, NY: HarperCollins Publishers, 2012.

[19] B. McKelvey, "Self-Organization, complexity catastrophe, and microstate models at the edge of chaos," in Variations in Organization Science: In Honor of Donald T. Campbell, J.A.C. Baum and B. McKelvey, Eds. Thousand Oaks, Calif: SAGE Publications, pp. 279-307, 1999.

[20] B. McKelvey, "Towards a Campbellian realist organization science," in Variations in Organization Science: In Honor of Donald T. Campbell, J.A.C. Baum and B. McKelvey, Eds. Thousand Oaks, Calif: SAGE Publications, pp. 279-307, 1999.

[21] D.L. Levy, "Applications and limitations of complexity theory in organizational theory and strategy," in Handbook of Strategic Management, J. Rabin, G.J. Miller, and W.B. Hildreth, Eds. New York: Marcel Dekker, pp. 67-87, 2000.

[22] J.M. Anderies, B.H. Walker, and A.P. Kinzig, "Fifteen weddings and a funeral: Case studies and resilience-based management," Ecology and Society, vol. 11, no. 1, article 21, 2006.

[23] J.C. Rocha, "The domino effect: A network analysis of regime shifts drivers and causal pathways," Master's Thesis, Ecosystems, Governance and Globalization Program, Stockholm University, May 2010. Available online: http://www.diva-portal.org/smash/get/diva2:369772/FULLTEXT02. Accessed on 2015.11.24.

[24] Ö. Bodin and M. Tengö, "Disentangling intangible social-ecological systems," Global Environmental Change, vol. 22, no. 2, pp. 430-439, May 2012.

[25] F. Morrison, The Art of Modeling Dynamic Systems: Forecasting for Chaos, Randomness and Determinism. Mineola, NY: Dover Publications, 2008.

[26] J. Bohn, V. Coroamă, M. Langheinrich, F. Mattern, and M. Rohs, "Social, economic, and ethical implications of ambient intelligence and ubiquitous computing," in Ambient Intelligence, W. Weber, J.M. Rabaey, and E. Aarts, Eds. Springer Berlin Heidelberg, pp. 5-29, 2005.

[27] R. Valerdi et al., "A research agenda for systems of systems architecting," International Journal of System of Systems Engineering, vol. 1, no. 1/2, pp. 171-188, 2008.

[28] S.R. Carpenter, C. Folke, M. Scheffer, and F. Westley, "Resilience: Accounting for the noncomputable," Ecology and Society, vol. 14, no. 1, article 13, 2009.

[29] S. Poslad, Ubiquitous Computing: Smart Devices, Environments and Interactions. Wiley, 2009.

[30] K.J. Shim, N. Pathak, M.A. Ahmad, C. DeLong, Z. Borbora, A. Mahapatra, and J. Srivastava, "Analyzing human behaviour from multiplayer online game logs: A knowledge discovery approach," Trends and Discoveries, vol. 26, no. 1, pp. 85-89, 2011.

[31] R.D. Stacey, "The science of complexity: An alternative perspective for strategic change processes," Strategic Management Journal, vol. 16, no. 6, pp. 477-495, September 1995.

[32] E. Thelen, "Dynamic systems theory and the complexity of change," Psychoanalytic Dialogues vol. 15, no. 2, pp. 225-283, 2005.

[33] C. Folke et al., "Resilience thinking: Integrating resilience, adaptability, and transformability," Ecology and Society, vol. 15, no. 4, article 20, 2010.

[34] S.R. Carpenter et al., "General resilience to cope with extreme events," Sustainability, vol. 4, pp. 3248-3259, 2012.

[35] L. Chelleri, J.J. Waters, M. Olazabal, and G. Minucci, "Resilience trade-offs: addressing multiple scales and temporal aspects of urban resilience," Environment & Urbanization, pp. 1-18, 2015.

[36] M. Scheffer et al., "Anticipating critical transitions," Science, vol. 338, no. 6105, pp. 334-348, 19 October 2012.

[37] D. Rickles, P. Hawe, and A. Shiell, "A simple guide to chaos and complexity," Journal of Epidemiology and Community Health, vol. 61, no. 11, pp. 933-937, November 2007.

[38] T. Taylor, "Exploring the concept of open-ended evolution," Proc. Thirteenth International Conference on the Simulation and Synthesis of Living Systems (Artificial Life 13), C. Adami, D.M. Bryson, C. Ofria, and R.T. Pennock, Eds. MIT Press, 2012, pp. 540-541.

[39] D.R. Gilpin and P.J. Murphy, Crisis Management in a Complex World. Oxford, NY: Oxford University Press, 2008.

[40] R.T. Pascale, "Surfing the edge of chaos," MIT Sloan Management Review, Spring 1999. Available online: http://sloanreview.mit.edu/article/surfing-the-edge-of-chaos/, accessed 2015.11.25.

[41] T.G. Lewis, Bak's Sand Pile: Strategies for a Catastrophic World. Williams, CA: Agile Press, 2011.

[42] D. Levinthal, "Adaptation on rugged landscapes," Management Science, vol. 43, no. 7, pp. 934-950, 1997.

[43] J.P. Crutchfield, "The hidden fragility of complex systems – Consequences of change, changing consequences," in Cultures of Change: Social Atoms and Electroniuc Lives, G. Ascione, C. Massip, J. Perello, Eds. Barcelona, Spain: ACTAR D Publishers, pp. 98-111, 2009.

[44] A. Farazmand, "Chaos and transformation theories: A theoretical analysis with implications for organization theory and public management," Public Organization Review: A Global Journal, vol. 3, Kluwer Academic Publishers, pp. 339-372, 2003.

[45] P.H. Longstaff, T.G. Koslowski, and W. Geoghegan, "Translating resilience: A framework to enhance communication and implementation," Proc. 5th International Symposium on Resilience Engineering, June 2013.

[46] J.H. Miller and S.E. Page, Complex Adaptive Systems. Princeton, NJ: Princeton University Press, 2007.

[47] D. Krotov, J.O. Dubuis, T. Gregor, and W. Bialek, "Morphogenesis at criticality," Proceedings of the National Academy of Sciences of the United States of America (PNAS), vol. 111, no. 10, pp. 3683-3688, March 2014.

[48] G.P. Wagner and L. Altenberg, "Perspective: Complex adaptations and evolution of evolvability," Evolution, vol. 50, no. 3, pp. 967-976, June 1996.

[49] B. Keim, "Biologists find new rules for life at the edge of chaos," WIRED, Science, May 2014. Available online: http://www.wired.com/2014/05/criticality-in-biology/, accessed 2015.11.25.

[50] C. Royle, "Dialectics, nature and the dialectics of nature," International Socialism, vol. 141, pp. 97-118, 2014.

[51] M. Fisher, "The Emperor's speech: 67 years ago, Hirohito transformed Japan forever," The Atlantic, Aug 15, 2012. Available online: http://www.theatlantic.com/international/archive/2012/08/the-emperors-speech-67-years-ago-hirohito-transformed-japan-forever/261166/, accessed 2015.11.30.

[52] C.S. Holling, "Resilience and stability of ecological systems," Annual Review of Ecology and Systematics, vol. 4, pp. 1-23, 1973.

[53] A.W. Covert III, R.E. Lenski, C.O. Wilke, and C. Ofria, "Experiments on the role of deleterious mutations as stepping stones in adaptive evolution," Proceedings of the National Academy of Sciences of the United States of America (PNAS), vol. 110, no. 34, August 20, 2013.

[54] IRGC, Guidelines for Emerging Risk Governance. Lusanne: International Risk Governance Council (IRGC), 2015. Available from: http://www.irgc.org. Available online: http://www.mercerfinancialservices.com/content/dam/mmc-

web/Global%20Risk%20Center/Files/IRGC-Emerging-Risk-WEB-31Mar.pdf, accessed 2015.11.30.

[55] J.A. Schumpeter, Capitalism, Socialism and Democracy. New York: Harper & Brothers Publisher, 1942.

[56] H. Maruyama, R. Legaspi, K. Minami, and Y. Yamagata, "General resilience: taxonomy and strategies," Proc. IEEE 2014 International Conference and Utility Exhibition on Green Energy for Sustainable Development (ICUE), IEEE Press, March 19-21, 2014, pp. 1-8.

[57] R. Albert and A.-L. Barabási, "Dynamics of complex systems: Scaling laws for the period of boolean networks," Physical Review Letters, vol. 84, no. 24, pp. 5660-5663, June 2000.

[58] K.A. Hawick, H.A. James, and C.J. Scogings, "Simulating large random Boolean networks," Research Letters in the Information and Mathematical Sciences, vol. 11, pp. 33-43, 2007.

[59] R.S. Cohen, "How useful is the complexity paradigm without quantifiable data? A test case: The patronage of 5th-6th century Buddhist caves in India," in Chaos and Society (Frontiers in Artificial Intelligence and Applications), A. Albert, Ed. Amsterdam, The Netherlands: IOS Press, pp. 83-99, 1995.

[60] C. Gershenson, "Updating schemes in random Boolean networks: Do they really matter?" Proc. Ninth International Conference on Simulation and Synthesis of Living Systems (Artificial Life IX). MIT Press, 2004, pp. 238-243.

[61] S.A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," Journal of Theoretical Biology, vol. 22, no. 3, pp. 437-467, March 1969.

[62] T. Zhou, B.-H. Wang, P.-L. Zhou, C.-X. Yang, and J. Liu, "Self-organized Boolean game on networks," Physical Review E, vol. 72, no. 046139, pp. 1–6, October 28, 2005.

[63] A.M. Machado and A.L.C. Bazzan, "Self-adaptation in a network of social drivers: Using random Boolean networks," Proc. Workshop on Organic Computing, held in conjunction with the 8th International Conference on Autonomic Computing (ICAC '11), 2011, pp. 33-40.

[64] C. Gershenson, "Guiding the self-organization of random Boolean networks," Theory in Biosciences, vol. 131, no. 3, pp. 181-191, November 30, 2011.

[65] B. Luque, F.J. Ballesteros, and M. Fernandez, "Variances as order parameter and complexity measure for random Boolean networks," Journal of Physics A: Mathematical and General, vol. 38, pp. 1031-1038, 2005.

[66] I. Harvey and T. Bossomaier, "Time out of joint: Attractors in asynchronous random Boolean networks," Proc. Fourth European Conference on Artificial Life, July 1997, pp. 67-75.

[67] R.V. Solé, B. Luque, and S. Kauffman, Phase Transitions in Random Networks with Multiple States," Santa Fe Institute Working Paper 2000-02-011, 2000. Available online: http://www.santafe.edu/media/workingpapers/00-02-011.pdf, accessed 2015.11.30.

[68] C. Müssel, M. Hopfensitz, and H.A. Kestler, BoolNet Package Vignette, February 25, 2015. Available online: http://cran.r-project.org/web/packages/BoolNet/vignettes/BoolNet_package_vignette.pdf, accessed 2015.11.30

[69] J.T. Lizier, S. Pritam, and M. Prokopenko, "Information dynamics in small-world Boolean networks," Artificial Life, vol. 17, no. 4, pp. 293-314, Fall 2011.

[70] A.-L. Barabási and E. Bonabeau, "Scale-free networks," Scientific American, vol. 288, no. 5, pp. 60-69, May 2003.

[71] Weka 3: Data Mining Software in Java. Available online: http://www.cs.waikato.ac.nz/ml/weka/index.html, accessed 2015.09.15.

[72] R.C. Barros, M.P. Basgalupp, D.D. Ruiz, A.C.P.L.F. de Carvalho, and A.A Freitas, "Evolutionary model tree induction," Proc. ACM Symposium on Applied Computing (SAC '10), 2010, pp. 1131-1137.

[73] M. Göndör and V.P. Bresfelean, "REPTree and M5P for measuring fiscal policy influences on the Romanian capital market during 2003-2010," International Journal of Mathematics and Computers in Simulation, vol. 6, no. 4, pp. 378-386, 2012.

[74] A. Goodman, "Calculating the cost of disaster vs. the price of resilience," GreenBiz, December 7, 2012. Available online: http://www.greenbiz.com/blog/2012/12/07/Calculating-cost-disaster, accessed 2015.12.01.

[75] T.B. Fowler and M.J. Fischer, Eds., Rare Events: Can We Model the Unforeseen? Sigma, vol. 10, no. 1. Noblis, September 2010. Available online: http://www.noblis.org/noblis-media/20f758e0-b3b9-4b76-8b81-4cde0d7341f9, accessed 2015.12.01.

[76] R. Legaspi, H. Maruyama, R. Nararatwong, and H. Okada, "Perception-based Resilience: Accounting for the impact of human perception on resilience thinking," Proc. 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, December 2014, pp. 547-554.

[77] H. Atlan, F. Fogelman-Soulie, J. Salomon, and G. Weisbuch, "Random Boolean Networks," Cybernetics and Systems: An International Journal, vol. 12, nos. 1-2, pp. 103-121, 1981.

[78] A. Dixit, "Comments on conference version of paper," in New Directions in Trade and Theory, A.V. Deardoff, J.A. Levinsohn, and R.M. Stern, Eds. University of Michigan Press, pp. 47-52, 1995.

[79] J.W. Dower, Embracing Defeat: Japan in the Wake of World War II. W.W. Norton & Co./ The New Press, 1999.

[80] C.M. Christensen, The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail. Harvard Business School Press, 1997.

[81] B. Derrida and Y. Pomeau, "Random networks of automata: A simple annealed approximation," Europhysics Letters, vol. 1, no. 2, pp. 45-49, 1986.

[82] C. Müssel, M. Hopfensitz, and H.A. Kestler, "BoolNet – An R package for generation, reconstruction and analysis of Boolean networks," Bioinformatics, vol. 26, no. 10, pp. 1378-1380, 2010. Package available online: http://cran.r-project.org/web/packages/BoolNet/index.html, accessed 2015.11.30.

[83] C. Gershenson, RBNLab, 2003. Online software: http://rbn.sourceforge.net.

Designing Data Processing Systems with NumEquaRes

Stepan Orlov and Nikolay Shabrov

Computer Technologies in Endineering dept.
St. Petersburg State Polytechnical University
St. Petersburg, Russia
Email: majorsteve@mail.ru, shabrov@rwwws.ru

***Abstract*—A new Web application for numerical simulations, NumEquaRes, is presented. Its design and architecture are motivated and discussed. Key features of NumEquaRes are the ability to describe data flows in simulations, ease of use, good data processing performance, and extensibility. Simulation building blocks and several examples of application are explained in detail. Technical challenges specific to Web applications for simulations, related to performance and security, are discussed. In conclusion, current results are summarized and future work is outlined.**

Keywords–Simulation; Web application; Ordinary differential equations.

