# International Journal on Advances in Intelligent Systems


2009 vol. 2 nr. 4

The *International Journal on Advances in Intelligent Systems* is Published by IARIA.
ISSN: 1942-2679
journals site: http://www.iariajournals.org
contact: petre@iaria.org



Sponsored by IARIA
www.iaria.org

**International Journal on Advances in Intelligent Systems**
**Volume 2, Number 4, 2009**

- Thomas C. Schmidt, University of Applied Sciences – Hamburg, Germany
- Karolj Skala, Rudjer Bokovic Institute - Zagreb, Croatia
- Chieh-yih Wan, Intel Corporation, USA
- Hoo Chong Wei, Motorola Inc, Malaysia

**Ubiquitous Systems and Technologies**

- Matthias Bohmer, Munster University of Applied Sciences, Germany
- Dominic Greenwood, Whitestein Technologies AG, Switzerland
- Arthur Herzog, Technische Universitat Darmstadt, Germany
- Reinhard Klemm, Avaya Labs Research-Basking Ridge, USA
- Vladimir Stantchev, Berlin Institute of Technology, Germany
- Said Tazi, LAAS-CNRS, Universite Toulouse 1, France

**Advanced Computing**

- Dumitru Dan Burdescu, University of Craiova, Romania
- Simon G. Fabri, University of Malta – Msida, Malta
- Matthieu Geist, Supelec / ArcelorMittal, France
- Jameleddine Hassine, Cisco Systems, Inc., Canada
- Sascha Opletal, Universitat Stuttgart, Germany
- Flavio Oquendo, European University of Brittany - UBS/VALORIA, France
- Meikel Poess, Oracle, USA
- Said Tazi, LAAS-CNRS, Universite de Toulouse / Universite Toulouse1, France
- Antonios Tsourdos, Cranfield University/Defence Academy of the United Kingdom, UK

**Centric Systems and Technologies**

- Razvan Andonie, Central Washington University - Ellensburg, USA / Transylvania University of Brasov, Romania
- Kong Cheng, Telcordia Research, USA
- Vitaly Klyuev, University of Aizu, Japan
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Willy Picard, The Poznan University of Economics, Poland
- Roman Y. Shtykh, Waseda University, Japan
- Weilian Su, Naval Postgraduate School - Monterey, USA

**GeoInformation and Web Services**

- Christophe Claramunt, Naval Academy Research Institute, France
- Wu Chou, Avaya Labs Fellow, AVAYA, USA
- Suzana Dragicevic, Simon Fraser University, Canada
- Dumitru Roman, Semantic Technology Institute Innsbruck, Austria
- Emmanuel Stefanakis, Harokopio University, Greece

**Semantic Processing**

- Marsal Gavalda, Nexidia Inc.-Atlanta, USA & CUIMPB-Barcelona, Spain
- Christian F. Hempelmann, RiverGlass Inc. - Champaign & Purdue University - West Lafayette, USA
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Massimo Paolucci, DOCOMO Communications Laboratories Europe GmbH – Munich, Germany
- Tassilo Pellegrini, Semantic Web Company, Austria
- Antonio Maria Rinaldi, Universita di Napoli Federico II - Napoli Italy
- Dumitru Roman, University of Innsbruck, Austria
- Umberto Straccia, ISTI – CNR, Italy
- Rene Witte, Concordia University, Canada
- Peter Yeh, Accenture Technology Labs, USA
- Filip Zavoral, Charles University in Prague, Czech Republic

**International Journal on Advances in Intelligent Systems**
**Volume 2, Number 4, 2009**

**CONTENTS**

# Estimation of Mobile User's Trajectory in Mobile Wireless Network

Framework, Formulation, Design, Simulation and Analyses

Sarfraz Khokhar
Mobile Internet Group
Cisco Systems, Inc.
Research Triangle Park, NC USA
khokhar@cisco.com

Arne A. Nilsson
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC, USA
nilsson@ncsu.edu

***Abstract*—There has been extensive research and development on Location Based Services (LBS). The focus here is on the present location of the mobile user. A new trend of mobile wireless services based on trajectory of a mobile phone and hence that of a mobile user, called Mobile Trajectory Based Services (MTBS) is emerging. We propose trajectory estimation method suitable for MTBS which uses an adaptive polling scheme such that the polling is not frequent to overburden the air resources unnecessarily, and it is not sparse that leads to an ambiguous and erroneous mobility profile. The polling method includes a new map matching scheme using GIS (Geographical Information System) map. The scheme is based on previous location, map topology and travel time from one intersection to the other. So, from erroneous and moderately polled location points with time stamps, we form a spatiotemporal trajectory for a mobile user. A set of such trajectories for all days of the week would form a mobility profile, on which plethora of MTBS could be provisioned. We present a framework, formulation, and design for mobile user's trajectory. We ran trajectory simulation for an urban area of St. Paul, Minnesota and a suburban area of North Carolina. Our design worked very well in the urban area for location error of 100 meters or less. For the rural area, we ran simulation for location error up to 400 meters. The trajectory estimation error was very minimal.**

***Keywords-component; mobile location; mobile trajectory; mobile trjectory-based Services; GIS; map matching; mobility profile.***

## I. INTRODUCTION

The location specific services came to usage with the deployment of NAVSTAR GPS (**Nav**igation **S**ignal **T**iming **a**nd **R**anging **G**lobal **P**ositioning **S**ystem), commonly called, Global Positioning Systems (GPS), by US Department of Defense in the 70's. The services were exclusively for military purposes. In 1984 US government granted GPS civilian access, however with an intentional error, called selective access. In May 2000 selective access was removed and the civilian are enjoying relatively lower error positioning systems [2]. Within the last few years there is a massive surge of interest in yet another positioning methodology and applications that could use the positioning information using cellular networks and handsets. The reasons for this surge is mainly because new wireless positioning technologies have the capability of alleviating shortcomings of GPS, such as high power consumption and slow to acquire initial position and legislation in the United States requiring cellular phone operators to provide location information to 911 emergency call centers. A large proportion of the 911 calls come from cellular phones. The FCC mandate (docket 94-102) [3] requires mobile network operators to provide positional information to emergency services accurate within 125 meters. This regulatory requirement provided a stimulus for the commercial development of wireless positioning technologies and applications built on them, hence the birth of location based services (LBS). The location in this context is either an exact mobile location or covering a large area, cell or several cells referred to as location area. Mutual advances in location determination and geo-spatial technologies have provided a rich platform to develop mobile LBS [4]. Within the context of LBS, mobile wireless communication and geo-spatial database go hand in hand. The term LBS refers to mobile services in which the user's location is used in order to add value to the service as a whole. Such services are based on user's location in x-y coordinates (or location area) on a map, which is determined by different location determination technology, such as Time of Arrival (TOA), Time Difference Of Arrival (TDOA), Angle of Arrival (AOA), Time Advance (TA), Cell-ID, and Assisted GPS (A-GPS) [4][5]. A new trend of mobile wireless services based on the trajectory of a mobile user, called Mobile Trajectory Based Services (MTBS) is in its early research [6]. MTBS capitalize on the mobility profile of a mobile user. The mobility profile of a mobile user is a set of prerecorded trajectories of a mobile user that were captured by the mobile operator over a period of time, a week for example. In this paper we propose a method of estimating a mobile user's trajectory.

With GPS one has the luxury of numerous data points, since the location estimation takes place in the GPS receiver, so no extra air resources are used. Secondly GPS has incredible accuracy [7]. In case of mobile wireless, anomalies such as minor interference and reflection because of non line of sight or multipath result in erroneous time of travel estimation and hence a wrong distance. This leads to an erroneous location point determined by the underlying technology. For AOA, non line of sight causes wrong angle

measurement which results in erroneous location. Mobile wireless location estimation systems are prone to anomalies such as hearability of remote base stations, and geometric dilution of precision. Hence, the location coordinates estimated in mobile wireless system are erroneous; therefore we cannot trace the actual trajectory with simple interpolation methods. In mobile wireless the location error could be as large as 300 meters. Another important factor is the location polling frequency. A very frequent polling would potentially yield to a dense spatial distribution, obviously offset with error and could potentially yield a set of candidate trajectories closer in shape to the actual trajectory, however it is overburdening of the air channels and the location servers, which would affect the general operation of a wireless network. Higher polling frequency would also consume more energy on the mobile devices. On the other hand a very low frequency would lead to very ambiguous trajectory. Such trajectory would not resemble in shape with any of the paths taken from a GIS (Geographical Information System) map. If it does resemble to one, it would very likely be a wrong one. So, we need to poll enough for location estimation to yield us the trajectory that conserves underlying feature of the shape of the curve. Our assumption is that a mobile user travels on roads. Map matching techniques map the estimated raw location point to a point on the road network [8][9]. It can also map potentially the whole curve consisting of erroneous estimated location points to a curve consisting of a set of roads on GIS map which corresponds to the mobile user's true trajectory. It is called curve-to-curve matching [10]. Each road in GIS database is associated with direction and maximum speed limit [11]. Since there is an error, ε, associated with estimated location, the mobile user could be on any road segment contained in a circle of radius ε with center at the estimated location coordinates.

To analyze and formulate a mobile user's trajectory we define a framework of a metric space, we called it Mobile User space. Each estimated location corresponds to a closed ball; we refer to it as an error ball. A set of such embedded error balls forms a basis for the estimation of the desired trajectory. The trajectory so formed is used to map match curve-to-curve with the GIS database

The core of our map matching technique is based on adaptive location polling which is governed by the map topology and its constraints; we called it Intersection Polling Method (IPM). The idea behind IPM is to poll the mobile user's location, when the user is estimated to be at an intersection. Using the intersection points the mobile user's trajectory would be estimated with the least number of points. However, due to variations in the speed of a mobile user for different reasons and location error, we cannot determine exactly, when the mobile user is at an intersection every time. Therefore we may need to poll the mobile user's location more than one time from one intersection to another. Before the mobile user is polled for location, the previous erroneous location needs to be corrected and candidate road segments need to be identified. The process of determination of candidate road segments referred to as geodesic candidates must be fast.

The roads are represented in GIS as polylines. So, essentially GIS database for a road network is a set of coordinate points. The estimated location is a point with some error. In our study we assume a uniform circular error distribution. We introduce conditions for geodesics candidacy in terms of the estimated location and the road segment vertices, i.e., the road network GIS database. Using these conditions the geodesic candidacy is established with very little processing, since it is merely a comparison. [12] provides candidacy conditions, however they are not in terms of data points that could be processed to establish candidacy before another location polling may be required specially when there are frequent turns. After we have a set of all the geodesic candidates, the trajectory is estimated by connecting the segments. In some cases there may be multiple geodesic candidates for a road segment. The most likely candidate is picked by applying directional and path constraints. We introduce a measure of directional negativity of a geodesic with respect to the direction of travel. The direction of travel is determined from the two consecutive estimated locations. A candidate geodesic with an angle difference of greater than $\frac{\pi}{2}$ or less than $-\frac{\pi}{2}$ with the direction of travel, is defined to have a directional negativity and therefore is eliminated. Path constraints from the previous corrected location to the recent corrected location on the candidate geodesic further eliminates the unrealistic geodesics. In case there is at least one instance where we still have multiple geodesic candidates for the final trajectory, a curve-to-curve matching is performed on each candidate path with the path of the estimated location points. The candidate path that resembles the most with that of the estimated location points is selected as the path of the mobile travel for that particular segment of the mobile trajectory.

To run simulation on our proposed method we used GIS data [13] of metropolitan area of St. Paul, Minnesota and that of a suburban area of North Carolina. We developed a software tool, Digitizer, using C Sharp in Microsoft Windows to convert images of the paper maps of the areas into GIS database. The database schema of Digitizer is similar to what is widely used for road network [14]. The proposed mobile trajectory estimation method was implemented in Microsoft Windows as well. The proposed method performed very well with the exception of a scenario of very large location error in dense metropolitan area. In rural area it performed flawlessly even with larger errors.

This paper is an extended version of a conference paper referenced in [1]. It is organized as follows. Section II outlines how road networks have been represented in the literature and describes our road network model and its representation. Section III lays out mobile trajectory

estimation framework. It covers in details our metric space framework; we called it Mobile User space. Erroneous mobile location correction, location polling scheme and topological constraints are discussed in this section. Section IV covers the detailed steps to estimate the trajectory, including a flow chart for the algorithm. In Section V we describe the implantation of the proposed method, including details of Digitizer and our mobile trajectory estimation application, Mobility Profiler. Section VI is dedicated for simulation results. The paper concludes in Section VII.

## II. ROAD NETWORK REPRESENTATION

We are interested in the mobility profiling of a mobile user who traverses on roads. This leaves us with the road network as the natural container of a mobile user. A theoretical framework for the mobility profiling is closely tied to the road network's representation. A variety of road network representation schemes have been proposed, each discussed in a different framework. A brief description of some popular schemes is given below:

1) Steering direction summarizes the segmentation data by a steering direction, independent of the actual road geometry. This is the approach taken in the ALVINN (Autonomous Land Vehicle in a Neural Network), a neural net road follower [15]. In principle, ALVINN could learn appropriate steering commands for roads which change slope, bank, etc. In practice, images are backprojected onto a flat ground plane and re-projected from different points of view to expand the range of training images. This may prove to be a limiting factor on hilly roads.

2) Linear on a locally flat ground plane road model has three parameters, the road width and two parameters describing the orientation and offset of the vehicle with respect to the centerline of the road. LaneLok and Scarf algorithms take this approach [16]. The main limit of this type of scheme is the need to move a sufficiently small distance between road parameter estimates so that the straight path being driven along does not diverge too much from the actual road.

3) Modeling the road as a cross-section swept along a circular arc explicitly models road curvature but retains the flat earth assumption used in linear models. VaMoRs (Universität der Bunderwehr München's autonomous road vehicle) [17], and YARF (Yet Another Road Follower) [18] use this approach. The equations describing feature locations can be linearized to allow closed form least squares solutions for the road heading offset, and curvature, as well as the relative feature offsets.

4) A more general model of road geometry retains the flat earth assumption, but requires only that road edges to be locally parallel, allowing the road to bend arbitrarily. This can be done by projection onto the ground plane, VITS (Vision system for autonomous land vehicle navigation) system [19], or in the image plane. The lack of higher order constraint on the road shape can lead to serious errors in the recovered road shape when there are errors in the results of the underlying image segmentation techniques.

Several algorithms have been developed to recover three dimensional variations in road shape under the assumption that the road does not bank. These current algorithms use information from a left and right road edge, which precludes integrating information from multiple road markings. Such algorithms may be very sensitive to errors in feature location by the segmentation processes. This is due to the assumption of constant road width, which leads to errors in road edge location being interpreted as the result of changes in the terrain shape. Circular arc models would appear to be the technique of choices in the absence of algorithms for the recovery of three dimensional road structures which are robust in the presence of noise in the segmentation data. They have a small number of parameters, they impose reasonable constraints on the overall road shape, and statistical methods can be used for estimating the shape parameters, with all the statistical theory and tools that use of such methods allows the system to apply to the problem.

We are most interested in road network representation as described in GIS database. This road network representation scheme is similar to 2), mentioned above. Our road representation, also referred as road network element in our work, is described in the following subsection.

### A. Road Network Element

We represent a road as piece-wise linear curve also known as polyline. The curve is intersected by other road segments to realize intersections of different shapes: cross, Y, T etc. The other parameters associated with roads, we are interested in our work, are: direction and posted speed limits. Our road representation is depicted in Figure 1.

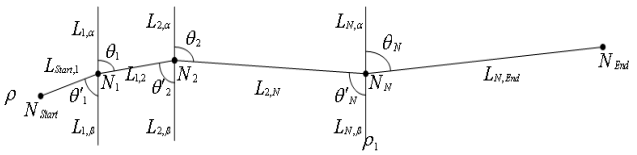



Figure 1. Road network element model.

The above model is a generic representation of roads of all shapes. A road, $\rho$, starts from point $N_{Start}$ and terminates at $N_{End}$. These points are referred to as start node and end node respectively. A road may be intersected by other roads, $\mathbf{L_{j,\alpha}}, \mathbf{L_{j,\beta}}$, j =1,2,3….N. at $N_i$, i =1,2,3,….N. In most of the cases $\mathbf{L_{j,\alpha}} \equiv \mathbf{L_{j,\beta}}$, i.e., both the segments are part of the same intersecting road network element for which $\rho$ is an intersecting road element. The intersecting roads intersect with $\rho$ at $N_i$, i=1,2,3,….N, making angles $\boldsymbol{\theta}_i, \boldsymbol{\theta}'_i$ with $\mathbf{L_{j,\alpha}}$ and $\mathbf{L_{j,\beta}}$ respectively. In most of the cases, for which

$\mathbf{L}_{\mathbf{j},\alpha} \equiv \mathbf{L}_{\mathbf{j},\beta}$, implies $\boldsymbol{\theta}_i \equiv \boldsymbol{\theta}'_i$. The number of intermediate nodes $N \geq 0$, i.e., intersections, is arbitrary.

### B. Road Network Database

As mentioned in the section above, our road network representation has the elements as described in Tables I and II. We store the roads in a database (GIS database) in the following format. Road ID is a distinct number assigned to different roads. The whole segment of the polyline representing that particular road has a starting point and end point. The order of the starting and end points codes the direction of traffic on the road. The two way roads are flagged in the database as bi-directional.

TABLE I. PARAMETERS OF A ROAD ELEMENT

| ROAD ID | START POINT | END POINT | INTERSECTIONS | POSTED SPEED |
|---|---|---|---|---|
| $N$ | $N_{Start}(x_0, y_0)$ | $N_{End}(x_p, y_p)$ | $(x_1, y_1)...(x_N, y_N)$ | S |

TABLE II. END TO END DEFINITION OF A ROAD ELEMENT

| ROAD ID | POINTS COORDINATES | IS ROAD ELEMENT DIRECTIONAL |
|---|---|---|
| $N$ | $(x_1, y_1),(x_2, y_2),(x_3, y_3),.....,(x_n, y_n)$ | False/True |

The intersections are represented as Cartesian coordinates of a road segment where it intersects or meets another road element. For mobility profiling technique we propose, these points are pivotal in determining rather calibrating the exact shape of the mobile user's trajectory. The posted speed which is a very important weight of the road network also plays a very important role in determining the mobile trajectory. This is the speed associated with the road segment as posted by department of transportation. All digital representation of a road carries this weight as one of the most important parameter. We shall discuss the usage of both the speed and intersection in details in mobility profiling section.

The detailed geographical definition of a road is simply represented as ordered pairs of shape determining points. For example a straight road may be represented just by two points whereas a curved road might take numerous points to represent it reasonably. The precision is chosen at the time of digitization of the map. The biggest cost of precision is the database memory. We shall discuss digitization in details in Section V.

## III. MOBILE TRAJECTORY ESTIMATION FRAMEWORK

Essentially, a road network is stored as Cartesian coordinate ordered pairs of points. These points are finite and have a notion of distance, Euclidean or otherwise, as a metric between them. Mobile location is represented by a point or a collection of potential location points (in an erroneous environment). These location points are associated to a mobile user traveling on a road network which is also represented by a collection of ordered points. All these points, both for location representation and road network representation, have a notion of distance, ruling over the points, therefore a metric space is a natural framework for the theory and analyses of mobile trajectory formation. It has been established in literature that the language in which a large body of ideas and results of functional analyses are expressed is that of the metric spaces [20].