I. INTRODUCTION

This work is an extended version of paper [1]. We present a new Web application, NumEquaRes [2] (the name means "Numerical Equation Research"). It is a general tool for numerical simulations available online. Currently, we are targeting small systems of ordinary differential equations (ODE) or finite difference equations arising in the education process, but that might change in the near future — see Section XI.

The reasons for developing yet another simulation software have emerged as follows. Students were given tasks to deduce the equations of motions of mechanical systems — for example, a disk rolling on the horizontal plane without slip [3], or a classical double pendulum [4], — and to try further investigating these equations. While in some cases such an investigation can more or less easily be done with MATLAB, SciLab, or other existing software, in other cases the situation is like there is no (freely available) software that would allow one to formulate the task for numerical investigation in a straightforward and natural way.

For example, the double pendulum system exhibits quasi-periodic or chaotic behavior [4], depending on the initial state. To determine which kind of motion corresponds to certain initial state, one needs the Poincaré map [5] — the intersection of phase trajectory with a hyperplane. Of course, there are ODE solvers in MATLAB that produce phase trajectories. We can obtain these trajectories as piecewise-linear functions and then compute intersections with the hyperplane. But what if we want 10^4–10^5 points in the Poincaré map? How many points do we need in the phase trajectory? Maybe 10^7 or more? Obviously, the simplest approach described above would be a waste of resources. A better approach would look at trajectory points one by one, test for intersections with hyperplane, and forget points that are no longer needed. But there is no straightforward way to have a simulation process like this in MATLAB.

Of course, there is software (even free software) that can compute Poincaré maps. For example, the XPP (X-Window PhasePlane) tool [6] can do that. But what we have learned from our examples is that we need certain set of features that we could not find in any existing software. These features are as follows:

- ability to explicitly specify how data flows in a simulation should be organized;
- reasonable computational performance;
- ease of use by everyone, at least for certain use cases;
- extensibility by everyone who needs a new feature.

The first of these features is very important, but it is missing in all existing tools we tried (see Section IX). It seems that developers of these tools and authors of this paper have different understanding of what a computer simulation can be. Common understanding is that the goal of any simulation is to reproduce the behavior of system being investigated. Therefore, numerical simulations most often perform time integration of equations given by a mathematical model of the system. In this paper, we give the term *simulation* a more general meaning: it is data processing. Given that meaning, we do not think the term is misused, because time integration of model equations often remains the central part of the entire process. Importantly, a researcher might need to organize the execution of that part differently, e.g., run initial value problem many times for different initial states or parameters, do intermediate processing on consecutive system states produced by time integrator, and so on.

Given the above general concept of numerical simulation, our goal is to provide a framework that supports the creation of data processing algorithms in a simple and straightforward manner, avoiding any coding except to specify model equations.

Next sections describe design decisions and technologies chosen for the NumEquaRes system (Section II); simulation specification (Section III) and workflow semantics (Section IV); software architecture overview (Section V); performance, extensibility, and ease of use (Section VI); simulation building blocks (SectionVII); examples of simulations (Section VIII); comparison with existing tools (Section IX); technical challenges conditioned by system design (Section X). Section XI summarizes current results and presents a roadmap for future work.

II. DESIGN DECISIONS AND CHOICE OF TECHNOLOGIES

Keeping in mind the primary goals formulated above, we started our work. Traditionally, simulation software have been

designed as desktop applications or high performance computing (HPC) applications with desktop front-ends. Nowadays, there are strong reasons to consider Web applications instead of desktop ones, because on the one hand, main limitations for doing so in the past are now vanishing, and, on the other hand, there are many well-known advantages of Web apps. For example, our "ease of use" goal benefits if we have a Web app, because this means "no need for user to install any additional software".

Thus, we have decided that our software has to be a Web application, available directly in user's Web browser.

Now, the "extensibility by everyone" goal means that our project must be free software, so the GNU Affero GPL v3 license has been chosen. That should enforce the usefulness of software for anyone who could potentially extend it.

The "Reasonable performance" goal has determined the choice of programming language for software core components. Our preliminary measurements have shown that for a typical simulation, native code compiled from C++ runs approx. 100 times faster than similar code in MATLAB, SciLab, or JavaScript (as of JavaScript, we tested QtScript from Qt4; with other implementations, results might be different). Therefore, we decided that the simulation core has to be written in C++. The core is a console application that runs on the server and interacts with the outer world through its command line parameters and standard input and output streams. It can also generate files (e.g., text or images).

JavaScript has been chosen as the language for simulation description and controlling the core application. However, this does not mean that any part of running simulation is executing JavaScript code.

The decision to use the Qt library has been made, because it provides a rich set of platform-independent abstractions for working with operating system resources, and also because it supports JavaScript (QtScript) out of the box.

Other parts of the applications are the Web server, the database engine, and components running on the client side. For the server, we preferred Node.js over other technologies because we believe its design is really suitable for Web applications — first of all, due to the asynchronous request processing. For example, it is easy to use HTML5 Server Sent Events [7] with Node.js, which is not the case with LAMP/WAMP [8].

The MongoDB database engine has been picked among others, because, on the one hand, its concept of storing JSON-like documents in collections is suitable for us, and, on the other hand, we do not really need SQL, and, finally, it is a popular choice for Node.js applications.

As of the client code running in the browser, the components used so far are jQuery and jQueryUI (which is no surprise), the d3 library [9] for interactive visualization of simulation schemes, the marked [10] and MathJax [11] libraries to format markdown pages with TEX formulas. In the future, we are planning to add 3D visualization using WebGL.

III. SIMULATION SPECIFICATION

The very primary requirement for NumEquaRes is to provide user with the ability to explicitly specify how data flows are organized in a simulation. This determines how simulations are described. This is done similarly to, e.g., the description of a scheme in the Visualization Toolkit (VTK) [12], employing the "pipes and filters" design pattern. The basic idea is that simulation is a *data processing system* defined by a scheme consisting of *boxes* (filters) with *input ports* and *output ports* that can be connected by *links* (pipes). Output ports may have many connections; input ports are allowed to have at most one connection. Simulation data travels from output ports to input ports along the links, and from input ports to output ports inside boxes. Inside each box, the data undergoes certain transformation determined by the box type.

Typically boxes have input and output ports, so they are *data transformers*. Boxes without input ports are *data sources*, and boxes without output ports are *data storage*.

Simulation data is considered to be a sequence of *frames*. Each frame can consist of a scalar real value or one-dimensional or multi-dimensional array of scalar real values. The list of sizes of that array in all its dimensions is called *frame format*. For example, format $\{1\}$ describes frames of scalar values, and format $\{500,400\}$ describes frames of two-dimensional arrays, each having size 500×400. The format of each port is assumed to be fixed during simulation. Figure 1 shows an example of sequences of data frames of different formats.

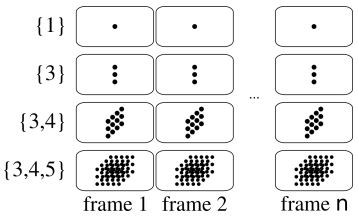


Figure 1. Examples of data frame sequences.

In addition, NumEquaRes supports *element labels* for scalar and one-dimensional data frames. The idea is to give a name to each element of a data frame. Due to labels, user can easily identify parameters specified for `Param` boxes (see Section VII-A), and have better understanding of the data. Notice that labels are not part of frame format, so format compatibility check does not rely on labels.

Links between box ports are logical data channels, they cannot modify data frames in any way. This means that data format has to be the same at ports connected by a link. Some ports define data format, while some do not; instead, such a port takes the format of the port connected to it by a link. Thus, data format *propagates along links* (together with element labels, if any). Furthermore, data format can also *propagate through boxes*. This allows to provide a quite flexible design to fit the demands of various simulations.

Figure 2 shows an example of connections between box ports. Each box has a type (e.g., **`Param`**, displayed in bold face) and a name (e.g., `odeParam`), and some input and/or output ports. The figure shows data flow direction along links with solid arrows, and frame format propagation direction with dashed arrows. For ports defining data format, dashed arrows start with a diamond. Notice, e.g., how the `odeInitState` box in this example knows that user should specify values $q, \dot{q}, t$ for pendulum initial state: `odeInitState` receives the format $\{3\}$ and element labels $q, \dot{q}, t$ from the `initState`

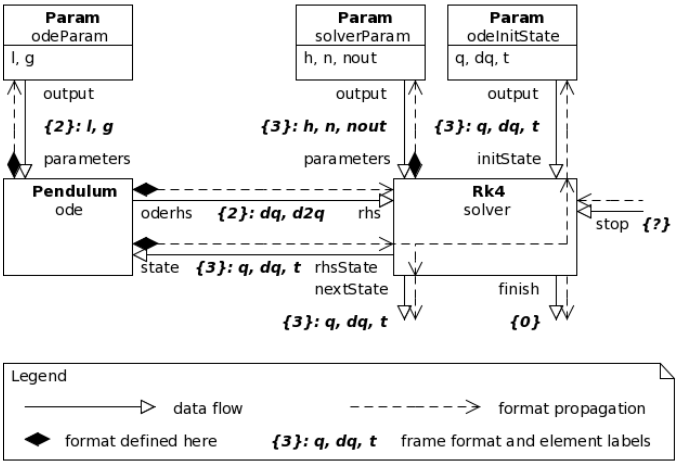


Figure 2. Boxes, ports, links, and frame format propagation.

port of box `solver`. The format of that port is induced by the format of port `rhsState` of the same box (due to format propagation through boxes of type `Rk4`, see Section VII-E4). The `rhsState` port of box `solver` receives the frame format and element labels from port `state` of box `ode`. The `ode` box is the origin of format and labels.

IV. Simulation workflow

This section explains how simulation runs, i.e., how the core application processes data frames generated by boxes.

Further, the main routine that controls the data processing is called *runner*.

A. Activation notifications

When a box generates a data frame and sends it to an output port, it actually does two things:

- makes the new data frame available in its output port;
- *activates* all links connected to the output port. This step can also be called *output port activation* (Figure 3).

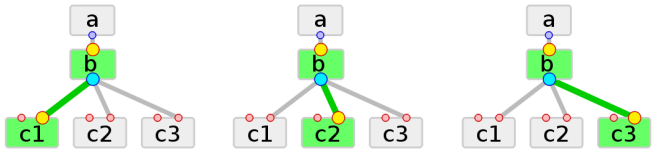


Figure 3. Output port activation (box `b` activates its output port).

Each link connects an output port to an input port, and its activation means sending notification to input port owner box. The notification just says that a new data frame is available at that input port.

When a box receives such a notification, it is free to do whatever it wants to. In some cases, these notifications are ignored; in other cases, they cause box to start processing data and generate output data frames, which leads to link activation again, and the data processing goes one level deeper. For example, the `Pendulum` box has two input ports, `parameters` and `state`. When a data frame comes to `parameters`, the activation notification is ignored (but next time the box will be able to read parameters from that port). When a data frame comes to `state`, the activation is not ignored. Instead, the box computes ODE right hand side and sends it to the output port `oderhs`.

B. Data source box activation

Each simulation must have at least one *data source* box — a box having output ports but no input ports. There can be more than one data source in a simulation.

Data sources can be *passive sources* or *generators*. A generator is a box that can be notified just as a link can be. A passive data source cannot be notified.

A passive data source produces one data frame (per output port) during the entire simulation. The data frame is available on its output port from the very beginning of the simulation.

C. Cancellation of data processing

Link activation notification is actually a function call, and the box being notified returns a value indicating success or failure. If link activation fails, the data processing is *canceled*. This can happen when some box cannot obtain all data it needs from input ports. For example, the `Pendulum` box can process the activation of link connected to port `state` only if there are some parameters available in port `parameters`. If it is so, the activation succeeds. Otherwise, the activation fails, and the processing is canceled.

If a box sends a data frame to its output port, and the activation of that output port fails, the box always cancels the data processing. Notice that this is always done by returning a value indicating activation failure, because the box can only do something within an activation notification.

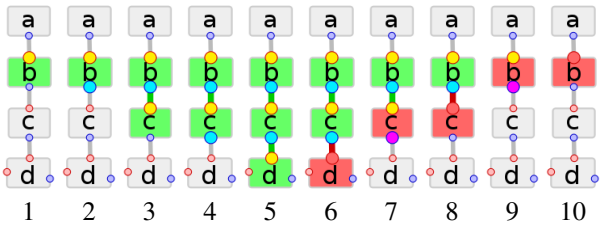


Figure 4. Data processing cancellation.

Figure 4 illustrates the cancellation of data processing: box `a` is the only data source, and its output port is connected to the input port of box `b`. Therefore, the runner activates the input port of `b` (1). Then `b` activates `c` (2, 3), and `c` activates `d` (4, 5). For some reason (e.g., no data on another input port), `d` returns activation failure (6). Callers are obliged to return activation failure as well, therefore the runner finally gets the activation failure (7–10).

D. Initialization of the queue of notifications

When the runner starts data processing, it first considers all data sources and builds the initial state of the *queue of notifications*. For each generator, its notification is enqueued. For each passive data source, the notification of each of its links is enqueued.

E. Processing of the queue of notifications

Then the queue is processed by sending the activation notifications (i. e., calling notification functions) one by one, from the beginning to the end. If a notification call succeeds, the notification is removed from the queue. Otherwise, if the notification call fails (i.e., the data processing gets canceled), the notification is moved to the end of the queue, and the process continues.

The runner processes its queue of notifications until it becomes empty, or maximum number of activation notification failures (currently 100) is exceeded. In the latter case, the entire simulation fails.

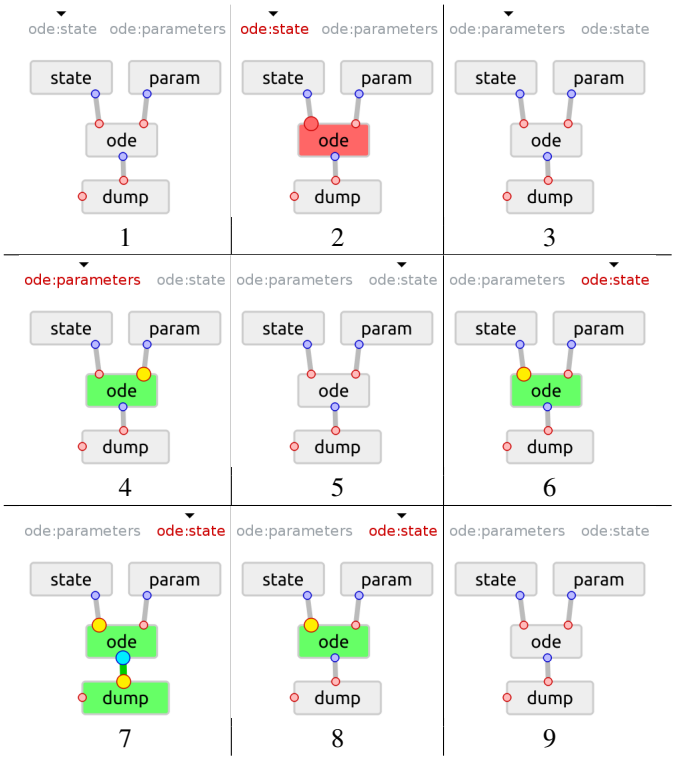


Figure 5. Data processing example.

To illustrate the data processing in a simulation, consider the following example. A box of type `CxxOde` (see Section VII-D10) has two input ports, `state` and `parameters`, and one output port, `rhs`. It ignores activation calls for port `parameters`. On the other hand, the activation of port `state` causes the box to compute ODE right hand side and write it to the output port `rhs`. But the right hand side can only be computed when the parameters are known, i.e., a data frame is available at port `parameters`. Otherwise, the activation of port `state` fails.

Now imagine a simulation with box `ode` of type `CxxOde` and two passive data sources, `state` and `param`, connected to the `state` and `parameters` ports of `ode`, respectively. Besides, the output port of `ode` is connected to the input port of a data storage box.

The runner does not know that `ode` wants parameters before state, so suppose it initializes the initial queue of notifications such that the port `ode:state` is first, and port `ode:parameters` is second. The data processing in this situation is shown in Figure 5 and is as follows.

1) Runner is about to activate port `ode:state`.
2) Runner activates port `ode:state`, and the data processing is canceled by `ode` because there is no data at port `ode:parameters`.
3) Notification for port `ode:state` is moved to the end of the queue. Runner is about to activate port port `ode:parameters`.
4) Runner activates port `ode:parameters`; the activation notification is ignored by `ode`, and control returns back to the runner.
5) Since the activation of `ode:parameters` succeeds, the runner proceeds to next element of the queue, which is `ode:state`.
6) Runner activates port `ode:state`.
7) `ode` computes ODE right hand side and sends it to the output port `ode:rhs`, which leads to the activation of the input port of box `dump`.
8) The `dump` box writes the incoming data frame to a text file and returns control back to `ode`.
9) The box `ode` has nothing more to do in response to the activation of the `state` port, so it returns control back to the runner. There are no more items in the notification queue, and simulation finishes.

F. Post-processing

When the queue of notifications becomes empty, the runner can enqueue *post-processors* before it stops the data processing. The only example of a post-processor is the `Pause` box. Post-processors, like generators, are boxes that can receive activation notifications.

G. User input events

The above process normally takes place during the simulation. In addition, there could be events that break the processing of the queue of notifications. These events are caused by *interactive user input*. Once a user input event occurs, an exception is thrown, which leads to the unwinding of any nested link activation calls and the change of the queue of notifications. Besides, each box gets notified about simulation restart.

The queue of notifications is changed as follows when user input occurs. First, the queue is cleared. Then one of two things happens.

- If the box that threw the exception specifies which box should be activated after restart, the notifications for that box are enqueued (if the box is a generator, its activation notification is enqueued; otherwise, the activation notifications of all links connected to its output ports are enqueued). An input box can only specify itself as the next box to activate, or specify nothing.
- If the box that threw the exception specifies no box to be activated after restart, the standard initialization of the notification queue is done.

After that, the processing of notification queue continues.

There is an important issue that must be taken care of. Simulation can potentially be defined in such a way that its

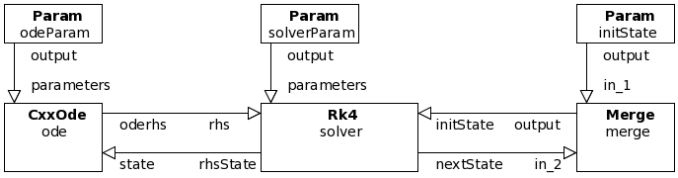


Figure 6. Example of invalid simulation (recursive activation of box `merge`).

execution leads to an infinite loop of recursive invocation of activation notifications. This normally causes program to crash due to stack overflow. In our system, however, some boxes (not all, but only those activating outputs in response to more than one input notification) are required to implement counters for recursive call depth. When such a counter reaches 2, simulation is considered to be invalid and is terminated. This allows to do some kind of runtime validation against recursion at the cost of managing recursive call counters. Figure 6 shows an example of invalid simulation that will detect recursive activation of box `merge`: First, its port `in_1` is activated by runner, which starts numerical integration in `solver`; once the solver outputs next state, it comes to port `in_2` of box `merge`. At this point, `merge` detects recursive activation, because the activation of port `in_1` is still in progress. As a result, the simulation fails.

V. Software architecture overview

This section presents an overview of the architecture of software that implements NumEquaRes.

Essentially, the software consists of the computational core and the web server, and can be deployed by everyone on any server machine running a Linux operating system. Both components are open source, hosted at GitHub (a link to the project is available on the web site [2]).

The computational core is a console application written in C++. Its responsibility is to load simulation specification, run the simulation, and communicate with the controlling process. The communications are necessary for supplying user input and synchronizing with the controlling process.

The web server is written in Node.js and is using several third party packages, most noticeably the express framework [13]. The web server has numerous responsibilities, including the following:

- generating and serving web pages;
- serving files;
- managing user accounts and data;
- managing user sessions;
- handling Ajax [14] requests done by the code running in browser on client machines;
- controlling the computational core.

Notice that web pages sent by the web server to a client contain JavaScript code to be executed by the web browser on the client machine, and that code communicates with the server using the Ajax technology. Therefore, we actually have an application distributed among server and client machines, which is nowadays typical for any web application.

The management of any user data requires a mechanism for persistent data storage. For this purpose, the MongoDB database engine has been chosen.

The interaction between software components is outlined in Figure 7. Large containers represent the server machine, the client machine, and the Internet between them. Rectangular-shaped elements in the containers represent software components that are parts of NumEquaRes or are used by it. Elliptical-shaped elements represent file system folders. Arrows between elements indicate data flow directions; arrow captions explain activities causing the corresponding data transfers.

The detailed discussion of software component architecture is beyond the scope of this paper. However, let us focus on the most important question, namely the interaction between the user, the web server, and the computational core.

User prepares a simulation in the *editor* HTML page. The page is sent by the web server to the client when requested, e.g., through the main menu available in all pages. It contains JavaScript code allowing to design the simulation from scratch or to load an example and further modify it if necessary. When the user prepares a simulation, little interaction with the web server can take place. This is only the case when the user modifies the C++ source code in a box like `CxxOde` (see Section VII-D10) and wants to know if the compilation is successful. Most of the time, no interaction with the server is necessary to prepare a simulation, so this is done locally on the client machine.

Once the user finishes preparing the simulation, the simulation can be saved in the database or run on the server machine. To manage that, the client sends the simulation to the web server as a JSON object within an HTTP request. Handling these requests, the web server either interacts with the database or runs the computational core, depending on the request. In the latter case, the JSON object describing the simulation is passed to the computational core through its standard input stream.

The computational core is a console application that implements a simple text protocol allowing the web server to interact with it. Writing specific lines of text to the standard input causes some commands to be executed, like start or stop the simulation, provide interactive input data, etc. On the other hand, the server reads the standard output of the computational core. Importantly, when the simulation starts, the core writes *annotations* for output files and input controls there — the web server sends these annotations to the client, so the client knows which files need to be requested as simulation results, and which elements need to be created for obtaining user input. Other important things written by the computational core to its standard output are the *synchronization markers*. Once the core writes an output file, it also writes such a marker to the standard output and waits for the corresponding marker on its standard input. At this point, the server reads the marker from the core, informs the client about the update of the output file, and then writes the synchronization marker to the standard input of the computational core. The core reads the marker and resumes the execution of the simulation. Notice that the marker based synchronization is not frequent (e.g., once per second, or other user-specified time period), therefore it does not impact the overall performance.

Notice also that when a simulation is running, the web server sends data from the computational core to the client using HTML5 Server Sent Events [7].

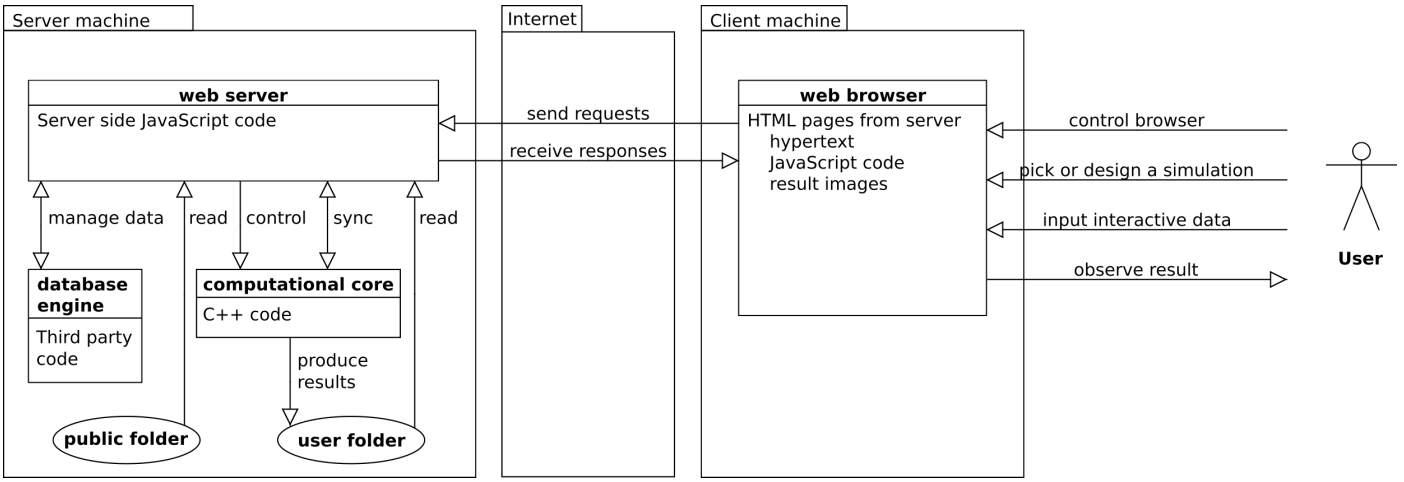


Figure 7. Interaction among software components of NumEquaRes and users.

When the user provides interactive input data for the running simulation, the browser sends requests to the web server; handling them, the server writes corresponding commands to the standard input of the computational core, so the core knows what the user input is. The computational core reads the standard input stream in a separate thread, which is synchronized with the main worker thread not too frequently in order not to impact the performance (see next section).

VI. Performance, extensibility, and ease of use

As stated in Section I, computational performance and functional extensibility are considered important design features of the NumEquaRes system. This section provides technical details on what has been done to achieve performance and support extensibility. Last subsection highlights design features that make system easier to use.

A. Performance

To achieve reasonable performance, it is not enough to just use C++. Some additional design decisions should be made. Most important of them are already described above. The ability to organize simulation workflow arbitrarily allows to achieve efficient memory usage, which is illustrated by an example in Section I. A number of specific decisions made in the design of NumEquaRes core are targeted to high throughput. They are driven by the following rules.

- Perform simulation in a single thread. While this is a serious performance limitation for a single simulation, we have made this decision because the simulation runs on the Web server, and parallelization inside a single simulation is likely to impact the performance of server, as it might run multiple simulations simultaneously. And, on the other hand, single thread means no synchronization overhead.
- No frequent operations involving interaction with operating system. Each box is responsible for that. For example, data storage boxes should not write output data to files or check for user input frequently. The performance might drop even if the time is measured using `QTime::elapsed()` too frequently.
- No memory management for data frames within activation calls. In fact, almost 100% of simulation time is spent in just one activation call made by runner (during that call, in turn, other activation calls are made). Therefore, memory management outside activation calls (e.g., the allocation of an element of the queue of notifications) is not a problem. Still some memory allocation happens when a box writes its output data, but this is not a problem as well, since such operations are not frequent.
- No movement of data frames in memory. If a box produces an output frame and makes it available in its output port, all connected boxes read the data directly from memory it was originally written to. This item and the previous one both imply that there are nothing like queues of data frames, and each frame is processed immediately after it is produced.
- No virtual function calls within activation calls. Instead, calls by function pointer are preferred.

A simple architecture of classes has been developed to comply with the rules listed above and, in the same time, to encapsulate the concepts of box, port, link, and others. These classes are split into ones for use at the initialization stage, when simulation is loaded, and others for use at simulation run time. First set of classes may rely on Qt object management system to support their lifetime and the exposure of parameters as JavaScript object properties. Classes of the second set are more lightweight; their implementations are inlined whenever possible and appropriate, in order to reduce function call overhead.

Although NumEquaRes core performance has been optimized in many aspects, it seems impossible to combine speed and flexibility. Our experience with some examples indicates that hand-coded algorithms run several times faster than those prepared in our system.

B. Extensibility

The functionality of NumEquaRes mostly resides in boxes. To add a new feature, one thus can write code for a new box. Boxes are completely independent. Therefore, adding a

new one to the core simply boils down to adding one header file and one source file and recompiling. The core will be aware of the presence of the new box through its box factory mechanism. Next steps are to support the new box on server by adding some meta-information related to it (including user documentation page) and some client code reproducing the semantics of port format propagation through the box. The checklist can be found in the online documentation.

Some extensions, however, cannot be done by adding boxes. For example, to add 3D visualization, one needs to change the client-side JavaScript code. We are planning to simplify extensions of this kind; however, this requires refactoring of current client code.

C. Ease of use

First of all, NumEquaRes is an online system, so user does not have to download and install any software, provided user already has a Web browser. All user interaction with the system is done through the browser.

To formulate a simulation as a data processing algorithm, user composes a scheme consisting of boxes and links, and there is no need to code.

Online help system contains a detailed documentation page for each box; it also explains simulation workflow, user interface, and other things; there is one step-by-step tutorial.

To prepare a simulation, user can find a similar one in the database, then clone it and modify. User can decide to make his/her simulation public or private; public simulations can be viewed, run, and cloned by everyone. To share a simulation with a colleague, one shares a hyperlink to it; besides, simulations can be downloaded and uploaded.