In road networks the distance between any two points $P_i(x_i, y_i), P_j(x_j, y_j)$ is not necessarily Euclidian. The distance is Euclidian only if the points are connected on the same polyline. The metric (distance) for road network is the length of the polyline line from $P_i(x_i, y_i)$ to $P_j(x_j, y_j)$ as shown in Figure 2.

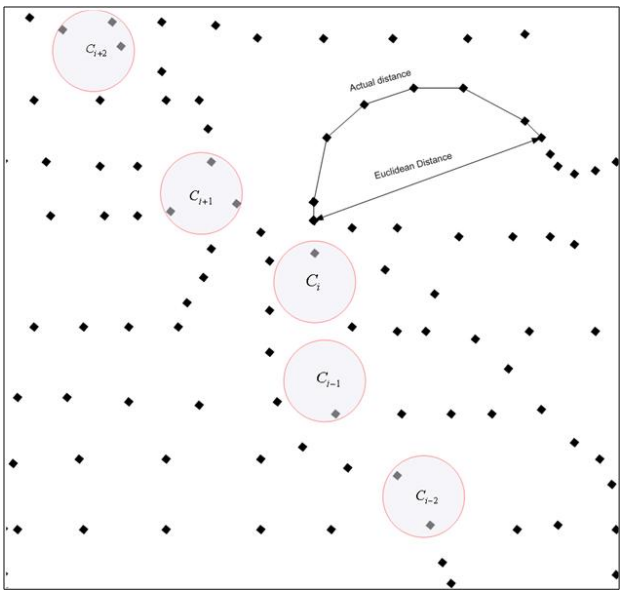



Figure 2. A logical representation of topology of metric space. The distance is through the connected points.

Definition 1: For a metric space $(\mathbf{X}, d)$, a geodesic path joining $x \in \mathrm{X}$ to $y \in \mathrm{X}$ (or more briefly, a geodesic from $x$ to $y$) is a map $c$ from a closed interval $[0, l] \subset \mathrm{R}$ to X such that $c(0) = x$, $c(l) = y$ and $d(c(t), c(t')) = |t - t'|$ $\forall t, t' \in [0, l]$ (in particular, $l = d(x, y)$). The image $\alpha$ of $c$ is called a geodesic segment with end points $x$ and $y$.

Definition 2: A geodesic space is a metric space wherein any two points are joined by a geodesic.

Definition 3: A metric space $(\mathbf{X}, d)$ is a length space if for every $x, y \in \mathbf{X}$, $d(x, y) = \inf_{\gamma} L(\lambda)$, where $\lambda$ is a rectifiable curve between $x$ and $y$.

Let $P_L \equiv v_0(x_0, y_0), v_1(x_1, y_1), ....., v_n(x_n, y_n)$ defines a polyline with n+ 1 vertex. The polyline has n edges.

$$P_L = \begin{cases} [(y_1 - y_0)x + (y_0 x_1 - x_0 y_1)]/(x_1 - x_0): & x_0 \leq x \leq x_1 \\ [(y_2 - y_1)x + (y_1 x_2 - x_1 y_2)]/(x_2 - x_1): & x_1 \leq x \leq x_2 \\ \vdots & \\ [(y_n - y_{n-1})x + (y_{n-1} x_n - x_{n-1} y_n)]/(x_{n-1} - x_n): x_{n-1} \leq x \leq x_n; & x_{i+1} \neq x_i \end{cases} \quad (1)$$

Please note $x_{i+1} = x_i$ is a special case where the segment is, $\{(x_k, y_k) \mid y_i \le y_k \le y_{i+1}; x_k = x_i = x_{i+1}\}$,
It is customary to use parametric representation of polylines: The parametric representation of (1) is:

$$P_L = \begin{cases} \frac{(t_1 - t)v_0 + (t - t_0)v_1}{t_1 - t_0}: & t_0 \le t \le t_1 \\ \frac{(t_2 - t)v_1 + (t - t_1)v_2}{t_2 - t_1}: & t_1 \le t \le t_2 \\ \vdots & \\ \frac{(t_n - t)v_{n-1} + (t - t_{n-1})v_n}{t_n - t_{n-1}}: & t_{n-1} \le t \le t_n \end{cases} \quad (2)$$

where, $|t_i - t_{i-1}| \neq 0$ ,corresponds to $|v_i - v_{i-1}|$, $\forall i$ as shown in Figure 3. Let $c:[a,b] \to V$ represents the polyline in interval [a,b] and $t_0 = a < t_1 < t_2 < \ldots\ldots < t_n = b$, the polygonal length of the polyline curve or the distance covered in $|t_n - t_0|$ is:

$$P_L(c, \{t_0, t_1, \ldots t_n\}) = \sum_{i=0}^{n} d(c(t_i), c(t_{i-1})). \quad (3)$$

Please note the polyline representation of a road is an approximation of a smooth curve which does not necessarily have these edges. The edges are referred as geodesics of the road representation. A road network, e.g., shown in Figure 3 is a set, $\mathbf{M} = \{P_{Lj} : j = 1,2,\ldots N\}$ that represents a road network of N polylines. Since our road element is a 2-dimensional model the information of intersections of the roads is not necessarily inclusive. The road may pass over another road without any intersection. Therefore connectivity matrix must accompany the set $\mathbf{M}$ for complete representation of a road network. Figure 4 is a representation of the actual road network. It is a graphical view of actual GIS database of an urban map.

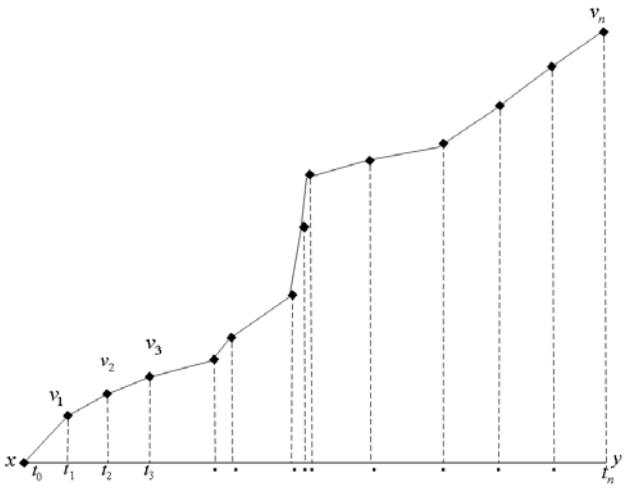



Figure 3. A parametric representation of a road element in topological space

Let $v_i(x_i, y_i)$ and $v_{i+1}(x_{i+1}, y_{i+1})$ be the end points on a geodesic and,

$$\mathbf{G}_i = \left\{ p(x,y) \mid y = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} x + \frac{y_i(x_{i+1} - x_i) + x_i(y_i - y_{i+1})}{x_{i+1} - x_i}; \right.$$
$$\left. \frac{x_i}{x_{i+1}} \le \frac{x}{x_{i+1}} \le 1; \frac{y_i}{y_{i+1}} \le \frac{y}{y_{i+1}} \le 1; x_{i+1} \neq x \right\} \quad (4)$$

In other words $\mathbf{G}_i$ contains all the points of a geodesic between two arbitrary vertices of a digital map, i.e., GIS database of the road network. $x_{i+1} = x_i$ is a special case, as explained before. From this point on this special case shall be implied whenever we state $\mathbf{G}_i$.

Let N be the total number of geodesics in a digital map, $\mathbf{G} = \bigcup_{i=1}^{N} \mathbf{G}_i \subset \mathbf{R}^2$ is set of all permissible points on a road network map. Let $d$ be a metric of distance between two locations $P_1$ and $P_2$ on the road network map. For all $x, y \in \mathbf{G}$, $d(x,y) = \inf_{\gamma} L(\lambda)$, where $\lambda$ is rectifiable curve between $x$ and $y$.

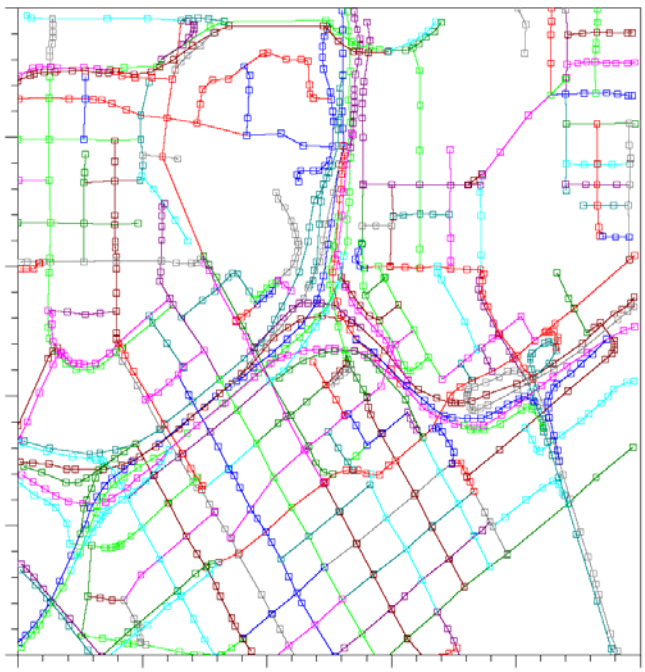


Figure 4. Topological representation of road map of an urban area of St.Paul, Minnesota. Roads are represented by connected and intersecting polylines. Polylines are expressed in vertices shown as squares.

We are interested in the analysis of $(\mathbf{G}, d)$ from metric space point of view. We call $(\mathbf{G}, d)$ Mobile User space. A Mobile User space, $(\mathbf{G}, d)$, is a non-symmetric metric space because $d(x,y) \neq d(y,x)$ always.

**Lemma 1:** A Mobile User space $(\mathbf{G}, d)$, is not a geodesic space.

**Proof:** Let $x, y \in \mathbf{G}$ , $d(x,y) = \| x, y \|$ , where $\|.\|$ is Euclidian distance, iff $x$ and $y$ belong to the same geodesic. But $x, y$ are arbitrary points,

$\therefore d(x,y) \neq \| x, y \|$ for any arbitrary points $x, y \in \mathbf{G}$. Hence $(\mathbf{G}, d)$ is not a geodesic space.

### *A. Closed Ball Embedded in Mobile User Space*

In all mobile location technologies, the estimated location is not an exact point. If exact location coordinates are claimed there is always an error associated with it. There are many factors contributing the errors. No single technology can provide a pin point location of the mobile user using the existing infrastructure of mobile systems. Instead of an exact location of a mobile user we know the feasible location area of the mobile user. One cannot come up with the exact shape of the feasible area; however there is always an upper bound on the exact area of the region where a mobile user is located. In our research we model this feasible area with a circle. The location probability density is constant throughout the circle. In some research work, a model with Gaussian distribution around the center of the circle has been used [21]. In real world scenario such model would lead to erroneous results when dealing with identifying the several candidate roads a mobile user is potentially on, which pass through the feasible location area. Different location estimation technologies have different accuracy associated with it. Assisted-GPS technique is the most accurate among all, TOA and Time Advance being among the least. We shall represent them with circles of corresponding error radius. When a circular feasible area is considered on a road map (mobile user space), this leads to a topology where a ball is embedded in a mobile user space. We shall discuss and analyze this topological space in details and derive some results, to be used in the grand scheme of mobility profiling. In this subsection we define the key terms used in our framework and formulate geodesic candidacy. This will cover how candidate road segments are selected.

Let $(\mathbf{G}, d)$ be a mobile user space and a ball $\mathbf{B}_{\mu}(x_0) = \{\mathbf{x} \in X : d(\mathbf{x}, x_0) \leq \mu\}$ being embedded in this metric space. Here *X* is a universal set. The ball, $\mathbf{B}_{\mu}(x_0)$, is essentially the feasible circular location area with center at $x_0$ and radius equal to $\mu$, estimated by a mobile wireless system. When a request for location estimation is made to the cellular system it shall return the location as, $x_0$.

**Location Error Proposition:** The embedding ball, $\mathbf{B}_{\mu}(x_0)$ intersects or contains at least one geodesic of $\mathbf{G}$, i.e., $\mathbf{B}_{\mu}(x_0) \bigcap \mathbf{G} \neq \varnothing$.

**Proof:** Let $\mathbf{G}$ contain N geodesics $\Rightarrow \mathbf{G} = \bigcup_{i=1}^{N} \mathbf{G}_i$. Where,

$$\mathbf{G}_i = \{p(x_i, y_i) \mid y_i = mx_i + b; \mathrm{u}_k \leq x_i \leq \mathrm{u}_{k+1}; \mathrm{v}_k \leq y_i \leq \mathrm{v}_{k+1}\}$$

Let $\mathbf{G}_l = \bigcup_{i=1}^{l} \mathbf{G}_i$ be the geodesics surrounding the ball, $\mathbf{B}_{\mu}(x_0) = \{\mathbf{x} \in X : d(\mathbf{x}, x_0) \leq \mu\}$. If $x_l \in \mathbf{G}_l$, by definition of the location error, $d(x_l, x_0) \leq \mu$, $\Rightarrow x_l \in \mathbf{G}_l$

$\because \mathbf{G}_l \subseteq \mathbf{G}$, $x_l \in \mathbf{B}_{\mu}(x_0) \bigcap \mathbf{G}$

Hence $\mathbf{B}_{\mu}(x_0) \bigcap \mathbf{G} \neq \varnothing$.

We are interested in conditions imposed on geodesics $\mathbf{G}_i$ for passing through the ball. The reason of the investigation is to map the erroneous location to the nearby road. Map matching corrects the location of a mobile user by projecting the estimated location to a road which user is most likely on.

Let $\mathbf{V} = \{v_i : i = 1, 2, .....N+1\}$ be the vertices of $N$ polylines in $\mathbf{G}$ and $\overset{\odot}{E}$ be the error in location estimation, the associated ball, let us call it error ball from this point on, is $\mathbf{B}_{\overset{\odot}{E}}(x_0) = \{\mathbf{x} \in X : d(\mathbf{x}, x_0) \leq \overset{\odot}{E}\}$. We shall define two important categories of geodesics:

**Definition 1:** A geodesic $\mathbf{G}_i$, as in (4), of any two vertices, $v_i(x_i, y_i)$, $v_{i+1}(x_{i+1}, y_{i+1})$, of a road, is said to be a candidate geodesic if it touches any point of the error ball, i.e., $\mathbf{G}_i \bigcap \mathbf{B}_{\overset{\odot}{E}}(\mathbf{x}_0) \neq \varnothing$.

**Definition 2:** A geodesic, $\mathbf{G}_i$, of any two vertices, $v_i(x_i, y_i)$, $v_{i+1}(x_{i+1}, y_{i+1})$, of a road, whose straight line extension,

$$\mathbf{L}_i = \left\{ p(x,y) \mid y = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} x + \frac{y_i(x_{i+1} - x_i) + x_i(y_i - y_{i+1})}{x_{i+1} - x_i}; x_i \neq x_{i+1} \right\}, \quad (5)$$

touches any point of the error ball, is said to be pseudo candidate geodesic, i.e., $\mathbf{L}_i \bigcap \mathbf{B}_{\overset{\odot}{E}}(\mathbf{x}_0) \neq \varnothing$.

**First Condition of Geodesic Candidacy:** A simple and trivial geodesic candidacy condition is: $\mathbf{W} \bigcup \mathbf{B}_{\overset{\odot}{E}}(\mathbf{x}_0) = \mathbf{B}_{\overset{\odot}{E}}(x_0)$, where $\mathbf{W} = \{v_j\}$ for some $j$. Please note $\mathbf{W} = \{\varnothing\}$ is not a valid scenario. If a vertex lies within the error ball the corresponding geodesic shall be a

candidate geodesic, however a geodesic may pass through the error ball but the vertices could lie outside the error ball. For the study of geodesic candidacy when the vertices are outside of the error ball we would need to study pseudo geodesic candidacy.

**Pseudo Geodesic Candidacy Theorem:** In a mobile user metric space, $(\mathbf{G}, d)$, the necessary and sufficient condition for pseudo geodesic candidacy of any arbitrary vertices $v_i(x_i, y_i), v_{i+1}(x_{i+1}, y_{i+1}) \in \mathbf{B}'_{\underset{E}{\odot}}(\mathbf{x}_0)$, for an error ball, $\mathbf{B}_{\underset{E}{\odot}}(x_0) = \{\mathbf{x} \in X : d(\mathbf{x}, x_0) \le \overset{\odot}{E}\}$ is

$$\left| \frac{y_r(x_{i+1} - x_i) - x_r(y_{i+1} - y_i) + x_i y_{i+1} - x_{i+1} y_i}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}} \right| \le \overset{\odot}{E}, \quad (6)$$

where $x_0(x_r, y_r)$ is the center of the circle, i.e., the estimated location of a mobile user and $\overline{\mathbf{W}} = \mathbf{B}'_{\underset{E}{\odot}}(x_0)$.

**Proof:** Let $v_i \equiv (x_i, y_i)$ and $v_{i+1} \equiv (x_{i+1}, y_{i+1})$ be the vertices of a polyline such that $v_i, v_{i+1} \in \mathbf{W}$ make a pseudo geodesic candidate. $\overline{KL}$ in Figure 5 is such an example.

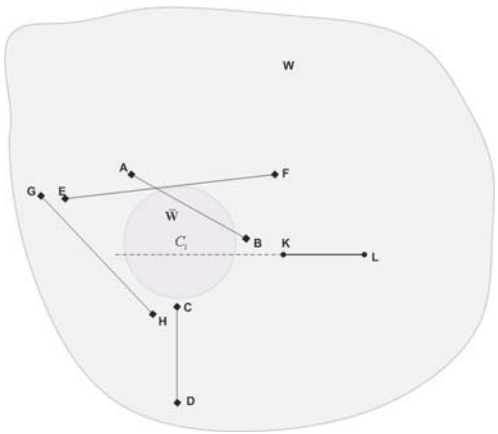



Figure 5. *Straight lines through some vertices outside the error ball may pass through it*

It follows from Definition 2 that a geodesic is a pseudo candidate if and only if its straight line extension, $\mathrm{L}_i$, given in (5) and $C_{\underset{E}{\odot}}(\mathbf{x}_0) = \{\mathbf{x} \in X : d(\mathbf{x}, \mathbf{x}_0) = \overset{\odot}{E}\}$ have a solution $\{p_1, p_2\} \equiv \{A \pm Q, B \pm mQ\}$, where $A$ and $B$ are constants and

$$Q = \frac{\sqrt{\overset{\odot}{E}^2 - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2}}{\sqrt{1+m^2}}. \quad (7)$$

Here, and throughout this paper, $m$ represents the slope of a geodesic and $b = \frac{y_i(x_i - x_{i+1}) + x_i(y_{i+1} - y_i)}{x_i - x_{i+1}}$.