Currently, user might have to specify part of simulation, such as ODE right hand side evaluation, in the form of C++ code. We understand this might be difficult for people not familiar with C++. To mitigate this problem, there are two features. Firstly, each box that needs C++ code input provides a simple working example that can be copied and modified. Secondly, NumEquaRes supports the concept of *code snippets*. Each piece of C++ input can be given a documentation page and added to the list of code snippets. These snippets can be created and reused by everyone.

VII. Simulation building blocks

This section explains the semantics of different boxes from which NumEquaRes simulations are built. There are currently 40 types of boxes; this section categorizes them and describes most important boxes. Knowing how the boxes work gives understanding of NumEquaRes simulations design.

In this section, we introduce the notation `Box:port`, where `Box` is the type of a box, and `port` is a name of one of its ports. For example, `Param:output` means the `output` port of a box of type `Param`; the part `Box:` is omitted when the box type is obvious from context.

A. Source boxes

There are three types of boxes that can be used as data sources: `Const`, `Param`, and `ParamArray`. The `Const` and `Param` boxes behave identically once their parameters are specified. A box of type `Const` or `Param` generates just one data frame (see Section III) per simulation run. The format of the frame is a one-dimensional array. The contents of the array is determined by fixed parameters of the box. So what user enters as parameters is turned by the box into a data frame.

The difference between `Param` and `Const` is how they manage their frame format. The `Param` box expects its format to be specified in a port connected to its `output` port. Therefore, before parameters can be specified for a `Param` box, it has to be connected to something providing a data format. For example, if there is a link `Param:output -> Pendulum:parameters`, the format of data frame generated by `Param` will be $\{2\}$ — an array of two elements which are the parameters of a pendulum: the length, l, and the acceleration of gravity, g. The parameters of the `Param` box will be l and g — see Figure 16 (a).

In contrast to `Param`, a box of type `Const` allows user to enter an arbitrary one-dimensional array of parameters. The data format of the box is determined by the user-specified array of parameters.

The `Param` type should typically be preferred to `Const` because its use guarantees format compatibility along the link between `Param:output` and another port. `Const` should only be used when connected port does not provide a frame format. Notice also that if `Param` is connected to more than one port, and these ports provide different sets of element labels, the box cannot be used because it cannot resolve parameter names.

The `ParamArray` box type is similar to `Param`, but it allows user to specify an array of sets of parameters. When user specifies an array of length n, the box generates n output data frames per simulation run. Combining `ParamArray` with an `Interpolator` box (see Section VII-E1 and Figure 18 (b)) allows to generate many data frames in which parameters gradually change between values specified in `ParamArray`. In addition, `ParamArray` box has port `flush`. The box writes an empty data frame to that port after it writes all data frames to `output`.

B. Data storage boxes

There are two types of boxes that store incoming data frames: `Dump` and `Bitmap`.

Each data storage box in a simulation corresponds to a file generated in user's directory on server. When the simulation runs, user sees these files in browser. Text files appear as tables and can be downloaded as text; image files are seen as images — see Figure 8.

A box of type `Dump` stores data frames of any format coming to its `input` port in a text file. The incoming frames are output to the file as lines of formatted decimal numbers. There is a limitation of 10^6 on total number of scalar values written to the file in order to avoid occasional generation of a large file on server.

A box of type `Bitmap` stores incoming data frames in an image file in the PNG or JPEG format. Each data frame coming to port `Bitmap:input` is transformed into a colored image as follows. The format of input data frame should be $\{w, h\}$, where w is the width, and h is the height of the image, in pixels. Each scalar element of incoming 2D data frame is transformed into a 32-bit RGB color value using the *color map*. The color map is a mapping from scalar value to color value; it is specified as a parameter of the `Bitmap`

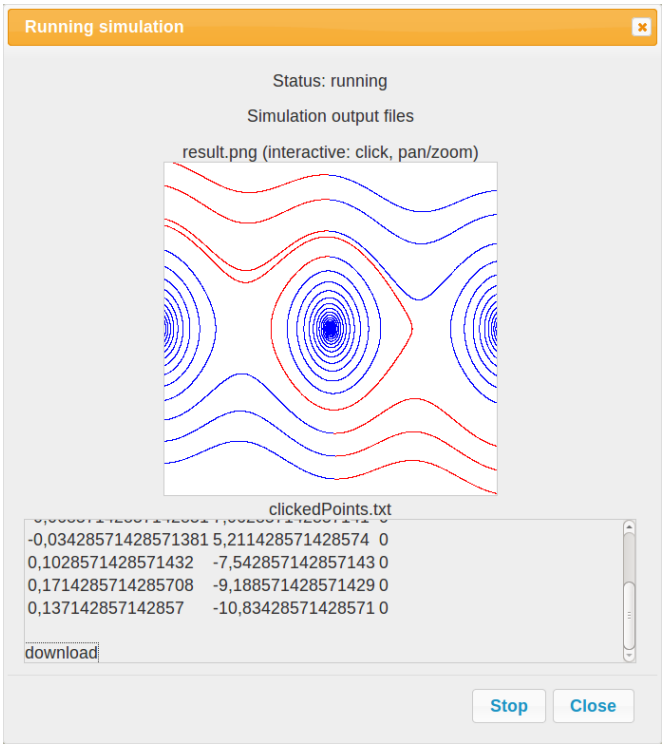


Figure 8. Simulation output files generated by storage boxes.

box. There is a limitation of 2000 pixels on w and h in order to avoid occasional generation of large files on server. Notice that `Bitmap` boxes usually receive data frames from `Canvas` boxes — see below.

C. The `Canvas` box

The `Canvas` box provides intermediate 2D array to store scalar values, e.g., for further image generation. It is initially filled with zero values. The box has several input ports, one output port, and a set of parameters.

The geometry of canvas is determined by its *range* and *resolution* in each of its two dimensions. The range is a pair of numbers $\{x_{\min}, x_{\max}\}$ for the x dimension and $\{y_{\min}, y_{\max}\}$ for the y dimension; the resolution is the number of array elements, N_x or N_y — see Figure 9 (a). Ranges and resolutions are canvas parameters; ranges can also be supplied through port `range`, such that each frame is $[x_{\min}, x_{\max}, y_{\min}, y_{\max}]$.

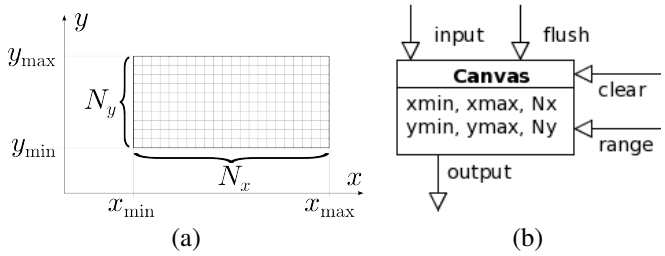


Figure 9. (a) Canvas geometry; (b) `Canvas` box ports and parameters.

The box receives input data at port `input`. The format must be $\{2\}$ or $\{3\}$ — a one-dimensional array of size 2 or 3. First two elements of an incoming data frame are the coordinates x, y of a point. The third element, if present, is a scalar value v; it defaults to 1 if absent. For each input data frame, the box computes canvas coordinates x_c, y_c using the formula

$$x_c = \left\lfloor N_x \frac{x - x_{\min}}{x_{\max} - x_{\min}} \right\rfloor, \qquad y_c = \left\lfloor N_y \frac{y - y_{\min}}{y_{\max} - y_{\min}} \right\rfloor,$$

and writes v into the array using x_c and y_c as indices, if they are valid ($0 \le x_c < N_x$, $0 \le y_c < N_y$).

The output of canvas data occurs either when user-specified timeout is exceeded, or when a data frame comes to port `flush`. Output data frame contains all values currently stored in the 2D array and has format $\{N_x, N_y\}$.

There is also port `clear`; when it receives a data frame, all elements of the internal array assign zero values. All box ports are shown in Figure 9 (b).

D. Boxes for simple transformations and filters

Many boxes have quite simple logics, producing one output data frame in response to incoming data frames. They have a primary input port, an output port, and optional input ports; they can also have parameters controlling their behavior. Below we briefly describe some of such simple boxes.

1) CountedFilter: The box passes to port `output` each n-th data frame coming to port `input`; n is the parameter of the box received through input port `count`.

2) Counter: The box counts data frames received in port `input`. Each time the counter value increases, it is sent to port `count`. The counter value can be set to zero by sending a data frame to port `reset`.

3) CrossSection: Data frames received at port `input` are one-dimensional arrays of length n. The elements of i-th data frame are interpreted as coordinates $[x_0^i, x_1^i, \ldots x_{n-1}^i]$ of point $\mathbf{x}^i$; the points $\mathbf{x}^i$ are considered to be consecutive points on a piecewise-linear curve in n-dimensional space. The box outputs points of intersection of the curve and the hyperplane $x_k = c$, where k and c are box parameters (Figure 10). Another box parameter allows to count only intersections with x_k increasing along the curve or decreasing along the curve. The `CrossSection` box is crucial for simulations that visualize Poincaré maps.

Figure 10. `CrossSection` box input and output.

4) IntervalFilter: The box is similar to `CrossSection`, but instead of one hyperplane $x_k = c$ it considers many hyperplanes, specified by equation $x_k = c + Tm$, where m is an arbitrary integer number, and T is a box parameter. The box considers that x_k increases monotonously in incoming points, since it is usually the time. The box can be used to visualize Poincaré maps in systems with periodic excitation.

5) Differentiate: The box computes differences between consecutive points $\mathbf{x}^i$, $\mathbf{x}^{i+1}$ coming to port `input`:

$$\mathbf{d}^i = \mathbf{x}^{i+1} - \mathbf{x}^i$$

The differences $\mathbf{d}^i$ are written to port `output`.

6) Scalarize: For each data frame $\mathbf{x} = [x_0, x_1, \dots x_{n-1}]$ received at port `input`, the box generates a scalar value v and writes it to port `output`. The method used to compute the scalar is the box parameter; user can choose it among several common norms, minimum, and maximum.

7) Projection: The box accepts input data frames of arbitrary format at its `input` port. Each data frame is interpreted as an array of values, $x_0^{in}, \dots x_{n-1}^{in}$, where n is the total number of elements in the incoming data frame. Once an input data frame is received, an output data frame is generated and written to port `output`. The output data frame contains m elements $x_0^{out}, \dots x_{m-1}^{out}$ and has format $\{m\}$. The elements of the output data frame are picked from input as follows:

$$x_k^{out} = x_{i_k}^{in}, \quad k = 0, \dots m-1.$$

In the above formula, the indices i_k are box parameters. The box is often used, for example, to pick two variables from a vector for plotting on `Canvas` (see Section VII-C and Figure 18 (a)).

8) Eigenvalues: The box expects a square matrix at its input port `matrix`, so the port format is $\{n,n\}$. As soon as a matrix is obtained, its eigenvalues are computed. The real parts of the eigenvalues are then written to output port `eig_real`, and imaginary parts are written to port `eig_imag`.

The implementation of this box uses the ACML library [15].

9) ThresholdDetector: The box receives a scalar-valued data frames, x, at port `input`. For each incoming value, the value v is computed as follows: the logical expression $x * T$ is evaluated; if the result is true, v is set to one; otherwise, v is set to zero. In the above expression, the binary operator $*$ can be one of $<, \leq, \geq, >, =, \neq$ and is determined by box parameter; the threshold value T can be either specified as a box parameter or passed in through port `threshold`.

Once v is computed, it is normally written to port `output`. For more flexibility, the box allows to suppress the output of zero values of v by specifying another box parameter, `quiet`. Values $v = 1$ are always written to `output`.

10) Other transformations: There are a number of other boxes that perform transformations. They all write a data frame $\mathbf{y}(\mathbf{x})$ to the output port as soon as they obtain a data frame $\mathbf{x}$ at the input port. There could be an additional input port for parameters. Here these box types are listed.

- `CxxFde` — a user-defined transformation. The box receives $\mathbf{x}$ at port `state` and writes $\mathbf{y}$ to port `nextState`. Both $\mathbf{x}$ and $\mathbf{y}$ are vectors of length n. The transformation is defined by user in the form of C++ source code. The code can also describe parameters to be obtained at input port `parameters`. The box is designed primarily for use with the `FdeIterator` box (see Section VII-E3) as the source of a system of finite difference equations.
- `CxxOde` — another user-defined transformation. The box receives $\mathbf{x}$ at port `state` and writes $\mathbf{y}$ to port `rhs`. The vector $\mathbf{x}$ contains n state variables and the time: $\mathbf{x} = [x_1, \dots x_n, t]$. The vector $\mathbf{y}$ contains n time derivatives of state variables: $\mathbf{y} = [\dot{x}_1, \dots, \dot{x}_n]$. The transformation and additional parameters to be obtained at input port `parameters` are defined by user in the form of C++ source code. The box is designed primarily for use with the `Rk4` box (see Section VII-E4) or other future solvers as the source of a system of ordinary differential equations.
- `CxxTransform` — yet another user-defined transformation. It gives user freedom to select arbitrary formats of data frames for $\mathbf{x}$ and $\mathbf{y}$. The transformation and optional parameters are also specified in the form of C++ code. The box can be used, e.g., to formulate a linear system of ordinary differential equations to further investigate the dependency of its stability on parameters with the `Eigenvalues` box (see Section VII-D8).
- `Pendulum`, `DoublePendulum`, `Mathieu`, `VibratingPendulum` — these are examples of hard-coded systems of ordinary differential equations; the logics and sets of ports in each of these boxes are the same as in the `CxxOde` box, therefore, they are interchangeable with `CxxOde`.

Notice that since simulations run on server side, C++ source code specified by user for boxes `CxxFde`, `CxxOde`, `CxxTransform`, is compiled and run on server. This creates potential security problem — running malicious code on server. The problem is addressed in Section X-B.

E. Iterators and solvers

Boxes described in this section are essentially iterators. The implementation of such a box contains a loop in its activation handler function, and output data frames are generated inside the body of the loop. Due to this, the activation of an input port can cause the generation of many output data frames.

1) Interpolator: Data frames received at port `input` are one-dimensional arrays of length n. The elements of i-th data frame are interpreted as coordinates $[x_0^i, x_1^i, \dots x_{n-1}^i]$ of point $\mathbf{x}^i$ in n-dimensional space. For each pair of consecutive points $\mathbf{x}^i$, $\mathbf{x}^{i+1}$, the box generates $N-1$ intermediate points $\mathbf{x}^{i,1}, \dots, \mathbf{x}^{i,N-1}$ using the formula

$$\mathbf{x}^{i,k} = (1 - t_k)\mathbf{x}^i + t_k \mathbf{x}^{i+1}, \quad t_k \equiv \frac{k}{N}, \quad k = 1, \dots N-1.$$

All points, including original $\mathbf{x}^i$ and interpolated $\mathbf{x}^{i,k}$ are passed to port `output`. N above is the number of interpolation intervals; it is a parameter of the box. Interpolator input and output are shown in Figure 11.

Figure 11. `Interpolator` box input and output.

2) GridGenerator: This box produces a d-dimensional grid of values for each data frame coming to its `input` port. Input data frames must be one-dimensional arrays.

For each input data frame, the grid generator produces, N output frames, $N = N_0 N_1 \dots N_{d-1}$, where N_k is the grid size in k-th dimension, $0 \leq k < d$.

The grid consists of points $\mathbf{x}_I$. Each point of the grid is identified by multi-index $I = i_1, i_2, \dots, i_d$ (indices i_k run from 0 to $N_k - 1$) and has coordinates $x_I^1, x_I^2, \dots, x_I^d$. The

coordinates x_I^k are computed by linear interpolation between parameters $x^{k,\min}$, $x^{k,\max}$:

$$x_I^k = \left(1 - t_I^k\right) x^{k,\min} + t_I^k x^{k,\max}, \quad t_I^k \equiv \frac{i_k}{N_k - 1}.$$

Notice that $x^{k,\min}$, $x^{k,\max}$ define *ranges*, just like for `Canvas` (see Section VII-C); they can either be specified directly as box parameters or supplied through input port `range`.

For each grid point, a data frame is generated and written to port `output` (Figure 12). The format of ports `input` and `output` is the same. The elements in the output data frame repeat those from the input data frame, except that d elements are replaced by coordinates x_I^k. The indices of replaced elements are box parameters.

Figure 12. `GridGenerator` box input and output.

In addition, the box generates empty data frame and sends it to port `flush` as soon as all output data frames for one input data frame are generated.

The `GridGenerator` box is very useful when it is necessary to repeat the same operation for parameters varying in certain ranges. One of its applications is the generation of stability diagrams (see Figure 24).

3) FdeIterator: As follows from the box name, it performs iterations of finite difference equations (FDE). The equations are formulated outside the box and should be connected to ports `fdeIn`, `fdeOut`.

Suppose that the state of a discrete-time system at k-th time step is described by vector $\mathbf{x}_k$. The explicit form of FDE gives the formula to compute the state of the system at next time step:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k).$$

To evaluate this formula, the `FdeIterator` writes $\mathbf{x}_k$ to port `fdeOut` and afterward expects that $\mathbf{x}_{k+1}$ has come to port `fdeIn`. The iterations proceed by reading data frames from `fdeIn` and writing them to `fdeOut`. In addition, data frames are written to output port `nextState` — this way we normally make use of the resulting points $\mathbf{x}_k$. For more flexibility, parameters n_o and n_s can be specified for the box, that control which points are written to port `nextState`: n_s is the number of initial points to skip, and n_o is the number of time steps between consecutive outputs. For example, if $n_s = 10$, $n_o = 2$, the points written to `nextState` are $\mathbf{x}_{10}, \mathbf{x}_{12}, \mathbf{x}_{14}, \ldots$ Notice also that setting the value of n_o to zero causes only the last system state to be written to `nextState`.

The iterative process is initiated by sending the initial state of the system, $\mathbf{x}_0$, to port `initState`. The process is controlled by parameters, supplied through port `parameters`, which are n_s and n_o described above, and the total number of iterations, n. If n is zero, the iterations never end. However, sending any data frame to port `stop` causes the iteration loop to terminate.

When iterations finish, an empty data frame is sent to port `finish`.

Ports of the `FdeIterator` box are shown in Figure 13 (a).

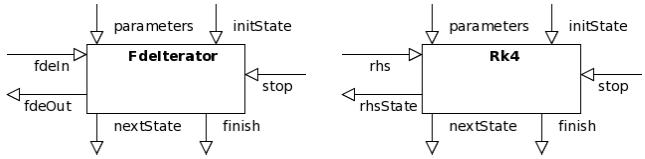


Figure 13. (a) `FdeIterator` box ports; (b) `Rk4` box ports.

4) Rk4: This box has logics very similar to that of `FdeIterator`, but the box is designed to obtain numerical solution of a system of ordinary differential equations (ODE). The ODE system in the normal form is specified by formula

$$\dot{\mathbf{x}} = f(\mathbf{x}, t),$$

where $\mathbf{x}$ is the state vector, t is the time, and $(\ldots)\dot{}$ is the time derivative.

The box implements the well known Runge — Kutta explicit 4-th order numerical integration scheme [16], hence its name. It has ports similar to ports of `FdeIterator`, but the ports for interaction with the ODE system are named `rhsState` and `rhs`, and they are slightly different from `FdeIterator:fdeOut`, `FdeIterator:fdeIn`. To evaluate ODE right hand side, the `Rk4` box writes a data frame containing system state vector $\mathbf{x}$ and the time t to port `rhsState`; it then expects the vector $f(\mathbf{x}, t)$ in port `rhs`.

Parameters of the `Rk4` box supplied through port `parameters` are the time integration step, the number of steps to perform, and n_o (see Section VII-E3).

All ports of the `Rk4` box are shown in Figure 13 (b).

5) LinOdeStabChecker: The box is designed to analyze the stability of linear ODE systems with periodic coefficients and zero right hand side:

$$\dot{\mathbf{y}} = \mathbf{A}(t)\mathbf{y}, \quad \mathbf{A}(t+T) = \mathbf{A}(t),$$

where $\mathbf{y} = [y_1, \ldots, y_n]^T$ is the vector of n state variables, t is the time, and $\mathbf{A}$ is an $n \times n$ matrix of coefficients, which are considered to be periodic functions of time, with period T.

The stability analysis is done as follows [5]. Take n initial states at $t = 0$, $\mathbf{y}^1(0), \ldots \mathbf{y}^n(0)$ such that

$$\mathbf{y}^k(0) = \left[y_1^k(0), \ldots y_n^k(0)\right]^T, \quad y_k^s(0) = \delta_k^s \equiv \left[\begin{array}{ll} 1, & \text{if } s = k \\ 0, & \text{if } s \neq k \end{array}\right.$$

In other words, vectors $\mathbf{y}^k(0)$ make up the $n \times n$ identity matrix: $\mathbf{Y}(0) \equiv \left[\mathbf{y}^1(0), \ldots \mathbf{y}^n(0)\right] = I$, $I_{ks} = \delta_{ks}$. For each initial state $\mathbf{y}^k(0)$, the initial value problem is solved and $\mathbf{y}^k(T)$ are obtained. Then the stability is determined by characteristic multipliers ρ_k — the eigenvalues of the monodromy matrix (system fundamental matrix computed at period T):

$$\mathbf{M}\mathbf{z}_k = \rho_k \mathbf{z}_k, \quad \mathbf{M} = \mathbf{Y}(T) \equiv \left[\mathbf{y}^1(T), \ldots \mathbf{y}^n(T)\right]$$

The solution is stable if the absolute value of each multiplier does not exceed 1, and is unstable if there is at least one k such that $|\rho_k| > 1$.

Practically, in many cases, for multipliers we have $|\rho_k| = 1$ (for all k) if the system is stable, and $|\rho_k| > 1$ (for one or more k) if the system is unstable. For such systems, numerical solution will most likely always give $1 < |\rho_k| < 1 + \varepsilon$ if the system is stable, where $\varepsilon \ll 1$ is a small value. Therefore, the stability detection is based on checking inequality $|\rho_k| < 1+\varepsilon$ rather than $|\rho_k| \leq 1$, and $\varepsilon = 10^{-5}$ is a hardcoded constant.

The box does not need to know the period T; however, the solver connected to the box should return the state $\mathbf{y}(T)$ when given initial state $\mathbf{y}(0)$.

The box is connected to an ODE solver by output port `initState`, to pass initial state $\mathbf{y}(0)$ to it, and by input port `solution`, to obtain the system state $\mathbf{y}(T)$. Since solver implementation typically involves the use of `Rk4` box, data frames at these ports actually include the time as well, so a data frame contains values $y_1, \ldots, y_n, t$.

The stability analysis is performed as soon as any data frame comes to input port `activator`. After that, the result is written to output port `result`. The output value is 1 if the system is stable, and 0 if unstable.

F. Boxes that have specific logics

This section describes some boxes that implement specific logics. Attempts to design certain simulations have led us to the invention of these boxes. We are not sure that the presented set of such logical boxes is complete in some sense, and that there is no better way to design them. Still the logic boxes are extremely useful in some simulations.

1) Join: The box has two input ports, `in_1` and `in_2` with formats $\{n_1\}$ and $\{n_2\}$, respectively, and one output port, `out`, with format $\{n_1 + n_2\}$. In short, the box glues together each two data frames coming to the input ports and writes the result to the output port.

Suppose that the input data frame at port `in_1` has elements $x_0^{in,1}, \ldots x_{n_1-1}^{in,1}$, and the input data frame at port `in_2` has elements $x_0^{in,2}, \ldots x_{n_2-1}^{in,2}$. Then the output data frame consists of all elements of data frame at port `in_1`, followed by all elements of data frame at port `in_2`: $x_0^{in,1}, \ldots x_{n_1-1}^{in,1}, x_0^{in,2}, \ldots x_{n_2-1}^{in,2}$.

The box has two internal boolean state variables, s_1 and s_2, indicating that an unprocessed data frame is pending at input ports `in_1` and `in_2`, respectively. The value of true means that an input data frame has been received but has not been processed so far. The value of false means that there were no data frames at all, or the last received input data frame has already been processed.

When an input data frame comes to port `in_1` or `in_2`, the value of state variable s_1 or s_2, respectively *must be* false (otherwise, simulation stops with the error message saying "Join box overflow"). Then, the state variable is set to true. After that, if the other state variable (s_2 or s_1, respectively) is false, the processing finishes — the box will be waiting for a data frame at the other input port. If both state variables s_1 and s_2 are true, the box resets them to false, generates one output frame, and writes it to port `out`.

The logics of the box ensures that k-th output data frame at port `out` is generated from k-th data frame at input port `in_1` and k-th data frame at input port `in_2`.

Notice that to satisfy the requirements of the `Join` box on the order of input data frames and ensure no overflow error, it is often used in combination with the `Replicator` box (see Section VII-F3 below).

Figure 14 (a) illustrates data processing by a `Join` box.

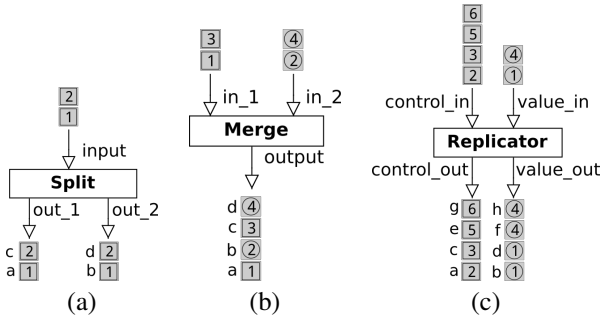


Figure 14. Input and output in (a) `Join`, (b) `Merge`, and (c) `Replicator` boxes. Numbers denote the order of input data frames and identify them; letters denote the order of output data frames.

2) Merge: The box has several input ports, `in_1`, `in_2`, etc. The number of input ports is a box parameter. There is one output port, `output`. All ports have the same format, which can be arbitrary. Once the box obtains a data frame at any of its input ports, it writes it to the output port immediately; see Figure 14 (b).

The `Merge` box is used to collect data frames from different ports.

3) Replicator: It is not obvious that there is a need for this box at all, but it is often really needed in simulations. The box has two input ports, `control_in` and `value_in`, and two output ports, `control_out` and `value_out`. It passes *control data frames* from `control_in` to `control_out` and *value data frames* from `value_in` to `value_out`.

When a value data frame comes to `value_in`, nothing happens. In contrast to that, when a control data frame comes to `control_in`, the box writes the incoming control data frame to `control_out`, and then it writes the previously received value data frame to port `value_out`, as shown in Figure 14 (c). If no value data frame has been received before control data frame, the processing is canceled.

As already mentioned, the `Replicator` box can be combined with the `Join` box to synchronize data frames; but it has many more different applications.

4) Split: The box has one input port, `input`, and several output ports, `out_1`, `out_2`, etc. The number of output ports is a box parameter. All ports have the same format, which can be arbitrary. Once the box receives a data frame at port `input`, it writes it to output ports `out_1`, `out_2`, etc. Importantly, the order of output port activation is guaranteed — first `out_1`, then `out_2`, and so on. Figure 15 (a) illustrates the data processing by the `Split` box.

Simulations in NumEquaRes allow to connect an output port of a box to any number of input ports of other boxes. However, this introduces uncertainty into simulation, because

when such an output port is activated, the order of activation of connected input ports is undefined (it is actually the order of link creation, but it cannot currently be seen or easily modified). The `Split` box potentially allows to design simulations that have no multiple connections of output ports at all: each multiple connection can be replaced by single connection to `Split:input` and several single connections to `Split:out_1`, `Split:out_2`, etc.

Practically, the order of activation of input ports connected to the same output port is not always important. When it is, the `Split` box should be used.

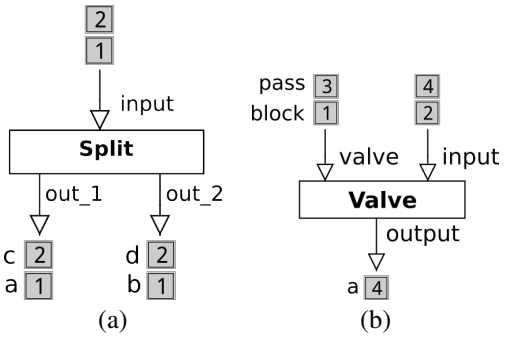


Figure 15. Input and output in (a) `Split` and (b) `Valve` boxes.

5) Valve: The box contains two input ports, `valve` and `input`, and one output port, `output`. The `valve` port accepts scalar *controlling values*; other two ports should have the same format, which can be arbitrary. In short, the box passes a data frame coming to its `input` port to the `output` port only if it has a nonzero value in the `valve` port; otherwise, the input data frame is not passed — see Figure 15 (b).

The logics of this box is similar to that of `Join` (see Section VII-F1) but slightly differs from it. Internally, the box holds two boolean state variables, s^i and s^v, indicating if there are unprocessed data frames at ports `input` and `valve`, respectively. Initially, they are both false.

When a controlling data frame comes to port `valve`, s^v is set to true (note: in contrast to the `Join` box, there is no requirement that s^v should be false at this moment). Current controlling value v is set to true if the controlling data frame element is nonzero, and to false otherwise. Then, if s^i is true, further processing is done: last data frame received at port `input` is written to port `output` if v is true and not written if v is false. Then s^v and s^i are both set to false.