For the solution, $\{p_1, p_2\}$, to be real, the discriminant $\overset{\odot}{E}^2 - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2$, in (7) must be non-negative.

$$\therefore \overset{\odot}{E}^2 - \left[ \frac{y_r(x_{i+1} - x_i) - x_r(y_{i+1} - y_i) + x_i y_{i+1} - x_{i+1} y_i}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}} \right]^2 \ge 0$$

$$\Rightarrow \left| \frac{y_r(x_{i+1} - x_i) - x_r(y_{i+1} - y_i) + x_i y_{i+1} - x_{i+1} y_i}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}} \right| \le \overset{\odot}{E}$$

**Corollary 1**: For a vertical geodesic the pseudo candidacy requirement is $|x_i - x_r| \le \overset{\odot}{E}$. The corollary easily follows by putting the vertical condition: $x_i = x_{i+1}$ in the above condition.

**Corollary 2**: For a horizontal geodesic the pseudo candidacy requirement is $|y_r - y_i| \le \overset{\odot}{E}$. The corollary easily follows by putting the vertical condition: $y_i = y_{i+1}$ in the above condition.

### *B. Estimated Location Correction within an Error Ball*

Once the candidate geodesics are identified, the estimated erroneous location can be corrected by mapping it to the candidate geodesics. The estimated location is presented as an error ball, $\mathbf{B}_{\underset{E}{\odot}}(x_0) = \{x \in X : d(\mathbf{x}, \mathbf{x}_0) \le \overset{\odot}{E}\}$, with center $x_0(x_r, y_r)$ and feasible area of $\pi \overset{\odot}{E}^2$. The location coordinates are the center of the error circle, $x_0$. Since a mobile user traverses on road, the feasible areas transform to feasible locations on the candidate geodesic contained in the error ball.

The feasible location now is bounded by the lines $\mathbf{L}_i$ within the error circle. The feasible location is given by, $\bigcup_{i=1}^{l} \mathbf{L}_i = \mathbf{B}_{\underset{E}{\odot}}(\mathbf{x}_0) \cap \left( \bigcup_{i=1}^{l} \mathbf{L}_i \right)$, where $i = 1, 2, 3, \ldots . l$ and $l$ is the number of candidate geodesics of the error ball.

If $v_i(x_i, y_i)$, $v_{i+1}(x_{i+1}, y_{i+1}) \notin \mathbf{B}_{\underset{E}{\odot}}(\mathbf{x}_0)$, we shall use the center of the line segment, $\mathbf{L}_i$ as the corrected location on $\mathbf{L}_i$ since that is the average of all the possible location points on the line segment, $\mathbf{L}_i$. The corrected location is

essentially projection of the center of the error ball on the line segment, $\mathbf{L}_i$ .

To find the corrected location we solve the line, $(y_{i+1}-y_i)x+(x_i-x_{i+1})y+x_{i+1}y_i-x_iy_{i+1}=0$, passing through the vertices, $v_i(x_i,y_i)$, $v_{i+1}(x_{i+1},y_{i+1})$ and the normal, $(x_i-x_{i+1})x+(y_i-y_{i+1})y+x_r(x_{i+1}-x_i)+y_r(y_{i+1}-y_i)=0$, passing through the center , $x_0(x_r,y_r)$, of the error ball. The corrected location is:

$$P_c(x_c,y_c)=\left(\frac{x_r+m(mx_i-y_i+y_r)}{m^2+1},\frac{y_i+m(my_r-x_i+x_r)}{m^2+1}\right). \quad (8)$$

In case the vertices of the polyline lie within the error ball, i.e., $v_i(x_i,y_i)\in \mathbf{B}_{\underset{E}{\odot}}(\mathbf{x}_0)$ for $i=1,2,3,..k$, then the corrected location is the vertex $v_i$ for which, $\|x_0-v_i\|$ is minimum.

**Lemma 2**: In a mobile user metric space, $(\mathbf{G},d)$ , if a geodesic of vertices $v_i(x_i,y_i),v_{i+1}(x_{i+1},y_{i+1})\in \mathbf{B}'_{\underset{E}{\odot}}(\mathbf{x}_0)$; and their corresponding geodesic satisfies pseudo candidacy as given in (6) and $0\le(x_{i+1}-x_i)(x_r-x_i)+(y_{i+1}-y_i)(y_r-y_i)\le D^2$, where $D$ is the Euclidean distance between the vertices, the geodesic is a candidate geodesic.

**Proof:** Let $v_i(x_i,y_i)$ and $v_{i+1}(x_{i+1},y_{i+1})$ be the vertices of a polyline such that $v_i,v_{i+1}\in \mathbf{W}$ make a geodesic candidate. $\overline{AB}$, and $\overline{EF}$ in Figure 5 are such examples. Let $x_0(x_r,y_r)$ be the center of the error ball.

The corrected location, $P_c(x_c,y_c)$, of the estimated location, i.e., the projection of the center of the error ball onto the geodesic is as shown in (8) above. $P_c(x_c,y_c)$, $v_i(x_i,y_i)$ , and $v_{i+1}(x_{i+1},y_{i+1})$ are collinear points. Let $P_c(x_c,y_c)$ divides the geodesic with ratio $\frac{Q}{D}:1$ as shown in Figure 6.

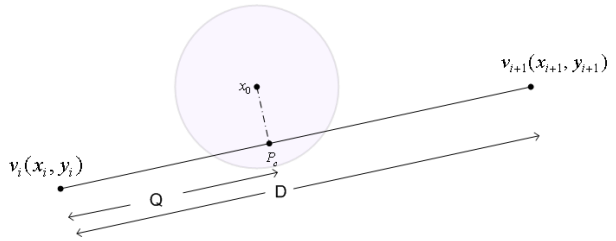



Figure 6. The corrected location must lie between the two vertices for the geodesic to be candidate geodesic

$\therefore Q=\frac{x_cD-Dx_i}{x_{i+1}-x_i}=\frac{y_cD-Dy_i}{y_{i+1}-y_i}$. For the geodesic to be the candidate geodesic, $Q$ must be positive and less than D. For $Q=D$ or $Q=0$, the geodesic is tangent to the error circle. For negative value of $Q$ , the geodesic shall be a pseudo candidate or may not intersect the error circle at all. So, for the geodesic to be a candidate geodesic the following condition must meet:

$0\le Q\le D$ where $D=|v_{i+1}-v_i|$

$$\frac{x_cD-x_iD}{x_{i+1}-x_i}=\frac{x_r+m(mx_i-y_i+y_r)}{\sqrt{m^2+1}}-x_i\sqrt{m^2+1}$$
$$=\frac{(x_r-x_i)+m(y_r-y_i)}{\sqrt{m^2+1}}$$

From the candidacy condition,

$$0\le\frac{(x_r-x_i)+m(y_r-y_i)}{\sqrt{m^2+1}}\le D$$
$$\Rightarrow 0\le\frac{(x_r-x_i)+\left(\frac{y_{i+1}-y_i}{x_r-x_i}\right)(y_r-y_i)}{\sqrt{\left(\frac{y_{i+1}-y_i}{x_{i+1}-x_i}\right)^2+1}}\le D$$
$$\Rightarrow 0\le(x_{i+1}-x_i)(x_r-x_i)+(y_{i+1}-y_i)(y_r-y_i)\le D^2 \quad (9)$$

Here, the candidacy condition is proved using $Q=\frac{x_cD-Dx_i}{x_{i+1}-x_i}$. We would get the same result for $Q=\frac{y_cD-Dy_i}{y_{i+1}-y_i}$, as well.

**Corollary 3**: For a vertical geodesic the candidacy requirement is, $0\le(y_{i+1}-y_i)(y_r-y_i)\le D^2$. The corollary easily follows by putting the vertical condition: $x_i=x_{i+1}$ in the above condition.

**Corollary 4**: For a horizontal geodesic the candidacy requirement simplifies to, $0\le(x_{i+1}-x_i)(x_r-x_i)\le D^2$ . The corollary easily follows by putting the horizontal condition: $x_i=x_{i+1}$ in the above condition.

There are several different approaches proposed in literature to identify the candidate road segments from the digital map against an estimated location. Most of them deal with GPS or GPS integrated with dead reckoning and hence are suitable only under those environments. One of the benefits of GPS from identifying a candidate road segment point of view is its known feasible area. Several methods are available to determine the feasible area. Variance-covariance associated with GPS receiver output has been used to define the feasible area as ellipse with known distribution function [22][23]. Secondly, the error is small and symmetrical;

therefore inherently it is less complex to identify the candidate road segments as compared to mobile network. Segment based methods where the heading of the vehicle from GPS points is compared with the road segments in the digital map for maximum parallelism. The road segments that are along the same direction as the line joining the GPS points are marked as the candidate road segments. This method has been used along with other measures [24][25] and as such [26]. Because of the small error and error symmetry this measure is considered quite reliable. However this is not suitable for mobile network, where the error is relatively large and not symmetrical. For example consider Figure 7. A vehicle moves from estimated location $L_1$ to $L_2$ in a mobile network. Clearly segment $s_2$ is the candidate road segment because $s_2$ passes through both the location feasible areas, however using direction similarity, road segment $s_1$ is picked wrongfully. Other disadvantage of this method in mobile system is that we need to have two locations polled very close to one another to identify the candidate road segments.

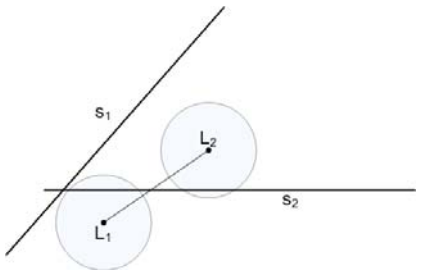



Figure 7. Vehicle heading from $L_1$ to $L_2$ wrongfully matches the road segment $s_1$

[27] identifies the candidate segment from projection of the estimated location on to the road segments in the digital map. Let $S$ and $E$ be the two vertices (also referred to as shaping points) of a geodesic and $G$ be the estimated location point. Define

$$K = \frac{\overrightarrow{SG}.\cos\theta}{\left|\overrightarrow{SE}\right|} = \frac{\overrightarrow{SG}.\overrightarrow{SE}}{\left|\overrightarrow{SE}\right|^2} \qquad (10)$$

If $0 \le K \le 1$, road segment $\overline{SE}$ (Figure 8) is selected, if $K < 0$, the road segment preceding the node $S$ is selected, if $K > 1$ the road segment following the node $E$ is selected. The drawback with this method is that it does not take into account the error region. Let us consider a comparison in Figure 8.

It presents two scenarios. In scenario 1, the selected road segment is a valid one, whereas in scenario 2, the selected road segment is not a valid candidate segment contrary to the proposed condition. The candidate selected in scenario 2 is definitely related to some other error circle, since $\overline{SE}$ does not even touch the error circle. Using this method erroneous geodesic shall be identified as candidates. This method is only valid for identifying a candidate road segment among the road segments that are fully or partially contained within the GPS feasible area.

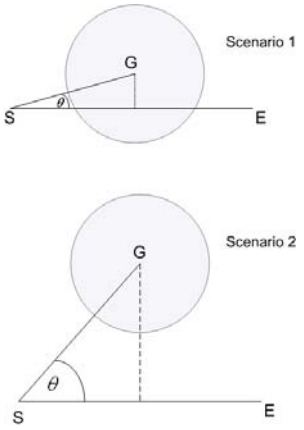



Figure 8. In both the projection scenario $\overline{SE}$ is selected as the candidate segment

[28] identifies the above mentioned flaw in general in finding the candidate geodesic when the vertices of geodesic (or arc in GPS literature) are outside the feasible are. It proposes this problem as future research to address such issue. [12] proposes a method that is along the line of proposed research.

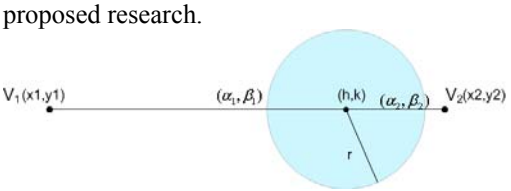



Figure 9. $\overline{V_1V_2}$ is a candidate geodesic in an error circle of radius r centered at (h,k)

The proposed conditions to identify a candidate geodesic are as follow:

First find the solution $(\alpha, \beta)$ for the equation of the error circle, $(x-h)^2 + (y-k)^2 = r^2$ and the straight line $\overline{V_1V_2}$, Figure 9. Here,

$$\alpha(\alpha_1, \alpha_2) = \frac{m^2k + h - mc}{1+m^2} \pm \frac{\sqrt{r^2 - g^2}}{\sqrt{1+m^2}}, \qquad (11)$$

$$\beta(\beta_1, \beta_2) = \frac{m^2k + mh + c}{1+m^2} \pm \frac{m\sqrt{r^2 - g^2}}{\sqrt{1+m^2}}, \qquad (12)$$

where, $m = \frac{\Delta y}{\Delta x}$, $c = \frac{x_1y_2 - y_1x_2}{\Delta x}$ and $g = \left(\frac{(k - mh - c)}{\sqrt{1+m^2}}\right)$

The conditions for candidacy are run online between polls. The calculations must be fast with as little run time as possible. Secondly, depending upon the size of the digital map and the search method, there could be very large

number of comparisons and processing required before the next poll. The author proposed for further research to identify candidate segments without solving (11) and (12). Our method identifies candidate geodesics without going through the process of finding $(\alpha,\beta)$ therefore cutting significant processing. Secondly, the conditions do not specify which particular solution point, $(\alpha_1,\beta_1)$ or $(\alpha_2,\beta_2)$ to be used for verification. The results would be different for different solution points. This is extremely important from implementation point of view.

### C. Discrete Error Balls

The simplest way to form a trajectory is by interpolating the location points collected between the start point and the destination. If we have frequent locations with very low error, hence a large number of points to interpolate, we may potentially come up with a trajectory close to the actual path. However this paper is on forming a trajectory with optimal number of points. As a matter of fact, our trajectory stems from sequence of discrete error balls. How far apart and frequent the error balls are, corresponds to the polling frequency. This is the frequency with which the mobile user's location is estimated in terms of error ball. The center of the error ball is the estimated location. The polling frequency is directly related to the topology and geometry of the road network the mobile user is traveling on. If a road network is dense and there are numerous intersections, we would need more embedding error balls to estimate the trajectory of a mobile user. On the other hand if roads are sparse with less intersection, for example rural areas, we need fewer error balls to form a trajectory.

Let an error ball be located at $C_i$ at time $\tau_i$. The next location of the error ball, $C_{i+1}$, will be estimated at time, $\tau_{i+1}$, using a method, we called it Intersection Polling Method (IPM). Let $\mathbf{B}_{\overset{\odot}{E}}(C_i)=\{\mathbf{x}\in X : d(\mathbf{x},C_i)\le \overset{\odot}{E}\}$ be the error ball located at $C_i(x_r,y_r)$, at $\tau_i$. Let $v_i(x_i,y_i)$ and $v_{i+1}(x_{i+1},y_{i+1})$ be two points on a candidate geodesic of the error ball. The corrected location on the geodesic as mentioned above, in (8) is $P_c(x_c,y_c)=\left(\frac{x_r+m(mx_i-y_i+y_r)}{m^2+1},\frac{y_i+m(my_r-x_i+x_r)}{m^2+1}\right)$. For example in Figure 10, the corrected locations on the candidate geodesics are $P_{c1}$ and $P_{c2}$. Let $\{I_i,I_{i+1}\}$ be the nearest intersections of the road which the geodesic is part of, and $s_i$ be posted speed of the road segment. The next position of the error ball is estimated at

$$\tau_{i+1}=\min_{k\in\{i,i+1\}}\frac{1}{s_i}d\left(P_c,I_k\right), \tag{13}$$

where $d$ is the metric distance of the Mobile space.

In scenarios where we have multiple candidate geodesics, as shown in Figure 10, the next position of the error ball is estimated at

$$\tau_{i+1}=\min\left\{\min_{k\in\{i,i+1\}}\frac{1}{s_i}d\left(P_{c1},I_k\right),\min_{k\in\{j,j+1\}}\frac{1}{s_j}d\left(P_{c2},I_k\right)\right\}, \tag{14}$$

where $s_i$ and $s_j$ are posted speeds on road segments, $L_i$ and

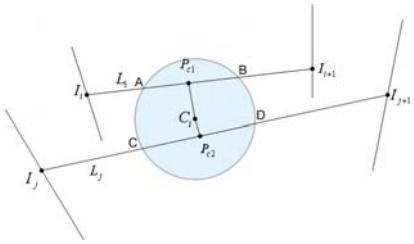



Figure 10. *The next position of the error ball is estimated at the nearest intersection from the corrected locations.*

$L_j$, respectively. If the roads segments are directional (one-way) then the candidate intersections are limited to the ones that are towards the directions of the road. In Figure 10 if the road segments are, $\overrightarrow{I_iI_{i+1}}$, and $\overrightarrow{I_{j+1}I_j}$, the candidate intersection are $\{I_{i+1},I_j\}$. In case there is only one geodesic passing through the error circle, the direction of motion and hence the candidate intersection assumed is towards the opposite end of the geodesic from the corrected points. It will help in eliminating the intersection that has been a candidate in previous estimation.

Let us consider the segment, $L_i$ .The probability of the mobile user is the same on all points of $\overline{AB}$. For the estimation of $\tau_{i+1}$ towards $I_i$, if $P_{c1}$ is taken as the start point for the estimation of $\tau_{i+1}$ but the mobile user is closer to point $A$, the time, $\tau_{i+1}$, would be longer than the required one. To fix this problem we need to further refine our approach of intersection polling method. In the refined approach instead of assuming the present location of the mobile user, i.e., the location of mobile user at $\tau_i$, we assume that the present location is the intersection of the geodesic and the error circle. We call this intersection point shifted projection point. Since there are two such shifted projection points, for example A,B for geodesic $L_i$ and C,D for geodesic $L_j$ in the figure above, we choose the shifted projection point that is closer to the road intersection. The shifted projection points of the geodesic, $\mathbf{G}_i$ with the error circle $\mathbf{C}_{\overset{\odot}{E}}(C_i)=\{\mathbf{x}\in X : d(\mathbf{x},C_i)=\overset{\odot}{E}\}$ are

$$\mathbf{C}_{\overset{\odot}{E}}(C_i)\cap \mathrm{G}_i=\{w_1,w_2\}. \tag{14}$$

Where, $w_1 = \left( \frac{my_r + x_r - mb}{1+m^2} + \frac{\sqrt{\odot^2 E - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2}}{\sqrt{1+m^2}}, \right.$ (15)

$\left. \frac{m^2 y_r + mx_r + b}{1+m^2} + \frac{m\sqrt{\odot^2 E - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2}}{\sqrt{1+m^2}} \right),$

$w_2 = \left( \frac{my_r + x_r - mb}{1+m^2} - \frac{\sqrt{\odot^2 E - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2}}{\sqrt{1+m^2}}, \right.$ (16)

$\left. \frac{m^2 y_r + mx_r + b}{1+m^2} - \frac{m\sqrt{\odot^2 E - [(y_r - mx_r - b)/\sqrt{1+m^2}]^2}}{\sqrt{1+m^2}} \right),$

The refined IPM dictates that, the location is updated at $\tau_{i+1} = \min_{k \in \{i, i+1\}} \frac{1}{s_i} d(w_i, I_i)$. We pick $w_i$ for which $d(w_i, I_i)$ is minimum.

Let us briefly analyze the performance of Intersection Polling method in terms of number of polls. The performance of the polling method is directly tied to the topology of the map. Topological factors that directly affect the performance of this method are: the number of intersections during the course of a path and direction of the roads. This polling method would perform the best on longs directional (one-way) roads or the on the grid-like roads, directional or non-directional. Grid-like roads are directional in general anyway. Assuming the start and destination points are arbitrarily anywhere on the roads but not on the intersection, the number of polls required for a path containing N intersections for a grid-like road network are $N+1$. The advantage comes from the fact that after only one polling the next intersection is identified because the next three intersections are approximately same distance away. Figure 11 depicts part of two-way grid-like road network. The next intersection from L is either A, B or C. Since A, B and C are almost equidistance from L, the polling time for each of them is the same. At the next polling time, any of the intersections, A, B or C, which is contained in the error ball, is map-matched as the next intersection of the path. Therefore for each intersection in the trajectory, only one location polling is enough.

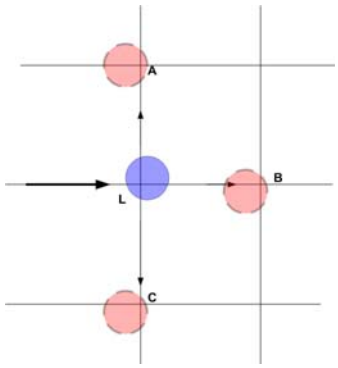



Figure 11. *Selection of next intersection in a grid-like road network.*

On directional roads of arbitrary layout, the highest number of polls for a path is $2N+1$. In this scenario going from one intersection to the other, would require two polls, one for the nearer intersection, the second one for the remaining intersection, i.e., the farther one. Please see Figure 12. Going from the present intersection L, to B, we need to poll once at $A^*$ after time $\tau_1$, required to reach intersection A and second poll at B. The number of polls could be reduced to $N+1$ with a minor modification. The modification is; after polling for the nearer intersection, A, after time $\tau_1$, pick the other intersection after time $\tau_2 - \tau_1$ without polling, where $\tau_2$ is the travel time from L to B. The drawback of this approach is that the destination would always be an intersection. To continue with performance analyses, we stick with our original approach. The lowest number of polls in this scenario is $N+1$. Here, at each polling the nearer intersection is map matched, for example intersection A, in Figure 12. So, the average number of polls on directional roads with arbitrary layout

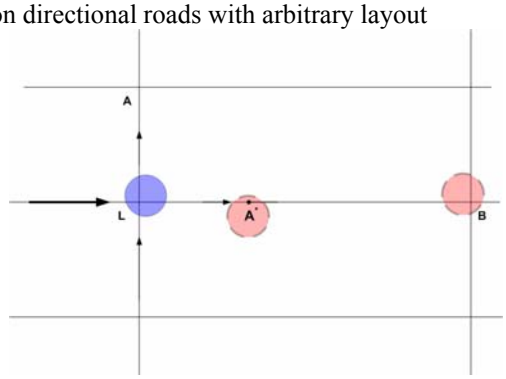



Figure 12. *Selection of next intersection in a directional road network.*

is $\frac{3}{2}N+1$. For non-directional with arbitrary layout road network, going from one intersection to the next one, we would require maximum three polls. One poll for the nearest intersection, the second one for the nearer of the remaining two intersections and the third one for the last intersection. This scenario is depicted in Figure 13. Here, $\overline{LA} < \overline{LB} < \overline{LC}$. Selection of intersection $C$ is the worst case scenario. Intersection $A$ will be polled first, then $B$ and C respectively. So, we have to poll three times between each intersection in this scenario. So, total number of highest possible polls for a non-directional arbitrary layout of road network is $3N+1$. The lowest number of polls on a path on this network would be $N+1$, as already explained in previous scenarios. Therefore the average number of polls for a path on such road network would be $2N+1$.

Since GPS location estimation does not cost any additional air resources, estimating the course of trajectory with least number of GPS location points has not been an area of interest. Furthermore most of the vehicles equipped

with GPS have DR (Dead Reckoning) integrated also. DR keeps tracks of the length and absolute heading of the displacement vector from previous known position. So, the present estimated location of the vehicle is known at almost all the times. Estimating the trajectory of a vehicle with numerous closely packed location points which have low symmetrical error is far easier than with those which are sparse with relatively larger error especially in urban areas.

Let us define an absolute ideal polling method that requires the least number of polling yet does not miss any turns. Most of the proposed methods in GPS world are arc-based (segment based). Such an ideal method would need at least one location point for a directional arc and two location points for a non-directional arc. We assume that this polling method is robust and can handle all the possible topologies of the digital map. An arc is defined as the road segments between two intersections. So, we can estimate the

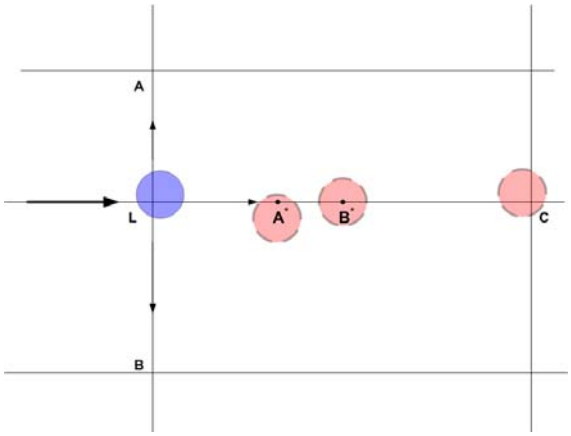



Figure 13. *Selection of next intersection in a two-way road network.*

performance of such an ideal polling method in terms of number of polls. This ideal poller would require $\frac{3}{2}N+1$ polls for an arbitrary path. Let us compare the average performance of Intersection Polling method with this ideal poller. The mean of number of polls required for an arbitrary path in all the three scenarios mentioned above is:

$\frac{(N+1)+\left(\frac{3}{2}N+1\right)+(2N+1)}{3}=\frac{3}{2}N+1$, which is equal to the number of polls of an ideal poller using arc-based map matching methods as defined above.

### *D. Error Ball's Directional Oreintation*

In this section we introduce the notion of error ball's image. An image of an error ball is, simply, an instance of its previous embedding. The trajectory of a mobile user is essentially estimated from the interpolation of several images of the embedded error ball. So, at a given point during the trajectory formation we have an embedded error ball and several of its images as shown in Figure 14. In the beginning, obviously, we just have the embedded ball, images are yet to be imprinted. Angle is one of the most important metric for comparing direction. We are interested in comparing the directional orientation of the transition of the error ball with respect to the actual path segments. We call this angle, direction difference. This direction difference is inscribed between the line joining the center of the embedded error circle and its image and a geodesic. In case there are multiple geodesics in the polyline and we are interested in finding the direction difference of this particular path segment and the transition of error ball, we connect the start point (vertex) to the end point of the path segment. The angle it makes with the reference, x-axis we call it the actual direction of motion during that interval.

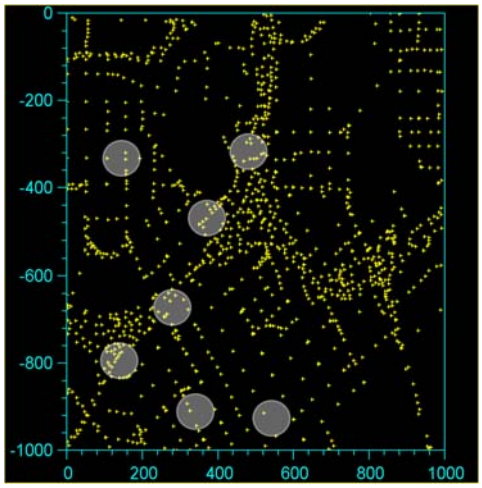



Figure 14. A graphical representation of our GIS database, error ball and its images.

Since the center of error ball, $C_1$, and that of its image, $C_0$ (Figure 15), are not the actual locations of a mobile user at time $\tau_{i+1}$ and $\tau_i$, respectively, the direction difference will not be accurate. There is an offset in the direction difference.

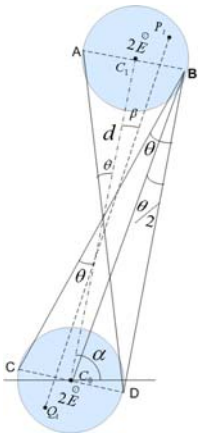



Figure 15. *Illustration of direction of transition of an error ball, $\alpha$ , the direction difference off set, $\theta$ ,and difference error, $\beta$ .*

In the figure above $\mathbf{B}_{\underset{E}{\odot}}(C_1)$ is an error ball centered at $C_1$ and $\mathbf{B}_{\underset{E}{\odot}}(C_0)$ is its previous image with center at $C_0$. Let $P_i \in \mathbf{B}_{\underset{E}{\odot}}(C_1)$ and $Q_i \in \mathbf{B}_{\underset{E}{\odot}}(C_0)$ be the two actual locations of the mobile user, $\angle\left(\overline{C_0C_1}, \overline{Q_iP_i}\right) = \beta$ is the direction error. We define the maximum offset error of an error ball as,

$$\theta = \max_{j \in \mathbf{B}_{\underset{E}{\odot}}(C_1)} \left[ \max_{i \in \mathbf{B}_{\underset{E}{\odot}}(C_0)} \angle\left(\overline{C_0C_1}, \overline{Q_iP_j}\right) \right]. \quad (17)$$

As illustrated in the figure above the maximum offset error is transcribed by the extreme points on the diameters of the two circles, which are diagonally opposite. For example C,B and A,D are such extreme points. Therefore the maximum offset,

$$\theta = \tan^{-1}\left(\frac{2\overset{\odot}{E}}{d}\right). \quad (18)$$

Please note $\theta$ is symmetrical, i.e., $\angle\left(\overline{C_0C_1}, \overline{BC}\right) = \angle\left(\overline{C_0C_1}, \overline{AD}\right) = \theta$, and it can be easily seen that $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$.

### *E. Direction Negativity*

Let $\gamma_1 : p_1, q_1$ be a path segment with start point $p_1$ and end point $q_1$ and $\gamma_2 : p_2, q_2$ be another path segment with start and end points $p_2$ and $q_2$, the angle between the two path $\angle(\gamma_1, \gamma_2) = \angle\left(\overline{p_1q_1}, \overline{p_2q_2}\right)$ is called the direction difference between paths $\gamma_1$ and $\gamma_2$. A direction negativity exists between two paths $\gamma_1$, $\gamma_2$ if $-\frac{\pi}{2} < \angle(\gamma_1, \gamma_2) < \frac{\pi}{2}$. Please note the range of $\angle(\gamma_1, \gamma_2)$ is $[0, \pi]$ or $[0, -\pi]$. The maximum negativity is reached when;

$$\angle(\gamma_1, \gamma_2) = \begin{cases} \pi \\ or \\ -\pi \end{cases}$$

This can be clearly visualized if you think one of the paths along the y-axis, i.e., a vertical path, the other path along the positive or the negative side of the x-axis. On either sides of the second path does not contain any y-component in their path. If the direction remains within first quadrant for the positive side of the path and remains within second quadrant for the negative side of the second path, they both have y-components that are positive. If the second path that is along the positive side of the x-axis moves towards fourth quadrant and the one along the negative side of the x-axis move towards the third quadrant, we start getting the negative y-components in the paths, i.e., they move into direction negativity. If we need to find the angle between a path, $\gamma$, and the transition of the error ball, $\angle\left(\gamma, \overrightarrow{\mathbf{B}_{\underset{E}{\odot}}(C_1)}\right)$, we need to add the offset to $\angle\left(\gamma, \overline{C_0C_1}\right)$. In a worst case scenario the offset is, $\theta = \tan^{-1}\left(\frac{2\overset{\odot}{E}}{d}\right)$.

$$\Rightarrow \max \angle\left(\gamma, \overline{\mathbf{B}_{\underset{E}{\odot}}(C_1)}\right) = \angle\left(\gamma, \overline{C_0C_1}\right) + \tan^{-1}\left(\frac{2\overset{\odot}{E}}{d}\right). \quad (19)$$

where, $d$ is the displacement of the transition of the error ball.

The condition for direction negativity between a path, $\gamma$, and transition of the error ball can be easily formulized using inner product. The condition is

$$\frac{\pi}{2} < \cos^{-1}\left(\frac{(x_j - x_i)(x_1 - x_0) + (y_j - y_i)(y_1 - y_0)}{\sqrt{\left[(x_j - x_i)^2 + (y_j - y_i)^2\right]\left[(x_1 - x_0)^2 + (y_1 - y_0)^2\right]}}\right) < \frac{-\pi}{2}, \quad (20)$$

where $(x_i, y_i)$ and $(x_j, y_j)$ are the end points of the path (or geodesic), $\gamma$, and $(x_1, y_1)$, $(x_0, y_0)$ are the coordinates of the center of the error ball and the error ball image, respectively. Here $\frac{-\pi}{2}$ is taken clockwise which is $\frac{3\pi}{2}$ counter clockwise. Please note a proper offset needs to be applied for accurate direction difference.

### *F. Path Constraint*

The different paths between the corrected points in error ball and its previous image are extremely important entity in determining the trajectory of a mobile user. This set of paths is essentially the domain of the possible paths. There are numerous possible paths to connect a projected location from the error ball image to the error ball. We need to assert a constraint to limit the number of paths between the corrected locations. The simplest, yet very pivotal constraint is the travel time. If the travel time of a path between the farthest corrected points is very large than the transition time of the ball, that path is very unlikely. This is very much related to the topology and geometry of the map. In most cases if the travel time of a path is more than twice the transition time of the error ball that path could be eliminated. On directional road, Manhattan style for example, the longest path is five times the direct distance assuming roads around the blocks make a square. The exact time multiplier very much depends on the map topology.

## IV. PROSPOSED METHOD FOR TRAJECTORY ESTIMATION

Location Points Collection: Suppose a mobile user starts from home at location **A,** travels to location **B, C,** and **D,**

and then comes home at location **A**. Figure 16 depicts logical parts of this travel trajectory. The points **B, C,** and **D** are the rest points. These rest points partition the trajectory into trajectory legs during the whole day of mobility. In this example there are four trajectory legs.

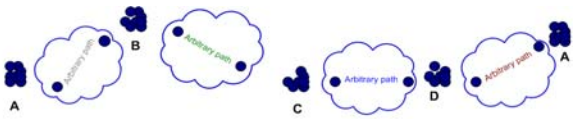



Figure 16. Logical partition of trajectory

We shall describe the mobility profiling method for one leg. For rest of the legs the steps are the same.

Step 1: Poll the mobile user for his location. Find the candidate geodesics as described in Section III B.

Step 2: Repeat Step 1 after determining polling time as described in Sections III C. Continue polling until destination of the leg is reached.

Step 3: Apply the condition for directional negativity to eliminate multiple geodesics in the error ball as explained in Section III E, please see Figure 17.

Step 4: Apply the path constrains using appropriate time multiplier as explained in Section III F.

Step 5: Map match the so obtained curve or set of curves with the digital map to obtain the closest to shape trajectory of the travel leg. Please see Figure 19.

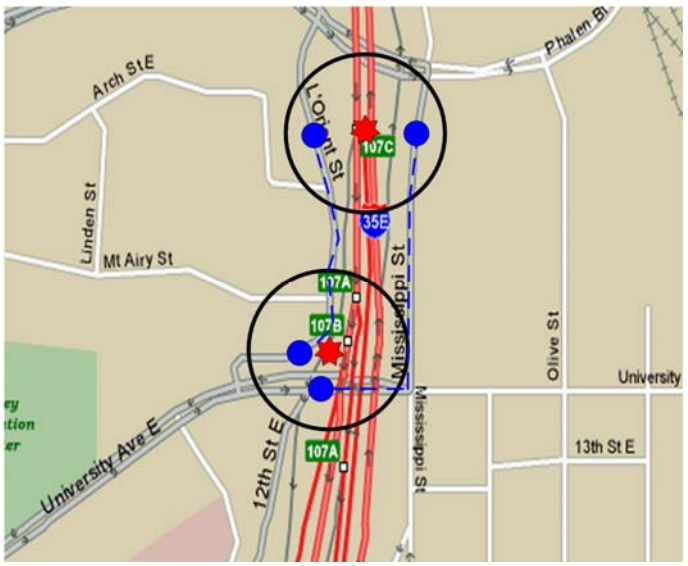



Figure 17. Direction negativity and path constrains on error ball and its images. If the user was on University Ave E prior to the recent last two polls shown in this picture, after applying the constraints, we are left with t only two candidates roads which are shown in broken blue lines.

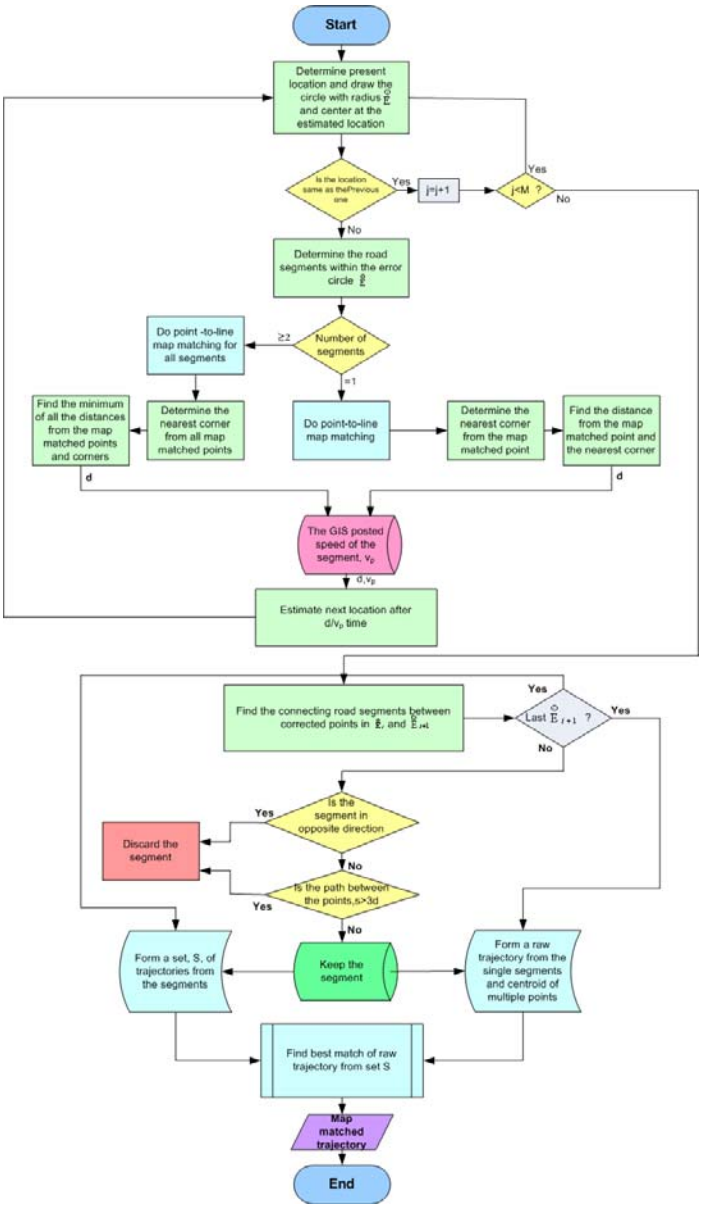



Figure 18. Trajectory estimation flow chart.