When an input data frame comes to port `input`, s^i is set to true (note: in contrast to the `Join` box, there is no requirement that s^i should be false at this moment). Then, if s^v is also true, further processing is done exactly the same way as explained above: last data frame received at port `input` is written to port `output` if v is true and not written if v is false. Then s^v and s^i are both set to false.

G. Input boxes

NumEquaRes provides several box types dedicated to interactive input of data.

1) General behavior of input boxes: All user input is non-blocking, which means that input boxes never wait for user input. On the other hand, an input box can only check if there is an input event when one of its input ports is activated. Most of input boxes have the `activator` port specifically for this.

Another way to activate an input box is to use the special `Pause` box (see Section IV-F) — in that case, all input boxes are activated when data processing finishes.

Once an input box receives user input data, it takes an action that depends on box type. For example, it can write a data frame to the port `output`, or it can restart simulation.

Among input box types, three of them (`SimpleInput`, `RangeInput`, and `PointInput` — see below) allow user to interactively input *vector data*. They all have the same logics, and only differ in how user inputs the data.

All vector data input boxes remember the data user entered within the last input event (or, at the beginning of simulation, they know that no input events have taken place).

Vector data input boxes have input port `input`. The format of this port is a one-dimensional array of arbitrary size. There is also the `output` port with the same format. Besides, vector input data boxes have the `activator` port.

When a data frame comes to port `input`, a data frame is written to port `output`. The output data is the same as the input data if no user input has taken place on this box yet. If, however, there was user input, the box changes part of the input data frame before writing it to `output`: it replaces some elements of input data frame with values user entered last time. Which elements are replaced depends on box parameters.

When a vector data input box is activated (either by sending a data frame to port `activator` or due to the activity of the `Pause` box), it first checks if any data is available at port `input`. If no data has been received on that port, nothing happens. User input data, if any, will be waiting for further processing, till the box is activated next time.

If some data is available at port `input`, the box checks for user input. If there is no unprocessed user input, nothing happens. If user input has taken place, the box reads the user input data and replaces part of last data frame obtained from `input` with new user input data. The resulting data frame will be available at port `data`, but the exact behavior of the box now depends on two boolean parameters, `restartOnInput` and `activateBeforeRestart`.

- If `restartOnInput` is false, the simulation data processing loop is exited and entered again, starting from the input box. The input box then writes the prepared output data frame to port `output`, and simulation continues.
- If `restartOnInput` is true, then
  - if `activateBeforeRestart` is true, the prepared output data frame is sent to port `output`; otherwise, it is not sent.
  - Then simulation data processing loop is exited and entered again, starting from data sources, as it happens when simulation is started (see Section IV).

Notice that the combination `restartOnInput`=true and `activateBeforeRestart`=true implies that there will be

no extensive data processing when the data frame is sent to port `output` before restarting (otherwise, there probably will be no restart at all). This combination can be used, for example, to specify ODE solver parameters: when solver receives them, it does nothing. More often both `restartOnInput` and `activateBeforeRestart` are false.

2) SimpleInput: Boxes of this type allow user to enter numeric values. Box parameters specify the display names for these values and the indices in the output data frame where these values are written to.

When a simulation having boxes of this type is running, user sees a set of named input fields. Entering a value into such a field causes user input event, which is processed as described above.

3) RangeInput: The box is similar to `SimpleInput`, but instead of entering numeric values user moves sliders. For each input value, it is necessary to specify value range and resolution in addition to the display name and index.

4) PointInput: The box allows user to enter coordinates x, y of points in plane by clicking on an image that corresponds to the `Bitmap` box (see Section VII-B) associated with `PointInput`. The coordinates x^{img}, y^{img} of pixel clicked on the image (notice that the point $x^{img} = y^{img} = 0$ is at the top-left corner of the image) are mapped to x, y using linear interpolation:

$$\begin{aligned} x &= x_{\min} + \frac{x^{img}}{N_x} (x_{\max} - x_{\min}), \\ y &= y_{\min} + \frac{N_y - y^{img}}{N_y} (y_{\max} - y_{\min}), \end{aligned}$$

where N_x and N_y are image pixel width and height, respectively.

Parameters $x_{\min}$, $x_{\max}$, $y_{\min}$, $y_{\max}$ determine the rectangle that the entire image maps onto. They can be specified as box parameters or supplied through additional input port `range`.

Other parameters of the box are the name of image file and the indices of elements in output data frames where x and y are written to.

5) RectInput: The box allows user to enter two data ranges that determine translation and scaling of a plane. These ranges are specified by parameters $x_{\min}, x_{\max}, y_{\min}, y_{\max}$. When user input occurs, the box computes new ranges and writes them to port `output` as one data frame. Several boxes described above (`GridGenerator`, `Canvas`, `PointInput`) have port `range` compatible with `RectInput:output` and can be connected to it.

Similarly to `PointInput`, a `RectInput` box must be associated with a `Bitmap` box by providing the name of image file. The input comes to `RectInput` when user rotates the mouse wheel on the corresponding image (this causes scaling) and drags across that image (this causes panning).

The `RectInput` box is used when basic pan/zoom functionality is desired for a generated image, e.g., to explore fractals (see Figure 26).

6) SignalInput: This is the simplest input box. A button is displayed for each box of this type at simulation run time. Pressing the button causes an empty data frame to be written to port `output`.

The box can be used to trigger some actions, for example, to clear `Canvas` (see Section VII-C) by connecting `SignalInput:output` to `Canvas:clear`.

H. Common box connections

In this section we provide a number of examples showing typical connections between boxes. These examples aim to ease the understanding of examples presented in Section VIII.

Figure 16 (a) shows an example of connecting the `output` port of the `Param` box to an input port of another box. This can always be done when the input port format is known and is $\{N\}$, i.e., one-dimensional array or scalar. The `Param` box extracts port format from its connection and exhibits corresponding values as its own parameters. User enters parameter values, and at simulation startup they are sent to receiver(s) in just one data frame. These parameters remain constant during the simulation. Using `Param` to specify constant parameters is very common in simulations.

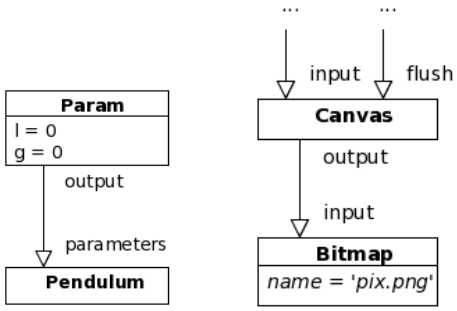


Figure 16.
(a) Using `Param` box to specify parameters.
(b) Attaching `Canvas` to `Bitmap`.

Figure 16 (b) shows the connection of `Canvas` box to `Bitmap` box. It is typical that the data for visualization is first accumulated in the canvas, and is written to image rarely (either when a data frame comes to `Canvas:flush` or automatically with user-specified time interval).

Figure 17 (a) explains how to make it possible to interactively modify a parameter during simulation. To do so, it is necessary to cut an existing link and place a `RangeInput` box (or other vector input box — see Section VII-G) in between, so that instead connection `a->b` we have two connections, `a->RangeInput:input` and `RangeInput:output->b`. Parameters of the input box determine which elements of data frames must be user-editable during simulation. In this example, the `RangeInput` box causes the slider bar shown in Figure 17 (b) to appear in running simulation, and the value of parameter a can be changed from 0 to 10 with step 0.01. Notice also that it is necessary to perform explicit activation through port `RangeInput:activator` frequently enough, because otherwise the box will not have any chance to process user input before the data processing finishes (see Section IV).

Figure 17 (c) shows the typical connection between the `Rk4` solver box (see Section VII-E4) and the `CxxOde` box (see Section VII-D10) providing the formulation of a system of ordinary differential equations. The `Rk4` box writes ODE

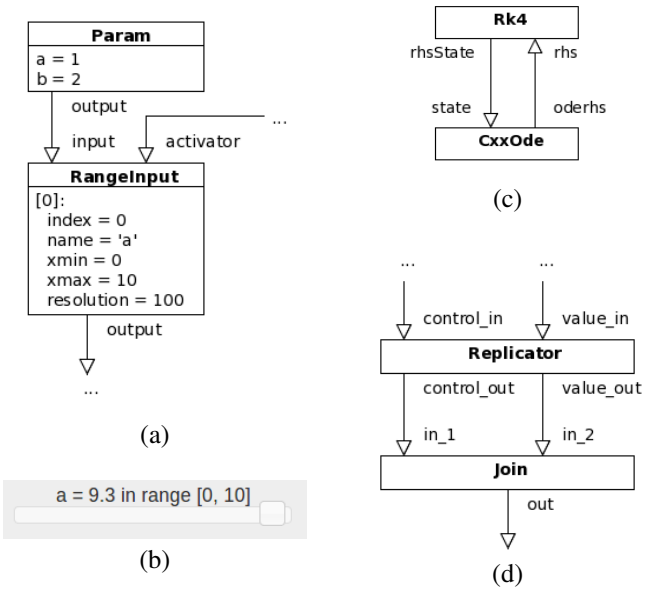


Figure 17.
(a) Using `RangeInput` to interactively modify parameters.
(b) Slider element for interactive input in running simulation.
(c) Coupling the `Rk4` solver and an ODE system.
(d) Using `Replicator` to feed `Join`.

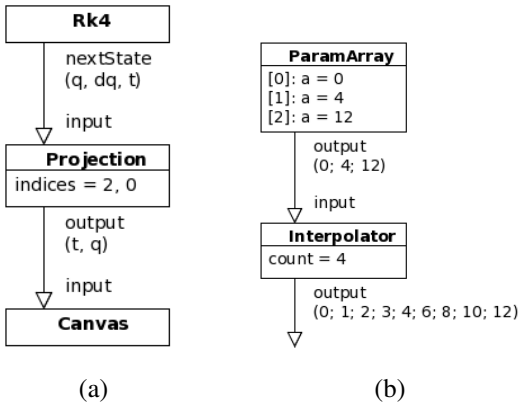


Figure 18.
(a) Obtaining 2D projection of points for plotting to `Canvas`.
(b) Example of usage for `Interpolator` box.

system state to port `Rk4:rhsState` and reads ODE right hand side from port `Rk4:rhs`. The `CxxOde` box computes the right hand side, as soon as it receives state at port `CxxOde:state`, and writes it to port `CxxOde:oderhs`.

Figure 17 (d) illustrates one of many possible applications of the `Replicator` box (see Section VII-F3). It is used here to make sure that the `Join` box (see Section VII-F1) receives equal number of data frames in ports `in_1` and `in_2`. When a data frame comes to `Replicator:value_in`, nothing happens. When a data frame comes to `Replicator:control_in`, it is written to `Replicator:control_out` and hence to `Join:in_1`. After that, the last data frame received at `Replicator:value_in` is written to `Replicator:value_out` and hence to `Join:in_2`. As a result, overflow never happens in the `Join` box.

Figure 18 (a) shows how a point can be projected onto 2D canvas. This is done using the `Projection` box. In this example, the box generates output data frames with elements t, q from input frames with elements q, $\dot{q}$, t: first element is t because it is the element with index 2 in the input frame; second element is q because it has index 0 in the input frame. Indices 2 and 0 are parameters of the `Projection` box.

Figure 18 (b) shows a combination of `ParamArray` (see Section VII-A) and `Interpolator` (see Section VII-E1) boxes. The interpolator in this example splits each span between two consecutive input data frames into 4 pieces and writes interpolated data to port `output`. Annotations near `output` ports of the boxes contain scalar data that is generated in this example.

Figure 19 shows a typical way to organize panning and zooming of image generated in a simulation. The image corresponds to a `Bitmap` box and is identified by image file

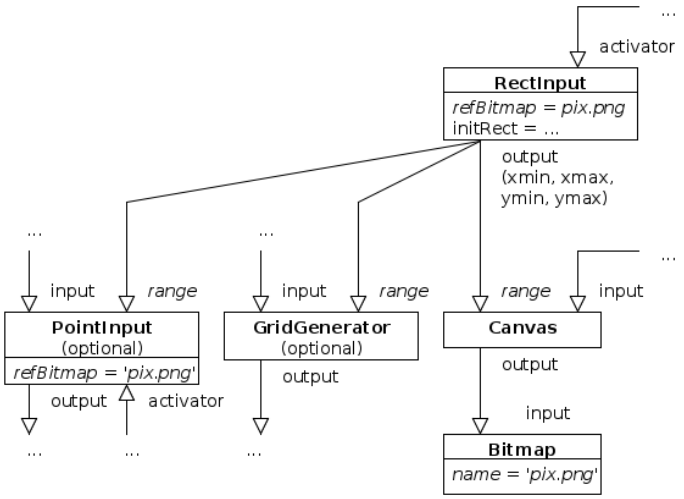


Figure 19. Implementation of panning, zooming, and point input.

name. Interactive user input for panning and zooming actions is provided by the `RectInput` box (see Section VII-G5). The box needs to be associated with image by specifying image file name as box parameter. The image is considered to cover the rectangle $x_{\min} \le x \le x_{\max}$, $y_{\min} \le y \le y_{\max}$ in plane x, y. Initial rectangle is specified by `RectInput` box parameters. Whenever user scrolls mouse wheel or drags across image, the box modifies rectangle parameters $x_{\min}$, $x_{\max}$, $y_{\min}$, $y_{\max}$, and writes them to port `output`. This port is connected to the `range` port of the `Canvas` box (see Section VII-C) supplying image data. It can also be connected to the `range` port of some other boxes, in accordance with simulation logics. For example, if each pixel on an image corresponds to a point of a grid, it is connected to port `GridGenerator:range` of box that generates the grid. If there is a `PointInput` box for the same image, its `range` port should also be connected to `RectInput:output`. Notice that all input boxes must be activated frequently enough through port `activator` in order to be able to process user input when simulation is running. This is currently the responsibility of simulation designer.

VIII. Examples of simulations

This section lists several examples of simulations. We will often use the notation `box:port`, where `box` is a name (not a type) of a box, and `port` is the name of its port.