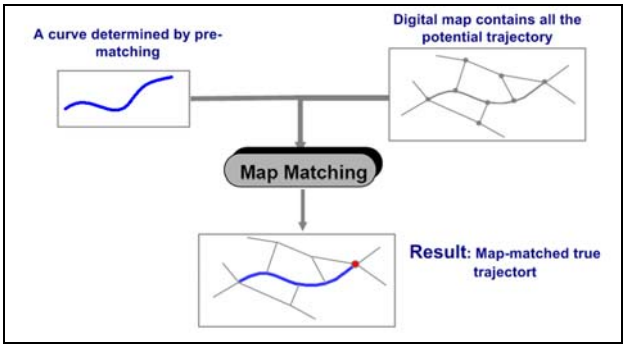



Figure 19. Map matching of pre-matched candidate curve with the set from GIS database.

A flow chart of the detailed algorithm is shown in Figure 18. The algorithm was implemented using C# (C sharp) in Microsoft windows. C# fulfilled the graphics requirements we needed for our simulation easily.

## V. IMPLEMENTATION OF PROPOSED METHOD

The implementation has two parts, as described in the following subsections.

### A. GIS Database of Road Networks

Our mobility profiling system is essentially a GIS embedded with a cellular system. The cellular part of the system estimates the mobile location; the GIS part corrects the estimated location and provides all necessary references for trajectory estimation. Database is the backbone of a GIS. For trajectory estimation we use a digital database of a road network. We have paper road maps, Figures 21 (scale 1:10,000) and 22 (scale 1:50,000), which need to be digitized to make a digital database. We use vector format data of a road network. Manual digitization is one of the widely used methods to digitize a paper map to vector data. It uses a digitizing table. A digitizing table is essentially an input device that interfaces with a computer, to store the coded data in the form of coordinate points. A digitizing table is an expensive tool. We developed a Microsoft Windows based application that emulates a digitizing table. The digitizing application, let us call it Digitizer from this point on, is not hard coded to exclusively digitize the maps shown in Figures 21, and 22 but it can load any map that needs to be digitized using the user interfaces depicted in Figure 20. The whole digitization process is a little labor intensive. We used point mode digitization .

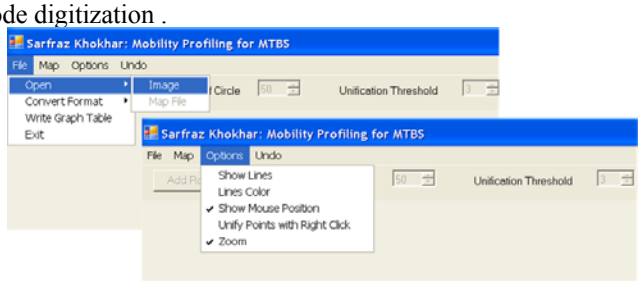



Figure 20. Digitizer's GUI.

The map image could be in JIFF, BMP, or JPEG format. Once the image is loaded the digitization starts by naming the road segment, i.e., polyline, and selecting the posted speed, and direction attribute of the road segment. The segments are digitized by mouse clicks at important points using point mode method. The most effective feature in the Digitizer is zooming, Figure. 21. This is where the Digitizer adds value in accuracy. The main source in digitization error is introduced by the operator who is doing the digitization [29][30]. One of the main reasons of operator's error in digitizing roads is that some roads are not thick enough on the map and it is difficult to stay on the center line of the road. The operator is supposed to stay within a band centered on the road center line. In cartography it is referred as epsilon band [31]. Zoom feature will help the operator to stay within the epsilon band and closer to the center line. The zoom window features a very fine pointer for further

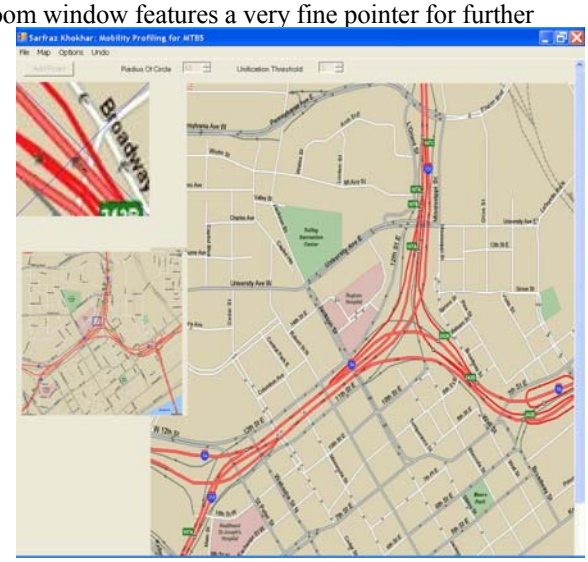



Figure 21. Zoom feature of the Digitizer helps in digitization by hand. The map being digitized is of an urban area of Saint Paul Minnesota.

accuracy. On thin or closely packed roads as shown in zoom window, the accuracy is about four times that that of a digitizing table. Since an operator concentrates on only one screen as compared to the digitizing table where the operator's attention is divided into two places, the monitor screen and the map, digitizing with software version of the digitizing table is more efficient. Once a map is digitized using a digitizing table, its digitized version is verified with zoom feature on the computer. So, zoom is a reliable feature in cartography world already. The process of manual digitization is a psychophysical process that depends on a harmony between human perception, adjustment of mental impression and physical balance of arm muscles [32], so, the accuracy of manual digitations varies from operator to operator. Manual digitization error on a 1:24,000 map ranges from 20-55 feet. Using a feature like zoom would reduce the error significantly.

The intersecting roads have a common point which must be selected as one of the points of the road segments during their digitization. One issue while marking this common point is that, one may be few pixels off while marking this point during the definition (digitization) of each road segment. This issue is remedied with one of the features of Digitizer; we called it "Unification Threshold". To combine points on different road segments into one intersecting point, we right click the mouse around the points, a unifying routine combines all the points in the threshold area. The threshold area is configurable as shown in Figure 20. After all roads are digitized and stored in the database, the ASCII database was converted to binary for faster access. We can convert ASCII data to binary and binary to ASCII. To verify the accuracy of the so obtained digitized database, we reconstructed the road network from the data accurately.

The mobility trajectory estimation simulation was done in two different topological scenarios: an urban area of St. Paul, Minnesota and suburban area of North Carolina. We digitized two maps, one shown in Figure 21 (urban) and the other one depicted in Figure 22 (suburban).

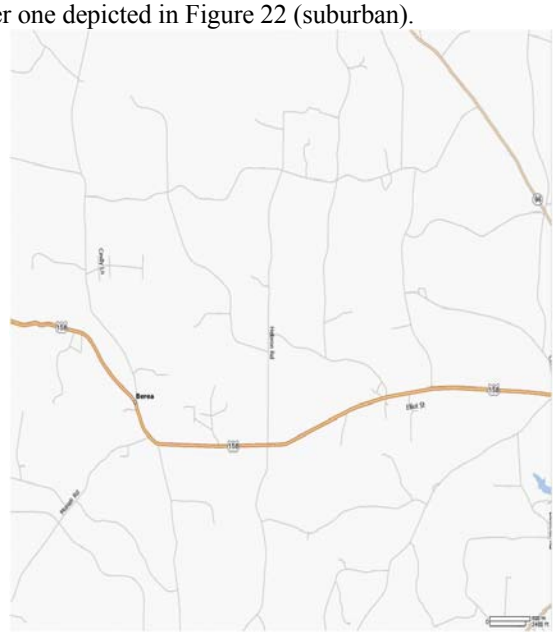


Figure 22. Map of suburban area digitized for GIS database.

The digitized version of the maps mentioned above are given below, Figures 23 and 24. Please note these maps are erected from the GIS data base as mentioned in Section II B.

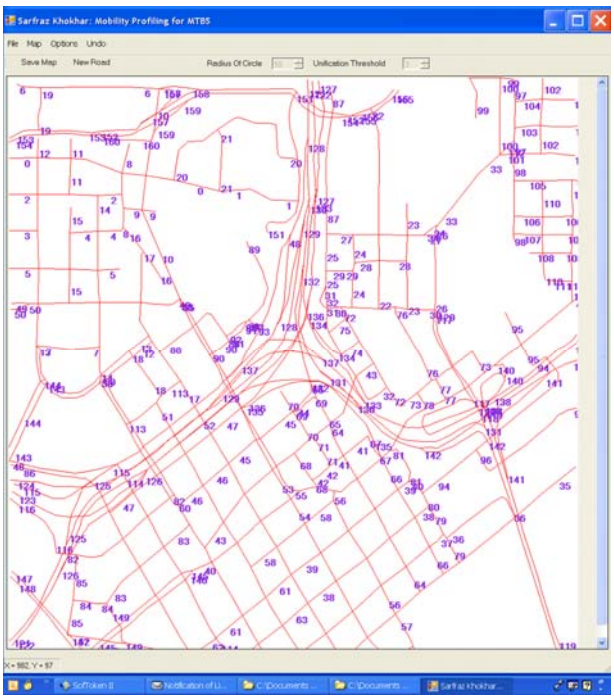



Figure 23. Reconstruction of map of Figure 21 from GIS database.

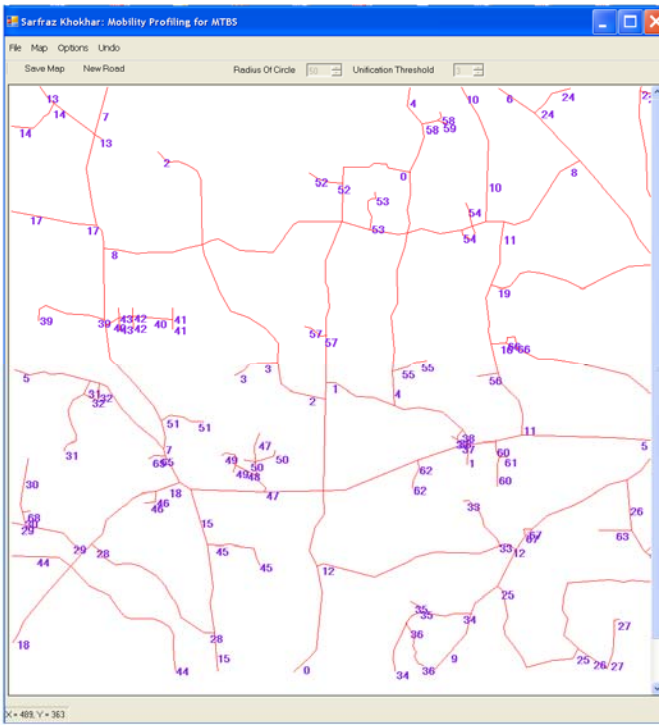



Figure 24. Reconstruction of map of Figure 22 from GIS database.

### B. *Mobility Profile Simulation*

To simulate, the mobile user's mobility, we developed another Windows based application, Mobility Profiler Simulator. The application has two major parts: one simulates mobility of the mobile user via traversing of a dinky car on the road network of a map; the other is an implementation of the mobility profiling method described in section IV. Figure 25 depicts the mobility simulation window of the application.

An optional step after invoking the application is to load the reference map image for viewing the simulation with respect to the map. Then digital map (GIS database) is loaded into the application. This is the actual road network which is referenced by the mobility profiler for location polling and map matching. The dinky car traverses on the path provided by the so called "path file". The path file is a text file containing the points and line segment information that defines a particular path for the mobile user. After path is loaded, the simulation is started from another window. The dinky car, shown in the circle in Figure 25, travels along the paths dictated in the path profile. This simulates the actual mobility of the mobile user. The Mobility Profiler Simulator estimates the mobile trajectory independently by polling mobile user's locations, correcting them by identifying the road segments and them combining the road segments.

The animation speed of the dinky car can be increased or decreased for demonstration purposes using the control option called Animation Speed. The dinky car travels on the

Figure 25. Mobility Profiling simulator's GUI, the mobility simulation window.

road segments according to the posted speed on them. A variation on the speed can be adjusted. The polling agent in mobility profiler polls for location of the car according to an implementation of Intersection Polling method, described in details in Section III C. The location is estimated with a random error. The process of location polling has been animated in the application to depict how it would happen in the actual deployment in the field.. In this process the polling agent inquires the exact location of the dinky car, the location is returned to the profiler with a radiating signal (animation) from the car, adding a random error. The error has been modeled with a circle of radius equal to the error associated with the underlying location estimation technology and centered at the returned estimated location.

The second window of the application, the mobility profiler, is invoked from a menu from the first application. This part of the simulator provides the interface for input variables; the location estimation error of the underlying location estimation technology and the posted speed variations. We can define lower and upper bounds for the speed variation in the posted speed of GIS database. These input variables are entered in the counters shown in red rectangles in the Figure 26. This window plots the trajectory per algorithm given in flow chart in Figure 18. During the first parsing of the simulation candidate road segments and the corresponding location points are determined. In the second parsing, direction and the distance constraints are applied. After these constraints are applied, the estimated trajectory is plotted as show in Figure 26.

If this trajectory does not match exactly with a trajectory from the digital map, we map match it with our digital map. In most of the cases the constraints rectify the shape of the trajectory. The actual path (from path file) and the estimated trajectory, both, are plotted in the simulator window. The reference map, which merely acts as background image makes the simulation result more meaningful for viewing, Figure 27. This gives a better understanding of the error in


Figure 26. Mobility profiling simulator's GUI, the trajectory estimation window.

the estimated trajectory. It shows the exact locations on the map where the estimated trajectory deviates from the actual one. The option Offset XY translates the reference trajectory right and down according to the set values in these counters for easy viewing and comparing of the two trajectories. Please refer to the first rectangle in the menu bar in Figure 26.

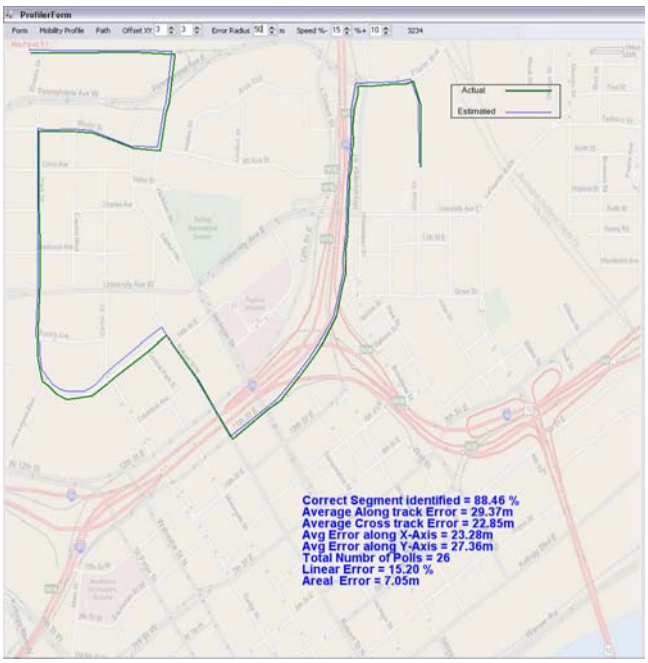



Figure 27. Mobility profiling simulator window with reference map.

All the simulation results are presented with reference path and reference map as shown in Figure 27. The green curve (darker if black and white) depicts the actual path whereas the blue curve (lighter if black and white) is the estimated trajectory. We ran the simulation, both in urban and suburban areas. The main variables in the simulation are the location error and the speed variation, which we could vary during the simulation also, if needed. After a simulation finishes, the mobility profiler prints important statistics pertaining to the simulation. It prints the following statistics:

- Percentage of segments identified correctly
- Average Along Track Error
- Average Cross Track Error
- Average Error along X-axis
- Average Error along Y-axis
- Total number of location polls
- Linear Error ( defined in next section)
- Areal Error (defined in next section)

## VI. TRAJECTORY ESTIMATION SIMULATION

Location points are building blocks of a trajectory. The estimated location of a mobile user does not necessarily map on a road because of the error in location estimation methods. Among the vertices defining roads, the points corresponding to roads intersections play a pivotal role. In urban area the roads are densely populated hence a large number of road intersections. This results in potentially high number of turns that leads to high polling frequency. Secondly, because of high road density, an error area (estimated location) encloses a large number of roads. On the other hand in rural or suburban areas the road density is smaller as compared to urban areas; therefore estimated location corresponds to fewer roads as candidates of the mobile user trajectory. So, the rural areas should provide more accurate user's trajectory. In this chapter we investigate both scenarios and provide the simulation results. We shall demonstrate how estimated location error affects the accuracy of the mobile user's trajectory. We shall correlate the error with the underlying location estimation technologies and present suitability of the location technology under different road topologies.

Our estimated trajectory is represented as a sequence of points, so essentially as a set. The error associated with the estimated trajectory is a measure of dissimilarity between the actual and estimated trajectories. The actual trajectory is also a sequence of points, so Hausdorff distance [33] would be natural choice as an error index. However the problem with Hausdorff distance is that, it gives the shortest distance between two points in the sets and it does not cover the course of the distance traveled. Therefore using Hausdorff distance to measure the dissimilarity would lead to wrong findings. The issue with Hausdorff distance is covered in Fréchet distance [34]; it takes the course of distance into account. It is explained by man-dog example in literature [35][36]. Fréchet distance is widely used to find distance (dissimilarity) between two curves. The curves are approximated as polygonal figures as done in the case of roads. However Fréchet distance shows an anomaly when comparing the two trajectories as depicted in Figure 28. The dissimilarity between two paths A and B and that of between A and C, according to Fréchet distance is the same, which is misleading.

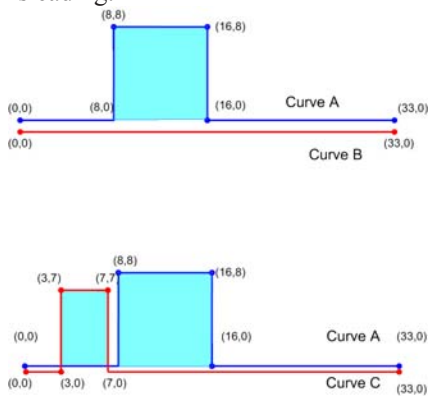



Figure 28. Anomaly in Fréchet distance.

The Fréchet distance between A and B is 8, which is also the case with A and C. Clearly A and C are more dissimilar than A and B. Therefore we need to define a meaningful index of error for an estimated trajectory. We called this error Trajectory Estimation Error (TEE). TEE can be represented as linear error or areal error as defined below. Both, linear and areal errors together, give a good understanding of the trajectory estimation error.

### A. Linear Error

Linear error demonstrates the length of the actual trajectory that is missing in the estimated trajectory. It is ratio of the path length that is erroneous, $\lambda_e$ , to the total path length $\lambda_t$ .

$$E_L = \frac{\lambda_e}{\lambda_t}. \quad (21)$$

### B. Areal Error

Areal error demonstrates the dissimilarity between the two trajectories (curve), the actual and the estimated one by the area bounded by them as shown in Figure 28. According to Fréchet distance the two set of curves, A,B and A,C are equally dissimilar, however their areal error portrays a better error index. For better comparison of areal error, bounded area per unit length of trajectory would be a better index of error. The areal error per unit length is:

$$E_A = \frac{\alpha}{L}. \quad (22)$$

Where, $\alpha$ is the error area, and $L$ is the total length of the actual trajectory. We have used linear error along with areal error to get a better degree of comparison between actual and the estimated trajectories.

### C. Trjectory Estimation Simulation Results

Trajectory estimation simulation was performed using two digital maps discussed above, one for urban area of Saint Paul Minnesota and the other for suburban area of North Carolina. A range of error corresponding to different location estimation technologies was used for the simulation. Figures 29 through 38 show the simulation results for an urban area for different location error. Green trajectory is the actual trajectory whereas the blue is the estimated one.


Figure 29. Result with location error of 25m.


Figure 30. Result with location error of 50m.

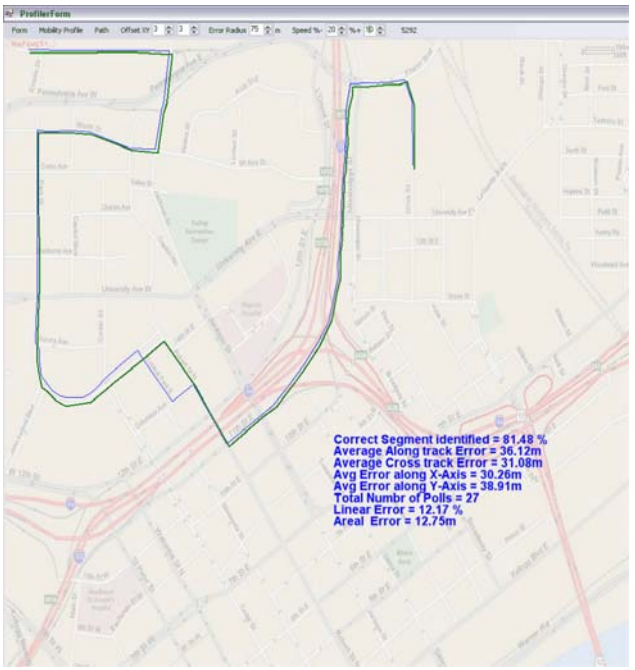



Figure 31. Result with location error of 75m.

In urban area for location error of 50m or less, the trajectory error was on the closely parallel roads only (Figures 29 and 30). However, for larger location error, 100m and more, the trajectory error expanded farther than the closely parallel roads (Figures 32 through 37). Figure 38 shows a graph for location error versus trajectory error for the urban area.

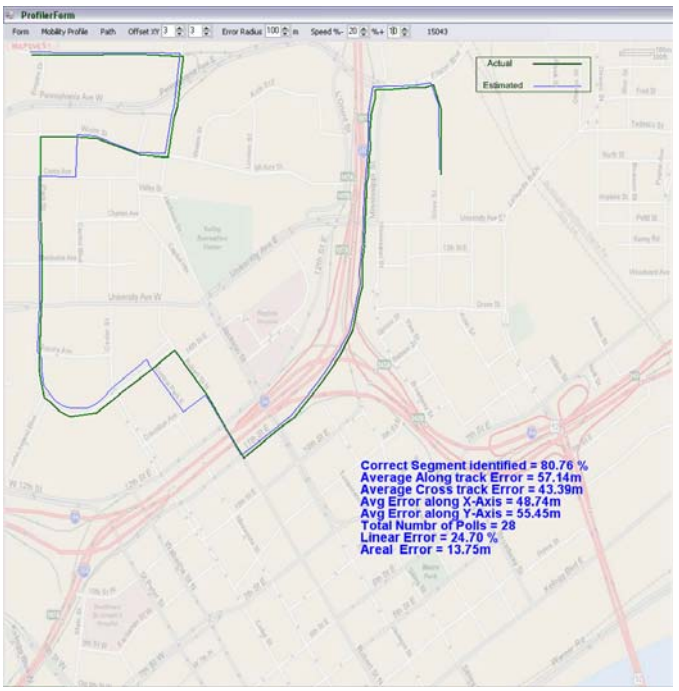



Figure 32. Result with location error of 100m.

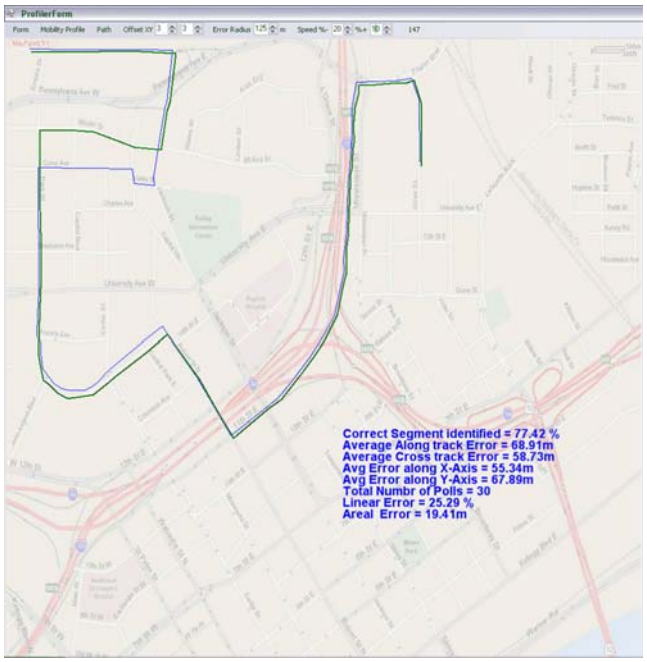



Figure 33. Result with location error of 125m.

For the suburban area the algorithm missed only a short segment of trajectory (Figures 40 through 42), even for large location error of 400m. We get this performance because of low number of intersections per unit area in suburban areas.

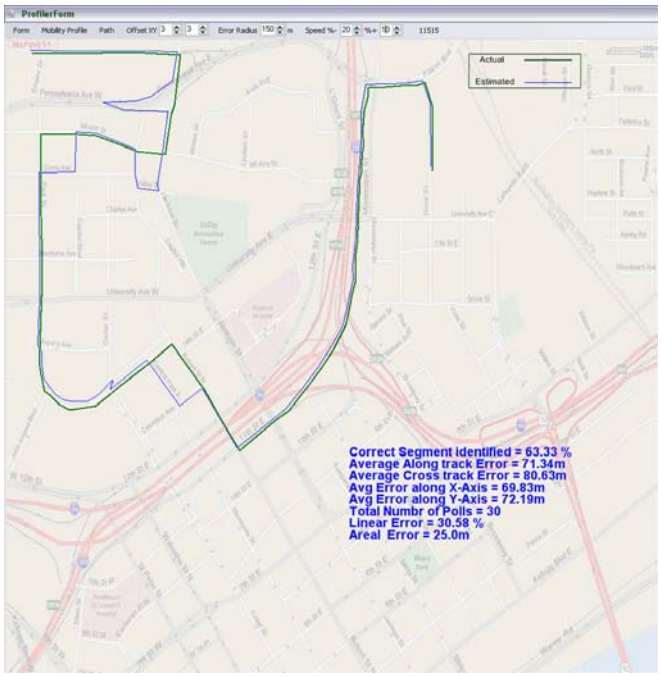



Figure 34. Result with location error of 150m.

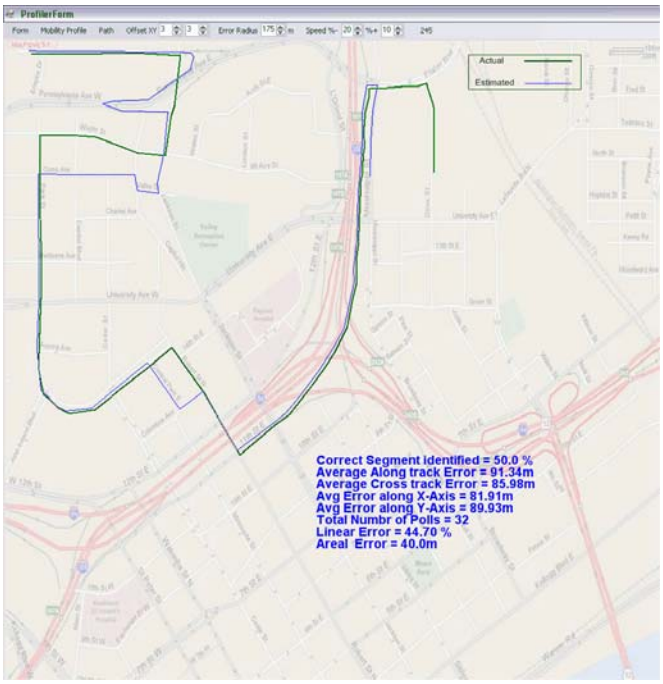



Figure 35. Result with location error of 175m.

Both areal and linear errors in trajectory estimation not only stem from location estimation error, but are intrinsic to the map topology and path. For example, if we choose a different path the same parameters used in Figure 36 would yield a different result, please see Figure 37.

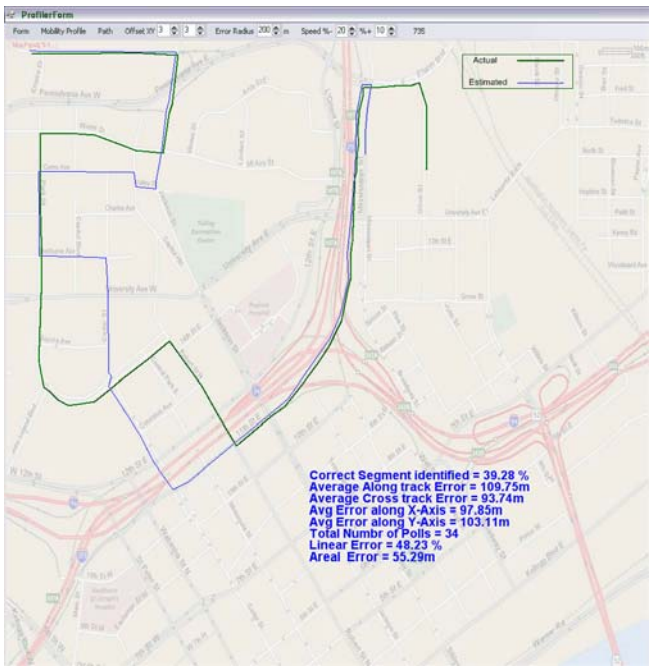



Figure 36. Result with location error of 200m.

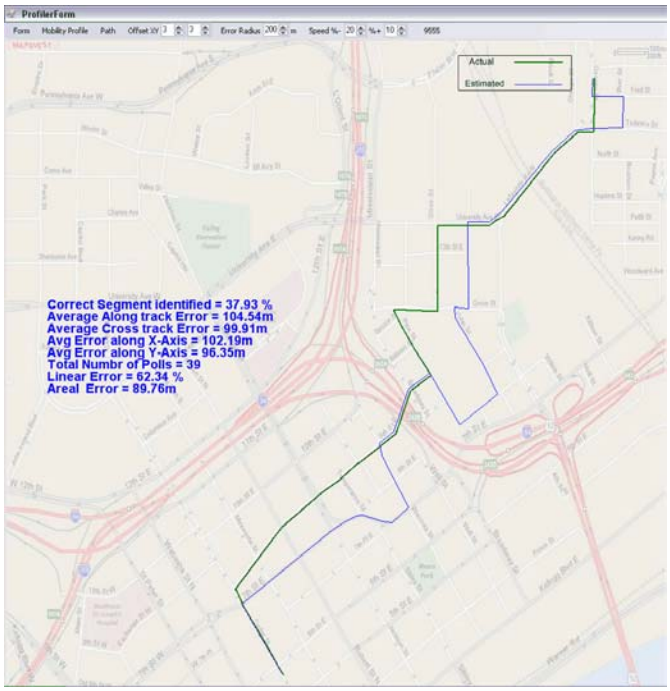



Figure 37. En error of 200 m yields higher trajectory error using dense part of the map.

The relationship of Trajectory Estimation Error (TEE) to the location estimation error (LEE) can be easily interpreted from the results above. Figure 38 depicts the LEE-TEE relationship for LEE of 0-200m with a step of 25m. For urban area, after 200m of LEE, the trajectories are not reliable for profiling. TEEs depicted in the graph are the typical values for ten simulations for each LEE. For smaller location estimation errors, 25m and 50m, each simulation run for that particular LEE gave almost similar trajectory. The error mainly comes from the two parallel roads about thirty meters apart. For LEE of 75m and 100m, there was some difference in the shape of trajectory for each run. After around 125m of LEE the trajectory curve (shape) varies significantly for each run with the same LEE, because the error circle was large enough to start inscribing multiple candidate road segments and road intersections. The three rectangles in Figure 38 provide a reference of known accuracy of different location estimation technologies, namely Assisted GPS, TDOA technologies such as E-OTD and U-TDOA (Uplink-TDOA), and AOA. It is evident from the figure which location estimation technologies suit for mobile trajectory estimation. It is important to note that dependence of TEE on LEE is very much tied to the map topology and the trajectory of the mobile user. Within the same map two trajectories of the same length under same LEE would lead to different set of linear and areal error as observed in comparing Figures 36 and 37. Since our location error is random with the upper bound of $\odot{E}$, with each run of LEE for specific $\odot{E}$, we get different values of TEE. This is true for both, linear and areal errors. With large number of simulation runs we get a range of TEE related to LEE. As LEE increases the range widens as expected. Please see Figure 39.

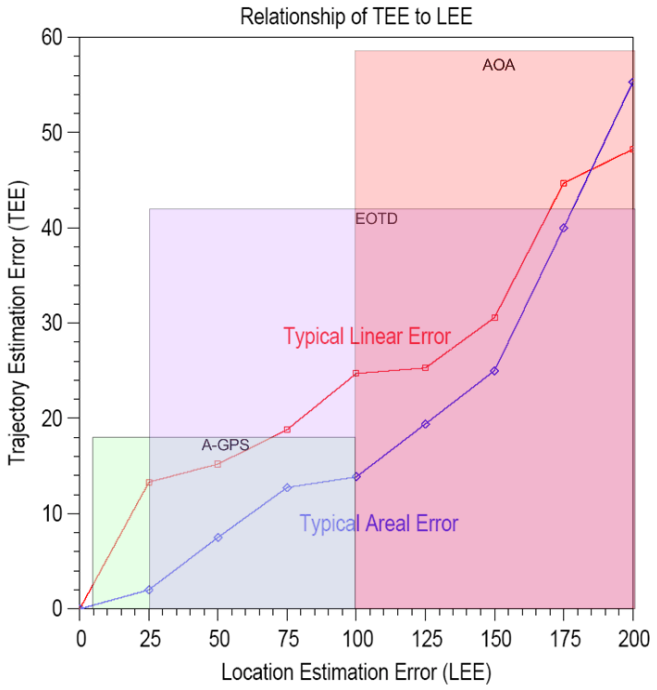



Figure 38. Trajectory estimation error vs location estimation error related to different location estimation techniques.

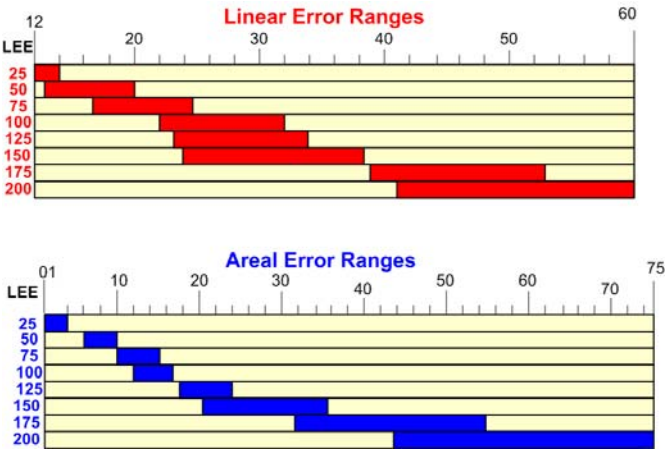



Figure 39. Linear and Areal Error Ranges.

Figures 40 through 42 show simulation results for rural area shown in map in Figure 22 for error of 100m, 200m and 400m respectively. The results are the same for these location error. The mobility profiler missed a small corner near the main highway. For smaller location error around 50 m, this minor error in trajectory estimation dimishes. This

behavior is manifested because there are two parallel path that are about 100 m apart and the both of them are candidate road segment. The segment that is picked in estimation is more closer in shape of the curve prescribed by the estimation location points.

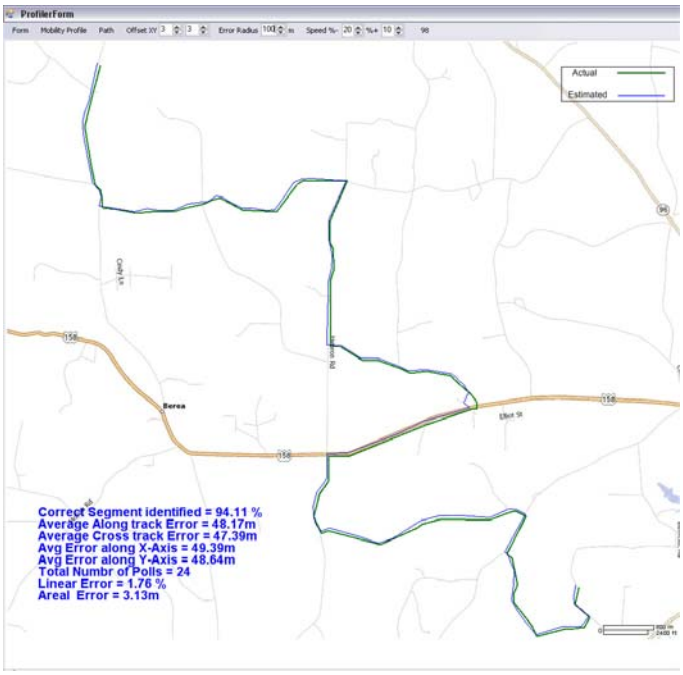



Figure 40. Suburban Trajectory estimation with 100m error.

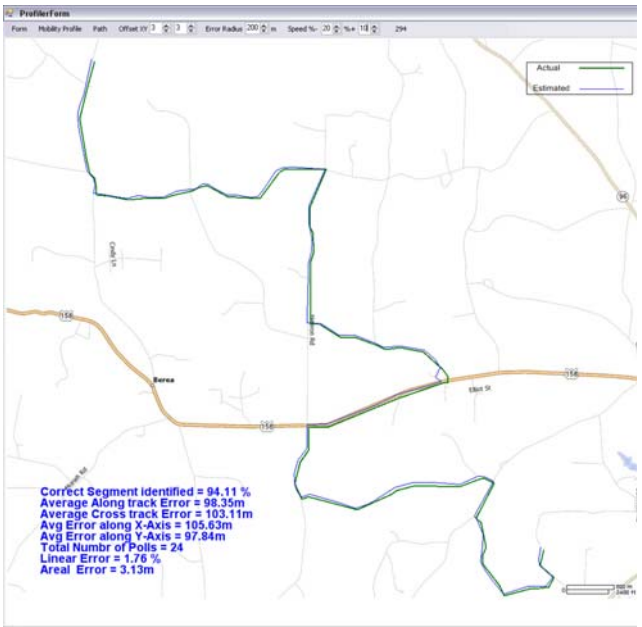



Figure 41. Result with location error of 200m.