A. Single phase trajectory of a simple pendulum

Figure 20 shows one of the simplest simulations — it plots a single phase trajectory for a simple pendulum. The ODE system is provided by the `ode` box (type `Pendulum`, see Section VII-D10). The box computes the right hand side $[\dot\varphi, \ddot\varphi]$ according to the pendulum equation

$$l\ddot\varphi + g\sin\varphi = 0,$$

where l is the pendulum length and g is the acceleration of gravity. The ODE right hand side depends on the state variables $[\varphi, \dot\varphi]$ and the vector of parameters $[l, g]$. They are supplied through input ports. Parameters are specified in the `odeParam` box. State variables come from the `solver` box (type `Rk4`, see Section VII-E4). The solver performs numerical integration of the initial value problem, starting from the user-specified initial state (the `initState` box). The solver is configured to perform a fixed number of time steps (the corresponding parameters come to the solver from the `solverParam` port). Each time the solver obtains a new system state vector, it sends the vector to its `nextState` port. Once the solver finishes, it activates the `finish` port to let others know about it. In this simulation, consecutive system states are projected to the phase plane (the `proj` box of type `Projection`, see Section VII-D7) and then rasterized by the `canvas` box (type `Canvas`, see Section VII-C). Finally, the data comes to the `bitmap` box (type `Bitmap`, see Section VII-B) that generates the output image file. Notice that this simulation has three data sources, `odeParam`, `solverParam`, and `initState`, of type `Param` — see Section VII-A.

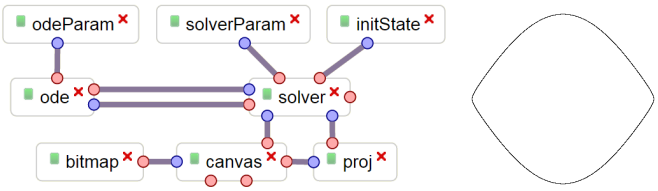


Figure 20. Single phase trajectory

From this simplest example one can see how to construct simulation scheme from boxes and links that computes what user needs. Other examples are more complex, but they basically contain boxes of the same types, plus probably some more.

B. Interactive phase portrait

An important aspect of a simulation is its ability to *interact with the user*. This can be achieved using input boxes (see Section VII-G). Figure 21 shows an example of interactive simulation: it generates phase trajectories passing through points clicked by the user on the phase plane. The box `isInput` has type `PointInput` and is responsible for that kind of input. Another available kind of user input is panning and zooming of the phase plane; it is handled by the `pan-zoom` box of type `RectInput`. Notice that there is no need to activate

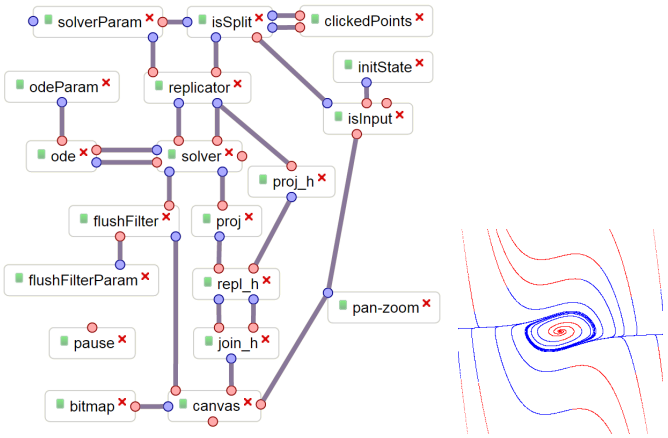


Figure 21. Interactive phase portrait

input boxes during data processing. It finishes very fast, and the input processing occurs after all data processing finishes, due to the presence of box `pause` of type `Pause`.

Each generated phase curve has two parts: blue in the time-positive direction (with resp. to the clicked point) and red in the time-negative direction. Many of the remaining boxes serve to achieve this behavior. The `solverParam` box has type `ParamArray`. It specifies two sets of solver parameters, one with positive value of time integration step h, and the other one with negative time step $-h$. Each data frame coming from `solverParam` causes an initial value problem to be solved, with a specific value of the time step. The data flow initiating the initial value problem solution is as follows. First, initial state travels the route `initState -> isInput -> isSplit -> replicator:value_in`. Then `replicator` returns control to `isSplit`, which then activates `solverParam`. That causes two sets of solver parameters to be generated in two data frames. When each of them reaches `replicator:control_in`, the `replicator` box first writes solver parameters to `solver:parameters`. At this point, the `solver` box (type `Rk4`) already knows which parameters to use, but it doesn't start the integration (it only does so when it receives an initial state). Therefore, the control is returned back to `replicator`. It then writes the initial state to `solver:initState`, which finally starts numerical integration.

When the solver produces a point of phase curve, $[x, \dot x, t]$, it is transformed into $[x, \dot x, \pm h]$ by boxes `proj` ($[x, \dot x, t] \to [x, \dot x]$), `proj_h` (solver parameters $\to \pm h$) of type `Projection`, `repl_h` (type `Replicator`), and `join_h` (type `Join`). Last two boxes glue data frames $[x, \dot x]$ and $\pm h$ together. Data frames $[x, \dot x, \pm h]$ come to the `canvas` box, so that points of time-positive part of phase curve are assigned value h, and time-negative parts are assigned value $-h$. The `bitmap` box has a color map that maps 0 to white, h to blue, and $-h$ to red, which finally gives us the desired look of the output image.

The remaining two boxes (`flushFilter` of type `CountedFilter` and `flushFilterParam` of type `Param`) are here in order to pass each second data frame from `solver:finish` to `canvas:flush`. As a result, the

image user sees in the browser is updated only when both branches of phase curve are computed.

Importantly, there is no need to modify this scheme to replace the ODE system: it is sufficient to provide the system formulation as a parameter of box `ode` (type `CxxOde`) and specify fixed system parameters in box `odeParam` (type `Param`).

C. Poincare map for double pendulum

The classical double pendulum system is a model of two pendulums moving in plane, with motionless support of the first pendulum and the second pendulum attached at the end of the first one, as shown in Figure 22. The parameters of the system are two masses, two lengths, and the acceleration of gravity. The configuration is determined by two angles, φ (rotation of the upper part) and ϑ (rotation of the lower part).

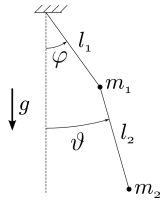


Figure 22. Double pendulum system

The equations of motion in the Lagrange form are as follows.

$$(m_1+m_2)l_1^2\ddot{\varphi} + m_2 l_1 l_2\left[\cos(\vartheta-\varphi)\ddot{\vartheta} - \sin(\vartheta-\varphi)\dot{\vartheta}^2\right] + g(m_1+m_2)\sin\varphi = 0,$$
$$m_2 l_2^2\ddot{\vartheta} + m_2 l_1 l_2\left[\cos(\vartheta-\varphi)\ddot{\varphi} - \sin(\vartheta-\varphi)\dot{\varphi}^2\right] + gm_2\sin\vartheta = 0.$$

This ODE system is non-integrable, and its phase trajectories can be quasi-periodic or chaotic, depending on the initial state. An easy way to reveal the type of behavior of a given trajectory is to look at its Poincaré map. This is done in the following simulation, shown in Figure 23.

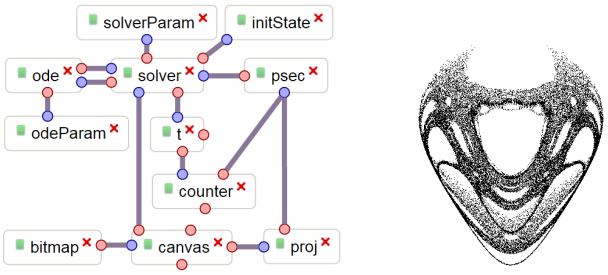


Figure 23. Double pendulum, Poincaré map (50000 points, 28.5 s)

Essentially, the scheme is very close to the one shown in Figure 20. But rather than to pass each next point of the phase curve from `solver:nextState` directly to `proj` and then to the `canvas`, we check for intersection with a hyperplane first. The `psec` box has type `CrossSection` (see Section VII-D3), hence `proj` receives points on the hyperplane; the rest of processing is same as for the simplest example in Section VIII-A.

Two boxes, `counter` of type `Counter` and `t` of type `ThresholdDetector`, are introduced in order to stop the integration as soon as 50000 points of the Poincaré map are obtained.

Importantly, there is no need to store phase trajectory or individual points of intersection of the trajectory with the plane during simulation. The entire processing cycle (test for intersection; projection; rasterization) is done as soon as a new point of the trajectory is obtained. After that, we need to store just one last point from the trajectory. Simulations like this are what we could not do easily in MATLAB or SciLab, and they have inspired us to develop NumEquaRes.

D. Ince–Strutt diagram

Figure 24 shows a simple simulation that allows one to obtain a stability diagram for a linear ODE system with periodic coefficients on the plane of parameters. Here the picture on the right is the Ince–Strutt diagram for the Mathieu equation [17]:

$$\ddot{q} + [\lambda - 2\gamma\cos(2q)]\,q = 0,$$

where λ and γ are parameters.

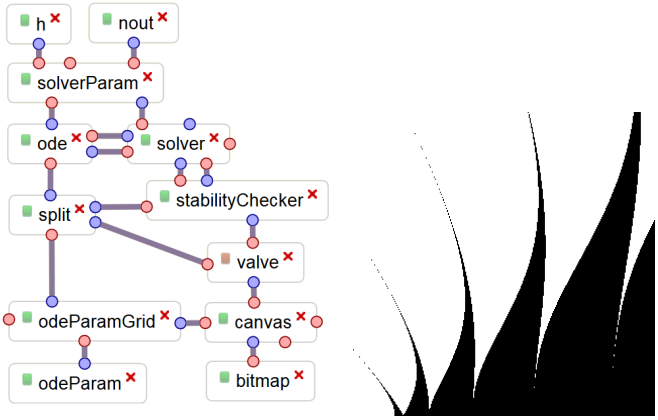


Figure 24. Ince-Strutt stability diagram (500 × 500 points, 6.3 s)

People who have experience with it know how difficult it is to build such kind of diagrams analytically, even to find the boundaries of stability region near the horizontal axis. What we suggest here is the brute force approach — it is fast enough, general enough, and it is done easily. The idea is to split the rectangle of parameters $\lambda_1 \le \lambda \le \lambda_2$, $\gamma_1 \le \gamma \le \gamma_2$ into pixels and analyze the stability in the bottom-left corner of each pixel (by computing eigenvalues of the monodromy matrix [5]), then assign pixel color to black or white depending on the result. In this simulation, important new boxes are `odeParamGrid` (type `GridGenerator`, see Section VII-E2) and `stabilityChecker` (type `LinOdeStabChecker`, see Section VII-E5). The former one provides a way to generate points $[\lambda, \gamma]$ on a multi-dimensional grid, and the latter one analyzes the stability of a linear ODE system with periodic coefficients.

The simulation works as follows. When a new point $[\lambda, \gamma]$ is generated by `odeParamGrid`, it is sent to the `split` box;

then split sends it to ode:parameters, valve:input, and finally to stabilityChecker:activator. The stabilityChecker box analyzes the stability of ODE system with given values of λ, γ, and sends the result to valve:valve. At this point, the valve box has received data frames at both of its input ports, so it decides whether to pass data frame from its port input to port output. The decision is determined by the result of stability analysis, since it comes to port valve. If the ODE system is stable, the data frame is passed, otherwise it is blocked. Therefore, the canvas box receives points $[\lambda, \gamma]$ for which the ODE system is stable, and the corresponding pixel in the bitmap box is drawn in black color. The canvas flushes its data to bitmap at the end of simulation due to the link odeParamGrid:flush -> canvas:flush, and also each second when simulation is running.

The box solverParam has type Rk4ParamAdjust not described in this paper; it is used here to compute the necessary number of numerical integration steps when period and time integration steps are known.

E. Strange attractor in forced Duffing equation

Figure 25 shows another application of Poincaré map, now in the visualization of the strange attractor arising in the forced Duffing equation [18]:

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = \gamma\cos(\omega t)$$

with parameters α, β, γ, δ, ω. User can change parameters interactively and see how the picture changes. This simulation is simpler than the one shown in Figure 23, because to obtain a new point on canvas, one just needs to apply time integration over known time period $T = 2\pi/\omega$ of system excitation. The solver is configured such that data frames $[x, \dot{x}, t]$ generated at port solver:nextState are in plane $t = kT$, with an integer k. Then the projection box throws t away, and canvas receives points $[x, \dot{x}]$.

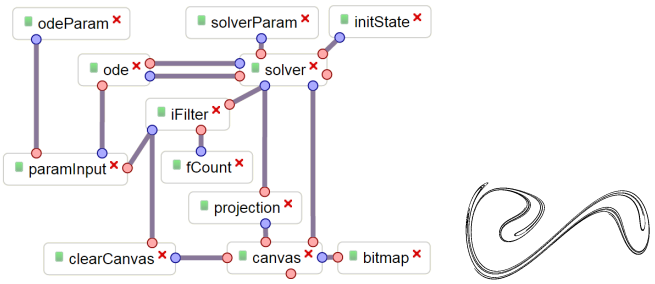


Figure 25. Strange attractor for forced Duffing equation (interactive simulation)

Boxes paramInput and clearCanvas have types RangeInput and SignalInput, respectively. The box paramInput allows user to modify parameters α, β, γ, δ by moving sliders. The box clearCanvas allows user to clear canvas by clicking a button. Notice that the user input is processed together with the data processing. Therefore, both input boxes have to be activated from time to time. This is done through box iFilter of type CountedFilter: as soon as a new point is generated at port solver:nextState, it comes to iFilter:input, and each 10-th point reaches iFilter:output and paramInput:activate, clearCanvas:activate connected to it. The counted filter box is used in order to activate the input boxes not too often, otherwise simulation performance could suffer due to the input processing.

F. The Mandelbrot set

Figure 26 shows an interactive simulation of the Mandelbrot set [19], which is defined as the set of complex numbers c for which the sequence $z_0, z_1, z_2, \ldots : z_0 = 0, z_{k+1} = z_k^2 + c$ is bounded.