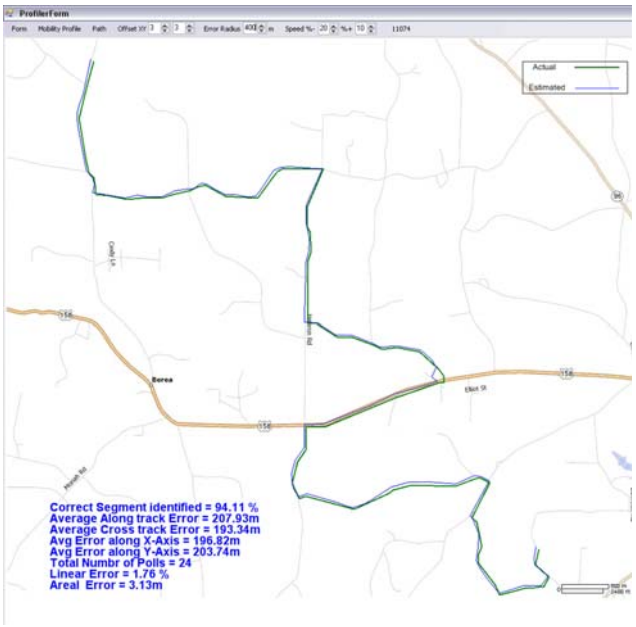



Figure 42. Suburban Trajectory estimation with 400 m error.

## VII. CONCLUSION

We applied the proposed trajectory estimation algorithm in two field scenarios; urban area and rural area. The objective was to determine how existing location estimation technologies would perform with the algorithm in urban and suburban areas. In urban area, where road network is dense and consequently there are a large number of potential turns for a mobile user, the trajectory error increases sharply with the error of underlying location estimation technology. Even within the urban area, the topological and geographical differences between different parts of the same road network bring about substantial difference in the accuracy of the estimated trajectory as seen in comparing Figures 36 and 37. The right half of the road network in Figures 36 has fewer turns. There are fewer inlets and outlet points per unit area on that side of the network. That's why; increasing the location estimation error did not affect that part of the road network as much as it did on the other side. When same location error was applied to a different path going through higher number of potential turns (Figure 37) the error increased to almost double. With the increase of LEE not only TEE increases, but also the number of location polls required estimating the mobile trajectory. Even under topological and geographical constraints, with larger error, it requires a larger number of polls to get out of the cluster of roads. These roads are enclosed within the error circle. Larger the error is, more candidate road segments for trajectory estimation are enclosed in the error circle. So, in urban area relatively precise location estimation methods like A-GPS or accurate time based methods such as EOTD are best suited for trajectory estimation. Whereas in suburban or

rural area, almost all technologies that fulfill E-911 mandate [3] qualify to estimate trajectory with good accuracy.

## REFERENCES

[1] Sarfraz Khokhar, and Arne A. Nilsson, "Estimation of mobile trajectory in a wireless network: A basis for user's mobility profiling for mobile trajectoy based services," 3rd International Conference on Sensor Technologies and Applications, Athens, 2009.

[2] Elliott D. Kaplan, and Christopher Hegarty, "Understanding GPS: Principles and applications," Artech House Publishers, 2005, Second Edition.

[3] Allison Kealy, Stephan Winter, and Günther Retscher, "Intelligent location models for next generation location-based services," Journal of Location Based Services, vol. 1, Issue 4, December 2007, pp. 237–255.

[4] Lin Ding-Bing , Juan G Rong-Terng, and Lin Hsin-Piao, "Mobile location estimation and tracking for GSM systems," Fifteenth IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications , Barcelona , 2004 , pp. 2835–2839.

[5] Sarfraz Khokhar, Arne. A. Nilsson, and Mo-Yuen Chow, "Mobile location estimation methods," GIS 2001 Conference & Exposition, Vancouver, BC, February, 2001.

[6] Sarfraz khokhar, Arne A. Nilsson, "Introduction to mobile trajectory based services: A new direction in mobile location based services," International Conference on Wireless Algorithms, Systems and Applications, Boston 2009, Springer-Verlag Berlin Heidelberg, pp. 398–407.

[7] Stanley Sokolowski, Raymond L. Filler, Nhut Vo, and Paul Olson**,** "Enhanced C2 functionality in urban and other complex terrain," IEEE Military Communications Conference, vol. 1, 17-20 October, 2005, pp. 258–264.

[8] Goran M. Djuknic, and Robert E. Richton, "Geolocation and assisted GPS," Computer, Volume 34, Issue 2, February, 2001, pp. 123–125.

[9] Wenjie Liao, Weifeng Lv, Tongyu Zhu, and Dong Dong Wu, "A map matching algorithm for intersections based on Floating Car Data,**"** 10th International Conference on Advanced Communication Technology, vol. 1, 17-20 February, 2008, pp. 311–316.

[10] David Bernstein, and Alain Kornhauser, "An introduction to map matching for personal navigation assistants, " New Jersey TIDE Center 1996.

[11] Chenhao Wang, Zhencheng Hu, and Keiichi Uchimura, "A Precise road network modeling and map matching for vehicle navigation,**"** 11th International IEEE Conference on Intelligent Transportation Systems, 12-15 October, 2008, pp. 1084–1089.

[12] Mohammed A. Quddus, "High integrity map matching algorithms for advanced transport telematics applications," Centre for Transport Studies, Imperial College London, United Kingdom, January, 2006.

[13] Hong-mei Yin, and Shan-wu Su, "Modeling for geospatial database of national fundamental geographic information**,"** IEEE International Conference on Geoscience and Remote Sensing, 2006, pp. 865–868.

[14] Stephen Wise, "GIS Basics," Taylor & Francis, April, 2007.

[15] Dean Pomerleau, "A neural network based autonomous navigation," Vision and Navigation, Kluwer Academic Publishers, 1990.

[16] Surender K. Kenue, "LAMELOK:Detection of lane boundaries and vehicle tracking using image-processing techniques," Parts I and II, SPIE Mobile Robots IV, 1989.

[17] E.D. Dickmanns, R. Behringer, D. Dickmanns, T. Hildebrandt, M. Maurer, F. Thomanek, and J. Schiehlen, "The seeing passenger car 'VaMoRs-P'," Proceedings of the Intelligent Vehicle Symposium, 1994, pp. 68 –73.

[18] Karl Kluge, and Charles Thorpe, "Representation and recovery of road geometry in YARF," Proceedings of the Intelligent Vehicles Symposium, 1994, pp. 1141–19.

[19] Mathew A. Turk, David G. Morgenthaler, Keith D. Gremban, and Martin Marra, "VITS: A vision system for autonomous land vehicle navigation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, Issue 3, May, 1988, pp. 342 3–61.

[20] Satish Shirali, and Harkrishan L.Vasudeva, "Metric spaces," Springer-Verlog London Limited, 2006.

[21] Sulata Mitra, and Sipra DasBit, "On location tracking and load balancing in cellular mobile environment- A probabilistic approach," 5th International Conference on Electrical and Computer Engineering , December, 2008, Dhaka, Bangladesh.

[22] Washington. Y. Ochieng, Mohammad A. Quddus, and R.B. Noland, "Map-matching in complex urban road networks," RBC - Revista Brasileira de Cartografia, N0 55/2, dezembro, 2003.

[23] Khalid Touil, Mourad Zribi, and Mohammed Benjelloun, "Application of transferable belief model to navigation system, Integrated Computer-Aided Engineering," IOS Press, 2007, pp. 93–105.

[24] Mohammad A. Quddus, Washington. Y. Ochieng, L. Zhao, and Robert .B. Noland, "A general map matching algorithm for transport telematics applications", GPS Solutions, 2003, pp.157–167.

[25] Yu Meng, Zhilin Li, Wu Chen, and Yongqi Chen, "Reliability and integrity issues for vehicle positioning systems," The International Symposium on GNSS/GPS, Sydney, Australia, December, 2004.

[26] Sudarshan S. Chawathe, **"**Segment-based map matching," Intelligent Vehicles Symposium, June, 2007, pp. 1190–1197.

[27] Pen -Shan Hung , Tsui -Chuan Su, "Map-matching algorithm of gps vehicle navigation system," Geographic Information System Research Center, Reng Chia University, Taichung, Taiwan, 1998.

[28] Mohammed A. Quddus , Washington Y. Ochieng, and Robert B. Noland, "Current map-matching algorithms for transportation applications: State-of-the art and future research directions," Transportation Research Part C, vol. 15, 2007, pp. 312–328.

[29] Paul V. Bolstad, and James L. Smith, "Errors in GIS assessing spatial data accuracy," Journal of Forestry, November, 1992, pp. 21–29.

[30] Paul V. Bolstand, Paul Gessler, and Thomas M. Lillesand, "Positional uncertainty in manually digitized map data," International Journal of Geographical Information Systems, vol. 4, 1990, pp.399–412.

[31] R. Dunn, A. R. Harrison, and J. C. White, "Positional accuracy and measurement in digital database of land use: an empirical study," International Journal of Geographical Information Systems, vol. 4, 1990, pp. 385–398.

[32] M. M. Yagoub, "The effect of psychophysics on the I of GIS/LIS," Geoinformatics and Socioinformatics, The Proceedings of Geoinformatics Conference, Ann Arbor, 19-21 June, 1999, pp. 1–9.

[33] Helmut Alt, and Ludmila Scharf, "Computing the Hausdorff distance between curved objects," The 20th European Workshop on Computational Geometry, 2004, pp. 233–236.

[34] Günter Rote, "Computing the Fréchet distance between piecewise smooth curves," The 20th European Workshop on Computational Geometry, 2004, pp. 147–150.

[35] R. Sriraghavendra, K. Karthik, and Chiranjib Bhattacharyya, "Fréchet distance based approach for searching online handwritten documents", Ninth International Conference on Document Analysis and Recognition , vol. 1, 23-26 September, 2007, pp. 461–465.

[36] Hui Ding, G. Trajcevski, and P. Scheuermann, "Efficient similarity join of large sets of moving object trajectories," 15th International Symposium on Temporal Representation and Reasoning, 16-18 June, 2008, pp. 79–87.

# Towards an Optimal Positioning of Multiple Mobile Sinks in WSNs for Buildings

Leila Ben Saad, Bernard Tourancheau
*LIP UMR 5668 of CNRS-ENS Lyon-INRIA-UCB Lyon*
*Lyon, France*
{*Leila.Ben.Saad, Bernard.Tourancheau*}*@ens-lyon.fr*

## Abstract

*The need for wireless sensor networks is rapidly growing in a wide range of applications specially for buildings automation. In such networks, a large number of sensors with limited energy supply are in charge of relaying the sensed data hop by hop to the nearest sink. The sensors closest to the sinks deplete their energy much faster than distant nodes because they carry heavy traffic which causes prematurely the end of the network lifetime. Employing mobile sinks can alleviate this problem by distributing the high traffic load among the sensors and increase the network lifetime. In this work, we aim to find the best way to relocate sinks inside buildings by determining their optimal locations and the duration of their sojourn time. Therefore, we propose an Integer Linear Program for multiple mobile sinks which directly maximizes the network lifetime instead of minimizing the energy consumption or maximizing the residual energy, which is what was done in previous solutions. We evaluated the performance of our approach by simulation and compared it with others schemes. The results show that our solution extends significantly the network lifetime and balances notably the energy consumption among the nodes.*

## Index Terms

*Wireless Sensor Networks, Sinks positioning, Mobile sinks, Network lifetime, Integer Linear Programming.*

## 1. Introduction

Recent years have witnessed an increasing need for wireless sensor networks (WSNs) in a wide range of applications specially for buildings automation. In fact, the WSNs can be used as a way to reduce the waste of energy inside buildings by reporting essential information from the indoor environment allowing, for instance, to turn off the unnecessary electric appliances in the rooms. Nevertheless, wireless sensor networks deployment inside buildings is a very challenging problem[1]. Such networks are composed of low cost tiny devices with sensing, data processing and communication capabilities. These sensors have a short operational life because they are equipped with a limited number of batteries supplying energy. Moreover, it is usually impractical and even impossible to replace or recharge them. The sensors, which are densely deployed in the area of interest, measure and monitor their indoor environment (temperature, humidity, light, sound, etc.,) and collaborate to forward these measurements towards the nearest resource-rich collector, referred to as the sink node. The sensor nodes which are far away from the sink use multi-hops communication. This mean of communication makes the sensors near the sinks deplete their energy much faster than distant nodes because they carry the packets of sensors located farther away in addition to their own packets. Therefore, what is known as a hole appears around the sinks and makes distant nodes unreachable and unable to send their data. Consequently, the network lifetime ends prematurely.

More and more efforts have been done recently to improve the lifetime of WSNs. Many communication protocols have been proposed including among others topology control[2][3], routing[4][5] and clustering[6]. However, further improvement can be achieved if we relocate the sinks in order to change over time the nodes located close to them. Thus, this can solve the energy hole problem and guarantee balanced energy consumption among the nodes.

In this work, our purpose is to determine where to place multiple sinks inside buildings, how long they have to stay in certain locations and where to move them to extend optimally the network lifetime. To answer these questions, we propose an Integer Linear Program (ILP) for multiple mobile sinks whose objective function directly maximizes the network lifetime instead of minimizing the energy consumption or maximizing the residual energy, which is what was done in previous solutions[7][8].

The contribution of our work concerns not only the definition of an ILP which determines the optimal locations of multiple mobile sinks but also shows that relocating mobile sinks inside a whole network is more efficient than relocating mobile sinks inside different clusters. Simulation results show that with our proposed solution, the network lifetime is extended and the energy consumption is more balanced among the nodes. Moreover, the lifetime improvement that can be achieved when relocating 3 sinks in a network with hundred sensors is almost 230 % in our experiments. Such results can provide useful guidelines for real wireless sensor network deployment.

The rest of this paper is organized as follows. In Section 2, we review the previously proposed approaches to solve the energy hole problem and extend the network lifetime in WSNs. Section 3 describes the system model including the major assumptions. Section 4 presents the formulation of our proposed ILP for multiple mobile sinks. Section 5 evaluates the performance of the proposed solution and presents the experimental results. Section 6 concludes the paper.

## 2. Related Work

In order to improve the network lifetime of WSNs, many researchers looked for approaches that help to solve the energy hole problem. Several solutions proposed to place more sensor nodes around the sink[9][10][11]. This solution is called nonuniform node distribution and consists in adding nodes to the areas with heavier traffic in order to create different node densities. However, these solutions are not always feasible in practice and result in unbalanced sensing coverage over different regions of the network.

Another way of optimizing the network lifetime is to use multiple sinks instead of one in order to decrease the average of packets that has to pass through the nodes close to the sinks. The location of these sinks has a great influence on the network lifetime. For this reason, many works focused on the optimal placement of multiple static sinks in WSNs. The majority of the sinks placement problem formulations are NP-complete depending on the assumptions and network model. Therefore to reduce this complexity, approximation algorithms were used[12] and heuristics were adopted to reduce the energy dissipation at each node[13]. In [14], the problem was formulated by a linear programming model to find the optimal positions of static sinks and the optimal traffic flow rate of routing paths in WSNs.

Most of the optimal multi-sinks positioning approaches described previously contribute to increase the network lifetime. Nevertheless, it was proved in [15] that using a mobile sink is more efficient than a static one and achieves further improvements in network lifetime by distributing the load of the nodes close to the sink. Furthermore, the sink mobility has many other advantages. In fact, it can improve the connectivity of the isolated sensors and may guarantee the sink security in case of malicious attacks.

The majority of related works studied the mobility of a single sink[15][16][17][18][19][20]. But, very few researches focused on the mobility of multiple sinks. In [17], the solution of repositioning a single sink is extended to a network with several sinks by dividing it into several clusters. Each sink can only move in its cluster.

There are basically three categories of sink mobility. Mobile sink may move in a fixed path[15], may take a random path[21] or may move in optimal locations in terms of network lifetime and energy consumption.

The authors of the paper [15] suggested that the sink moves on the periphery of the network to gather the data of the sensors deployed within a circle. The authors of [21] proposed to use ”Data Mules” which move randomly on the sensor field and collect data from the nodes.

An often used way to determine the locations of mobile sinks in the third category is to develop an algorithm. Some proposed algorithms make a moving decision of sinks according to the complete knowledge of the energy distribution of the sensors. In [22], the sinks move towards the nodes that have the highest residual energy. But, this strategy requires that the sensors send periodically to the sink additional information about their energy level to allow the sink to found out the nodes which have the highest energy. By doing so, a lot of energy will be wasted.

Some others algorithms find the locations of mobile sinks by solving a mathematical model[23][24]. In [23], the algorithm minimizes the average distances between sensors and nearest sinks. In [24], the algorithm selects the locations of sinks in the periphery of the network in such way that the difference between the maximum and the minimum residual energy of nodes is minimized.

To find the optimal locations of mobile sinks, some proposals formulated the problem as an Integer Linear Program ILP. In [8], the proposed ILP maximizes the minimum residual energy over all nodes. In [7], the proposed ILP minimizes the energy consumed at each node.

Most of proposed approaches to determine the locations of multiple mobile sinks in WSNs are centered on energy minimization. In our work, a different formulation of the problem is proposed, where the ILP proposed directly maximizes the network lifetime instead of minimizing the energy consumption or maximizing the residual energy. To us, this is closer to the need of sensors deployment in building monitoring for instance.

## 3. System Model

In order to deploy sensors and sinks inside buildings, we made the following assumptions for the system model.

### 3.1. Network Model

- All sensors are statically placed in a bi-dimensional grid of same size cells constructed from the building plan as shown in Figure 1.
- All sensors have a limited initial energy supply and a fixed transmission range equal to the distance between two nodes (i.e, cell size).
- Each sensor regularly generates the same amount of data.
- The number of sinks is fixed and known beforehand.
- The sinks can be located only in feasible sites where they are connected to power supply and the Internet.

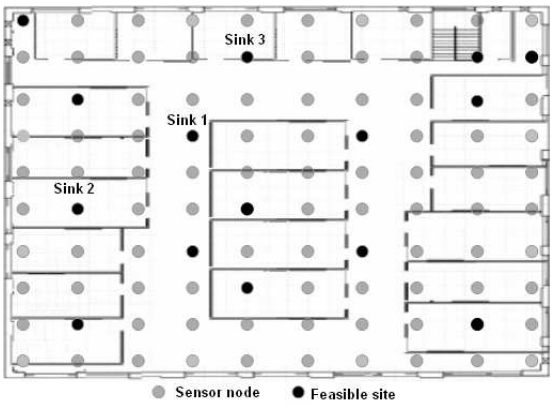



Figure 1. 10x10 Grid of cells with 100 sensors

- The sinks keep moving in the grid from one feasible site to another one until the network lifetime end.
- The network lifetime is defined as the time until the first sensor dies (i.e, it uses up its residual energy).
- The sinks should stay at a feasible site for at least a certain duration of time. At the end of this duration, they may stay or change of location.
- The traveling time of sinks between feasible sites is considered negligible for analytical simplicity.
- The sensor nodes which are not co-located with any sinks inside the grid, relay their generated data via multiple hops to reach the nearest sink.
- An ideal MAC layer with no collisions and retransmissions is assumed.
- In our assumptions, only the energy consumption for communication is considered due to the fact that communication is the dominant power consumer in a sensor node.