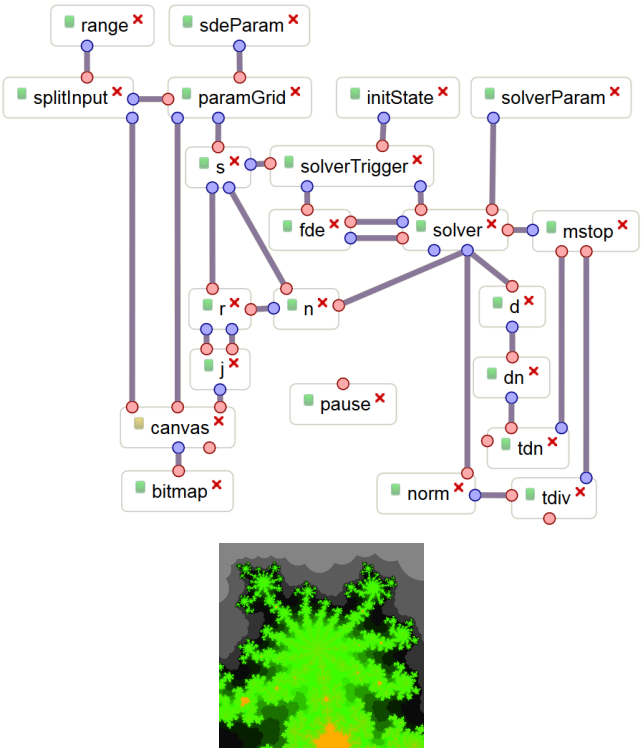


Figure 26. Colored Mandelbrot set (interactive simulation)

Since the Mandelbrot set is a fractal, it is important for the user to be able to pan and zoom the picture using the mouse. This is achieved by using the range box of type RectInput (see Section VII-G5) that feeds the ranges for real and imaginary parts of c to canvas:range and paramGrid:range through the splitInput box of type Split.

The paramGrid box (type GridGenerator) generates c on a grid covering the specified rectangle on complex plane. For each c from that grid, a part of the sequence z_k is evaluated, and its boundness is checked. Iterations stop as soon as sequence convergence or divergence is detected. The number of iterations, n, done for each c is stored in box n of type Counter; it determines the color of pixel corresponding to c in the final image.

The simulation works as follows. The initial state $z_0 = 0$ comes from initState to solverTrigger:value_in (the box solverTrigger is of type Replicator). Then paramGrid comes into play, activated by dummy value $c = 0$ from sdeParam. Each value of c

generated by paramGrid comes through box s (type Splitter) to n:reset (to reset iteration counter n), and then to solverTrigger:control_in, which causes the latter to write c to fde:parameters and z_0 to solver:initState.

The boxes fde and solver are of types CxxFde and FdeIterator, respectively. When z_0 comes to solver:initState, the solver box starts generating elements of sequence z_k. It generates at most 500 elements, but is stopped if sequence convergence or divergence is detected. The divergence analysis is performed by boxes boxes norm (type Scalarizer, computes $|z_k|$) and tdiv (type ThresholdDetector, evaluates $|z_k| > 3$). The convergence analysis is performed by boxes d (type Differentiate, computes $z_k - z_{k-1}$), dn (type Scalarizer, computes $|z_k - z_{k-1}|$), and tdn (type ThresholdDetector, evaluates $|z_k - z_{k-1}| < 10^{-5}$). When the expression in either tdiv or tdn evaluates to true, the solver is stopped (the box mstop is of type Merge, its port output is connected to solver:stop).

When the iterations of z_k stop, the control returns to the box s, and it writes c to r:control_in by activating its third output port. At this point, r (box of type Replicator) forwards c to its port r:control_out and the number of iterations done, n (obtained earlier from box n) to port r:value_out. Then the box j of type Join glues the coordinates of c together with n, and the data frame $[\mathrm{Re}\, c, \mathrm{Im}\, c, n]$ comes to canvas.

Importantly, we did not have to develop any new box types in order to describe the logics of convergence analysis for sequences of complex numbers generated by the system, but used standard general-purpose boxes instead.

IX. Comparison with other tools

Direct comparison between NumEquaRes and other existing tools is problematic because all of them (at least, those that we have found) do not provide an easy way for user to describe the data processing algorithm. In some systems, the algorithm can be available as a predefined analysis type; in others, user would have to code the algorithm; also, there are systems that need to be complemented with external analysis algorithms.

Let us consider example simulations shown in Figures 23, 24, 25, and try to solve them using different free tools; for commercial software, try to find out how to do it from the documentation. Further in this section, figure number refers to the example problem.

TABLE I. Comparison of NumEquaRes with other tools

Name	*Free*	*Web*	*Can solve*	*Fast*
Mathematica	no	yes	23, 24, 25; needs coding	n/a
Maple	no	no	23, 24, 25; needs coding	n/a
MATLAB	no	no	23, 24, 25; needs even more coding	no
SciLab	yes	no		no
OpenModelica	yes	no	none	could be
XPP	yes	no	23, 25	yes
InsightMaker	yes	yes	none	n/a

In Table I, commercial proprietary software is limited to most popular tools — Mathematica, Maple, and MATLAB. In many cases, purchasing a tool might be not what a user (e.g., a student) is likely to do.

All of the three example simulations are solvable with commercial tools Mathematica, Maple, and MATLAB.

In Mathematica, it is possible to solve problems like 23, 25 using standard time-stepping algorithms since version 9 (released 24 years later than version 1) due to the WhenEvent functionality. Problem 24 can also be solved. All algorithms have to be coded. Notice that Wolfram Alpha [20] (freely available Web interface to Mathematica) cannot be used for these problems.

Maple has the DEtools[Poincare] subpackage that makes it possible to solve problem 23 and others with Hamiltonian equations; problems 24, 25 can be solved by coding their algorithms.

With MATLAB or SciLab, one can code algorithms for problems 24, 25 using standard time-stepping algorithms. For problem 23, one needs either to implement time-stepping algorithm separately or to obtain Poincaré map points by finding intersections of long parts of phase trajectory with the hyperplane. Both approaches are more difficult than those in Mathematica and Maple. And, even if implemented, simulations are much slower than with NumEquaRes.

OpenModelica [21] is a tool that helps user formulate the equations for a system to be simulated; however, it is currently limited to only one type of analysis — the solution of initial value problem. Therefore, to solve problems like 23, 24, 25, one has to code their algorithms (e.g., in C or C++, because the code for evaluating equations can be exported as C code).

XPP [6] provides all functionality necessary to solve problems 23, 25. It contains many algorithms for solving equations (while NumEquaRes does not) and is a powerful research tool. Yet it does not allow user to define a simulation algorithm, and we have no idea how to use it for solving problem 24.

Among other simulation tools we would like to mention InsightMaker [22]. It is a free Web application for simulations. It has many common points with NumEquaRes, although its set of algorithms is fixed and limited. Therefore, problems 23, 24, 25 cannot be solved with InsightMaker.

X. Technical challenges

The design of NumEquaRes governs technical challenges specific to Web applications for simulations. They are related to performance and security, and are discussed in this section.

A. Server CPU resources

Currently, all simulations run on the server side. Some of them can be computationally intensive and consume considerable amount of CPU time. For example, there are simulations that consume 100% of single CPU core time for as long as user wishes. This is a problem if the number of users grows. Of course, we do not expect millions of users simultaneously running their simulations, but still there is a scalability problem.

The problem can be addressed in a number of ways. Firstly, the server can be an SMP computer, so it will be able to run as many simulations as the number of CPU cores, without any loss of performance. Secondly, it is technically possible to have a cluster of such computers and map its nodes to user sessions. Obviously, this approach requires the growth of server hardware to provide sufficient server performance.

A different approach is to move running simulations to the client side. In this case, the server loading problem will disappear. But how is it possible to offer user's browser to run something? Actually, today the only choice seems to be JavaScript. We will have to compile simulations into it, or to the asm.js subset of JavaScript. This approach is quite possible for some simulations, but is problematic for other ones that can make use of some large libraries like LAPACK.

B. User code security

NumEquaRes web server accepts C++ code as part of simulation description provided by user. This is the direct consequence of our wish to provide good computational performance of simulations. Such pieces of code typically describe how to compute the right hand side of an ODE system, or how to compute another transformation of input data frames into the output data frames. The server compiles that code into dynamic library to be loaded and executed by core application that performs the simulation. Potentially, we have serious risk of direct execution of malicious code.

Currently, this problem is solved as follows. Once user code is compiled into a library (shared object on UNIX or dynamically linked library on Windows), it is checked for the absence of any external or weak symbols that are not found in a predefined white list (the list contains symbol names for mathematical functions and a few more). Due to this, user code is not able to make any system calls. For example, it cannot open file `/etc/passwd` and send it to the user because it cannot open files at all. If the security check on the compiled library fails, no attempt to load it is done, and the user gets notified about the reason of check failure.

On the other hand, malicious code could potentially exploit such things as buffer overrun and inline assembly. It is an open problem now how to ensure nothing harmful will happen to the server due to that. However, the ban on any non-white-listed calls seems to be strong enough. Probably, one more level of protection could be achieved with a utility like chroot.

A better approach to provide security is to disallow any C++ code provided by user. But this would imply giving the user a good alternative to C++ allowing to describe his/her algorithms equally efficiently. For example, there could be a compiler of formulas into C++ code. Nothing like this is implemented at the moment, but can be done in the future. In this case, the user code security problem will vanish.

XI. Conclusion and future work

A new tool for numerical simulations, NumEquaRes, has been developed and implemented as a Web application. The core of the system is implemented in C++ in order to deliver good computational performance. It is free software and thus everyone can contribute into its development. The tool already provides functionality suitable for solving many numerical problems, including the visualization of Poincaré maps, stability diagrams, fractals, and more. Simulations run on server; besides, they may contain C++ code provided by user. This creates two challenges — potential problems of server performance and security. The security problem has been addressed in our work; the performance problem is not currently taken into account.

The algorithm of simulation runner implies that the order of activation calls it makes is not important, i.e., does not affect simulation results. While this is true for typical simulations, counter-examples can be invented. Further work is to make it possible to distinguish such simulations from regular ones and render them invalid. Another option is to eliminate internal uncertainty in simulation specification: only allow one connection per output port and require the initial order of source box activation to be explicitly specified by the user.

NumEquaRes is a new project, and the current state of its source code corresponds more to the proof-of-concept stage than the production-ready stage, because human resources assigned to the project are very limited. To improve the source code, it is necessary to add developer documentation, add unit tests, and deeply refactor both client and server parts of the Web interface.

Further plans of NumEquaRes development include new features that would significantly extend its field of application. Currently, the most serious bottleneck for user is having to supply equations in the form of C++ code. This problem can be addressed by implementing interoperability between NumEquaRes and other tools. For example, many simulation tools are able to formulate problem equation using the Functional Mock-up Interface (FMI) standard format [23]. It is well possible to develop a new box type with interface similar to `CxxOde` but taking its input from an FMI model exported from another tool. It is important to notice, however, that more advanced numerical time-stepping solvers (e.g., CVODE from the Sundials library [24]) have to be used to simulate these models.

Another set of planned features aims to enhance the level of presentation of simulation results (currently, it is quite modest). Among them is 3D visualization and animation.

Last but not least, an important usability improvement can be achieved with a feature that visualizes simulation data flows; its role is similar to debugger's.

References

[1] S. Orlov and N. Shabrov, "Numequares — web application for numerical analysis of equations," in SOFTENG 2015: The First International Conference on Advances and Trends in Software Engineering, S. F. Felipe and H. Jameleddine, Eds. IARIA XPS Press, 2015, pp. 41–47.

[2] "Numequares — an online system for numerical analysis of equations," URL: http://equares.ctmech.ru/ [accessed: 2015-11-25].

[3] E. J. Routh, The Advanced Part of a Treatise on the Dynamics of a System of Rigid Bodies, 6th ed. Macmillan, London, 1905, reprinted by Dover Publications, New York, 1955.

[4] L. Meirovitch, Elements of vibration analysis. New York: McGraw-Hill, 1986.

[5] G. Teschl, Ordinary Differential Equations and Dynamical Systems, ser. Graduate studies in mathematics. American Mathematical Soc., URL: http://books.google.ru/books?id=FSObYfuWceMC [accessed: 2015-11-25].

[6] B. Ermentrout, Simulating, Analyzing, and Animating Dynamical Systems: A Guide to XPPAUT for Researchers and Students, ser. Software, Environments and Tools. Society for Industrial and Applied Mathematics, 2002, URL: http://books.google.ru/books?id=Qg8ubxrA060C [accessed: 2015-11-25].

[7] "Using server-sent events," URL: https://developer.mozilla.org/en-US/docs/Server-sent_events [accessed: 2015-11-25].

[8] "Lamp (software bundle)," URL: http://en.wikipedia.org/wiki/LAMP_(software_bundle) [accessed: 2015-11-25].

[9] "D3.js — data-driven documents," URL: http://d3js.org/ [accessed: 2015-11-25].

[10] "A full-featured markdown parser and compiler, written in javascript," URL: https://github.com/chjj/marked [accessed: 2015-11-25].

[11] "Mathjax — beautiful math in all browsers," URL: http://www.mathjax.org/ [accessed: 2015-11-25].

[12] VTK user's guide. Kitware, Inc., 2010, 11th ed.

[13] E. Brown, Web Development with Node and Express. Sebastopol, CA: O'Reilly Media, 2014.

[14] "Ajax (programming)," URL: https://en.wikipedia.org/wiki/Ajax_(programming) [accessed: 2015-11-25].

[15] "Acml — amd core math library," URL: http://developer.amd.com/tools-and-sdks/archive/amd-core-math-library-acml/ [accessed: 2015-11-25].

[16] J. C. Butcher, Numerical Methods for Ordinary Differential Equations. New York: John Wiley & Sons, 2008.

[17] M. Abramowitz and I. Stegun, Mathieu Functions, 10th ed. Dover Publications, 1972, chapter 20, pp. 721–750, in Abramowitz, M. and Stegun, I., Handbook of Mathematical Functions, URL: http://www.nr.com/aands [accessed: 2015-11-25].

[18] C. M. Bender and S. A. Orszag, Advanced Mathematical Methods for Scientists and Engineers I: Asymptotic Methods and Perturbation Theory. Springer, 1999, pp. 545–551.

[19] J. W. Milnor, Dynamics in One Complex Variable, 3rd ed., ser. Annals of Mathematics Studies. Princeton University Press, 2006, vol. 160.

[20] "Wolframalpha — computational knowledge engine," URL: http://www.wolframalpha.com/ [accessed: 2015-11-25].

[21] P. Fritzson, Principles of Object-Oriented Modeling and Simulation with Modelica 2.1. Wiley-IEEE Computer Society Pr, 2003.

[22] S. Fortmann-Roe, "Insight maker: A general-purpose tool for web-based modeling & simulation," Simulation Modelling Practice and Theory, vol. 47, no. 0, 2014, pp. 28 – 45, URL: http://www.sciencedirect.com/science/article/pii/S1569190X14000513 [accessed: 2015-11-25].

[23] "Functional mock-up interface," URL: https://www.fmi-standard.org/ [accessed: 2015-11-25].

[24] "Sundials — suite of nonlinear and differential/algebraic equation solvers," URL: https://computation.llnl.gov/casc/sundials/main.html[accessed: 2015-11-25].

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601