### 3.2. Routing and Path Selection

The sensor nodes which are not co-located with any sinks inside the grid send their generated data hop by hop to the nearest sink. When a sensor node is located in the same horizontal or vertical line of the nearest sink position, there is only one shortest path between the two nodes. Otherwise, there are multiple shortest paths. In our routing protocol like in [18], we route "per dimension". We consider only the two paths along the perimeter of the rectangle, i.e., paths 1 and 2 in Figure 2. These two routes are considered equivalent.

### 3.3. Power Consumption

To calculate the power consumption, we consider the same realistic model as in [25]. Therefore, the power expended to transmit a $L_1$-bit/s to a distance $d$ is:

$$P_{T_x} = L_1\gamma_1 + L_1\gamma_2 d^{\beta} \quad (1)$$

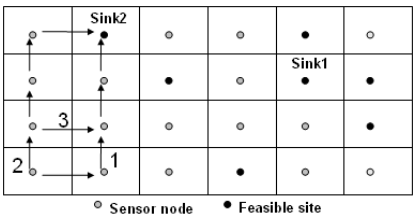



Figure 2. Path selection

where $\gamma_1$ is the energy consumption factor indicating the power consumed per bit by the sensor to activate transceiver circuitry, $\gamma_2$ is the energy consumption factor indicating the power consumed per bit by the transmit amplifier to achieve an acceptable energy per bit over noise spectral density and $\beta$ is the path loss exponent. The power expended to receive $L_2$-bit/s in the same radio model is:

$$P_{R_x} = L_2\alpha \quad (2)$$

where $\alpha$ is the energy consumption factor indicating the power consumed per bit at receiver circuit. Thus, the total energy consumed by a sensor node per time unit is:

$$P_{total} = P_{T_x} + P_{R_x} = L_1(\gamma_1 + \gamma_2 d^{\beta}) + L_2\alpha \quad (3)$$

## 4. Integer Linear Programming Formulation

The WSN is represented by the graph $G(V, E)$, where $V = S \cup F$ and $E \subseteq V \times V$. $S$ represents the set of sensors nodes, $F$ represents the set of feasible sites and $E$ represents the set of wireless links. We distinguish two scenarios with mobile sinks. The first one is when there are multiple sinks moving in the entire network. The second one is when there are multiple sinks moving separately inside different clusters.

### 4.1. Mobile sinks moving in the entire network

The parameters and variables used to describe the problem are the following:

- Parameters
  - $m$ is the number of sinks.
  - $T$ (s) is the minimum duration of common time units for which the sink should stay at a certain feasible site.
  - $e_0$ (J) is the initial energy of each sensor.
  - $e_T$ (J/bit) is the energy consumption coefficient for transmitting one bit.
  - $e_R$ (J/bit) is the energy consumption coefficient for receiving one bit.
  - $g_r$ (bit/s) is the rate at which data packets are generated.

- $r_{ij}^k$ (bit/s) is the data transmission rate from node $i$ to node $j$ where the nearest sink stays at node $k$.
- $N_i^k$ is the set of $i$'s neighbors whose their nearest sink is at node k.
- $p_i^{k_1 k_2 \ldots k_m}$ (J/s) is the power consumed in sending and receiving data by sensor node $i$ when the first sink is located at node $k_1$, the second sink is located at node $k_2$ etc. and the $m$-th sink is located at node $k_m$.
- $p_i^k$ (J/s) is the power consumed in sending and receiving data by sensor node $i$ when the nearest sink is located at node $k$, $k \in F$.

- Variables
  - $Z$ (s) is the network lifetime.
  - $l_{k_1 k_2 \ldots k_m}$ is an integer variable which represents the number of times when the first sink is located at node $k_1$, the second sink is located at node $k_2$ etc. and the $m$-th sink is located at node $k_m$ for a duration of time $T$, $k_1 \in F$, $k_2 \in F$ and $k_m \in F$.

$$\max\ Z\ = \sum_{k_1 \in F} \sum_{k_2 \in F} \ldots \sum_{k_m \in F} T\ l_{k_1 k_2 \ldots k_m} \tag{4}$$

$$\sum_{k_1 \in F} \sum_{k_2 \in F} \ldots \sum_{k_m \in F} T\ l_{k_1 k_2 \ldots k_m}\ p_i^{k_1 k_2 \ldots k_m} \leq e_0,\ \ i \in S \tag{5}$$

$$l_{k_1 k_2 \ldots k_m} \geq 0,\ k_1 \in F, k_2 \in F, \ldots, k_m \in F \tag{6}$$

The equation (4) maximizes the network lifetime and determines the sojourn times of all sinks at feasible sites. The equation (5) assures that the energy consumed in receiving and transmitting data by each sensor node doesn't exceed its initial energy. This energy is computed when the first sink is located at node $k_1$, the second sink is located at node $k_2$ etc. and the $m$-th sink is located at node $k_m$.

The power $p_i^{k_1 k_2 \ldots k_m}$ is computed as following:

$$p_i^{k_1 k_2 \ldots k_m} = p_i^k \tag{7}$$

where $k$=$NearestSink(k_1, k_2, \ldots, k_m, i)$, $NearestSink$ is a function which returns the nearest sink node to sensor node $i$. This sink node is determined by choosing the shortest path as presented in Section 3.2.

In our model, the energy consumption coefficient for transmitting a bit denoted by $e_T$ and the energy consumption coefficient for receiving a bit denoted by $e_R$ are constant:

$$e_T = \gamma_1 + \gamma_2 d^\beta \tag{8}$$

$$e_R = \alpha \tag{9}$$

From the equation (3), the total power consumed at a sensor node is:

$$P_{total} = e_T L_1 + e_R L_2 \tag{10}$$

Therefore, the $p_i^k$ is calculated as follows[18]:

$$p_i^k = e_R \sum_{j:i \in N_j^k} r_{ji}^k + e_T \sum_{j \in N_i^k} r_{ij}^k,\ i \in S, k \in F, i \neq k \tag{11}$$

$$p_i^k = e_T\ g_r,\ i \in S, k \in F, i = k \tag{12}$$

At each node, the total of outgoing packets is equal to the total incoming packets plus the data packets generated[18]:

$$\sum_{j:i \in N_j^k} r_{ji}^k + g_r = \sum_{j \in N_i^k} r_{ij}^k,\ i \in S,\ \ k \in F \tag{13}$$

Using the two equations (11) and (13), we obtain:

$$p_i^k = e_R \sum_{j:i \in N_j^k} r_{ji}^k + e_T \Big( \sum_{j:i \in N_j^k} r_{ji}^k + g_r \Big),\ i \in S, k \in F, i \neq k \tag{14}$$

## 4.2. Mobile sinks moving separately in clusters

In this section, we formulate an ILP for a network divided in different clusters. The movement of the sinks is restricted to their cluster.

The variables and parameters that differ from section 4.1:
- $Z_j$ (s) is the network lifetime of the cluster $j$, $j \in \{1, 2, \ldots, m\}$
- $l_k$ is an integer variable which represents the number of times when the sink is located at node $k$, $k \in F_j$ for a duration of time $T$.
- $p_i^k$ (J/s) is the power consumed in sending and receiving data by sensor node $i$ when the sink is located at node $k$, $k \in F_j$.
- $F_j$ is the set of feasible sites of cluster $j$, $j \in \{1, 2, \ldots, m\}$ and $F$ = $\cup\, F_j$.
- $S_j$ is the set of sensor nodes of cluster $j$, $j \in \{1, 2, \ldots, m\}$ and $S$ = $\cup\, S_j$.

For each cluster $j$, $j \in \{1, 2, \ldots, m\}$

$$\max\ Z_j\ = \sum_{k \in F_j} T\ l_k \tag{15}$$

$$\sum_{k \in F_j} T\ l_k\ p_i^k \leq e_0,\ \ i \in S_j \tag{16}$$

$$l_k \geq 0,\ k \in F_j \tag{17}$$

The lifetime of network, which is the time until the first sensor dies, is determined by the following equation:

$$Z = \min_j (Z_j) \tag{18}$$

## 5. Simulation and Results

To evaluate the performance of the proposed ILP, we built a simulator in Java environment with variable number of sensors and sinks deployed on different grid sizes.

In order to determine the network lifetime and the optimal locations of sinks, we solved the proposed ILP with CPLEX[26] version 11.2. The calculation of the power consumed by the sensor node when the nearest sink is located at a certain node ($p_i^k$) was made by a program written in Java.

The values of parameter variables were chosen according to the following realistic assumptions. The initial energy at each node was chosen equal to the energy found in two Alkaline batteries AA of 1.5V usually 2600mAh i.e., $e_o$= 28080 J. We also chose the energy consumption coefficient for transmitting and receiving one bit the same as vendor-specified values for the Chipcon CC2420[27] where $e_T = 0.225\ 10^{-6}$ J/bit and $e_R = 0.2625\ 10^{-6}$ J/bit. We fixed the minimum duration of sojourn time $T$ of the sinks to 30 days because it is economically not efficient to have technicians relocating sinks in buildings very often. The rate $g_r$ at which data packets are generated is equal to 1 bit/s which is typical sampling rate for HVAC parameter control added the network layer encapsulations. Notice that real micro-controllers stop running when the battery voltage is below a given threshold. This depends on the micro-controllers and can not be taken into account here.

To get a deeper understanding of the efficiency of sinks mobility according to our proposed ILP solution, we investigated the network lifetime, the pattern of the distribution of the sinks sojourn times at the different nodes, the energy consumption and the residual energy at each node. Furthermore, we made a comparative study with four other schemes.

Thus, the following schemes were implemented:

1) **Static**: Static sinks placed at their optimal locations using the equation (19)
2) **Periphery**: Sinks moving in the periphery of the network
3) **Random**: Sinks moving randomly
4) **Cluster**: Sinks moving separately in different clusters according to ILP solution
5) **ILP**: Sinks moving in the entire network according to ILP solution

To compute the network lifetime and the optimal locations of stationary sinks, we used the following equation[18]:

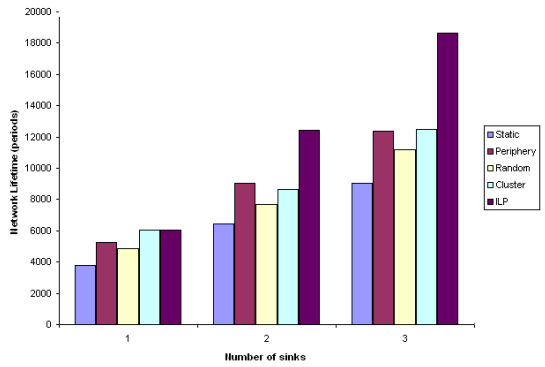
$$Z = \max_k \{\min_i (\frac{e_0}{p_i^k})\},\ k \in F,\ i \in S \qquad (19)$$

where $k = NearestSink(k_1, k_2, ..., k_m, i),\ k_1 \in F, k_2 \in F, ..., k_m \in F$.

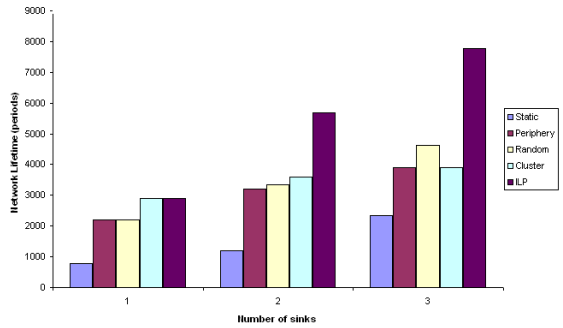
### 5.1. Network lifetime

We compared the network lifetime of our proposed ILP with the schemes described above by making a set of experiments. These experiments aim to study the effect of increasing, the number of sinks, the network size, the sinks sojourn times and the number of feasible sites, on the network lifetime.

In the first part of this section, we assume that all the sensor nodes are feasible sites.




(a) 5x5 grid network




(b) 10x10 grid network

Figure 3. The network lifetime

Figure 3 shows that in all the schemes and independently of the size of the network, the network lifetime increases notably when the number of sinks increases. Since the load traffic is distributed among a higher number of sinks, the nodes near the sinks forward less packets which leads to the reduction of energy consumption and lifetime improvement.

However in the Static scheme, the network lifetime is clearly shorter than in the other schemes with mobile sinks because nodes around the static sinks have to spend more energy to relay the packets of nodes farther away which leads them to drain their energy faster. Moreover, the first sensor dies relatively quickly in Periphery, Random and Cluster schemes comparing to the ILP scheme which manages to place optimally the sinks in the whole network.

The lifetime improvement percentages obtained in a 10x10 grid by deploying 3 mobile sinks according to the

ILP solution are 68 % against 3 mobile sinks moving randomly, 99 % against 3 mobile sinks moving separately in different clusters, 100 % against 3 mobile sinks moving in the periphery and 230 % against 3 static sinks.

Figure 4 shows that the network lifetime decreases considerably when the network size increases. This is explained by the fact that there are more data traffics. Hence, sensors which are near the sinks must retransmit a higher number of packets from their higher number of neighbors which leads to faster energy depletion.

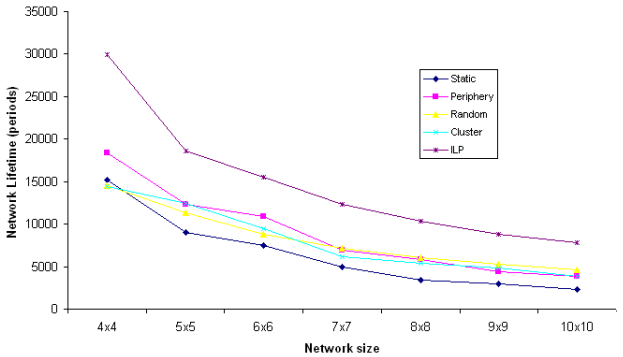



Figure 4. The network lifetime in different network size

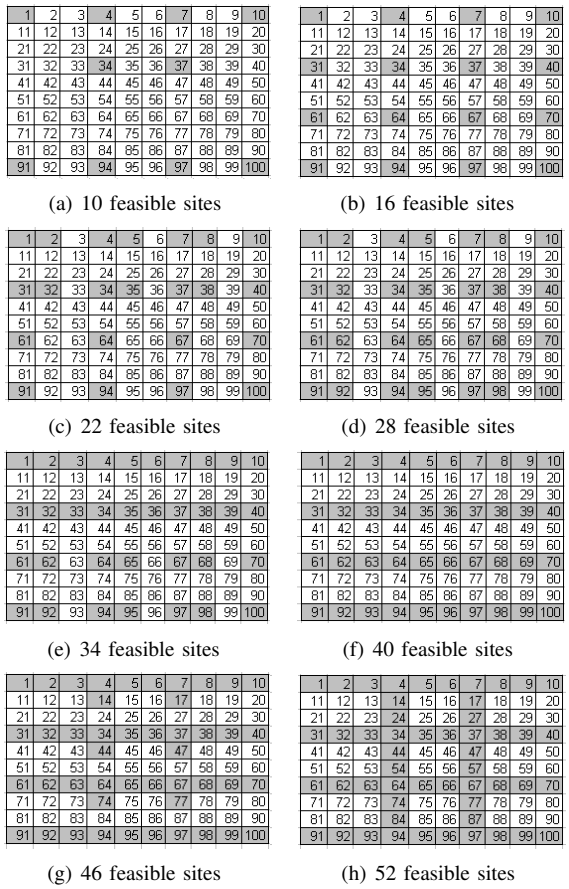
We investigated the network lifetime with different sinks sojourn times (see Table 1). The number of sinks was fixed to 3 and the number of sensors to 100. For Random, Periphery, Cluster and ILP schemes, the table shows that when the sojourn time of the sinks increases the network lifetime decreases slowly. In fact, the longer the sojourn time is, the less the sinks movements are which lead to shorter lifetime.

| Sinks sojourn times (days) | ILP | Cluster | Periphery | Random |
|---|---|---|---|---|
| 10 | 7774 | 3900 | 3892 | 4686 |
| 20 | 7773 | 3899 | 3892 | 4661 |
| 30 | 7772 | 3899 | 3891 | 4595 |
| 40 | 7771 | 3898 | 3891 | 4558 |
| 50 | 7770 | 3898 | 3891 | 4540 |
| 60 | 7769 | 3897 | 3890 | 4510 |

Table 1. The network lifetime (periods) with different sinks sojourn times

In the following, the sinks can move only on the feasible sites of Figure 5. Each node in the 10x10 grid is localized with its identifier number. The nodes in cells colored in gray are chosen as feasible sites.

We varied the number of feasible sites from 10 to 52 as shown in Figure 6. The number of sinks was fixed to 3, the number of sensors to 100 and the period sojourn time to 30 days. We notice that the network lifetime improves considerably when the number of feasible sites increases. In fact, the more the number of sites is, the more the sinks



(a) 10 feasible sites (b) 16 feasible sites

(c) 22 feasible sites (d) 28 feasible sites

(e) 34 feasible sites (f) 40 feasible sites

(g) 46 feasible sites (h) 52 feasible sites

Figure 5. The network with different number of feasible sites

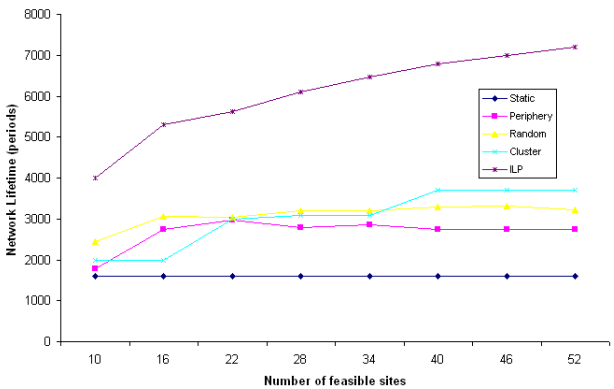



Figure 6. The network lifetime with different number of feasible sites

to efficient locations move which leads to longer lifetime. Nevertheless, the choice of the feasible sites positions has a great influence on the network lifetime.

### 5.2. The Sinks Sojourn Times

We studied the pattern of the distribution of the sinks sojourn times at the different nodes. All the nodes of the grid can be feasible sites. Figures 7, 8, 9, 10, and 11 show the sojourn times of three sinks in 10x10 grid for respectively Static, Periphery, Random, Cluster and ILP schemes.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 2349 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 2349 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 2349 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

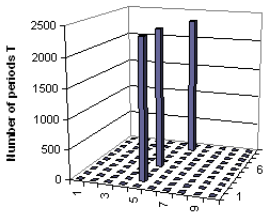



Figure 7. Sinks sojourn times at the different nodes of 10x10 grid network with Static scheme

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 325 | 325 | 325 | 325 | 324 | 324 | 324 | 324 | 324 | 324 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 324 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 324 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 325 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 325 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 325 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 325 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 324 |
| 324 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 324 |
| 325 | 325 | 325 | 325 | 325 | 325 | 325 | 325 | 325 | 325 |

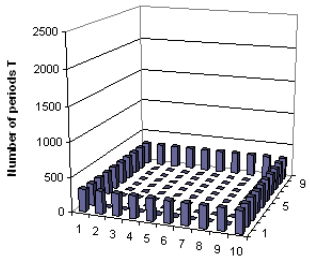



Figure 8. Sinks sojourn times at the different nodes of 10x10 grid network with Periphery scheme

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 19 | 67 | 50 | 46 | 46 | 66 | 35 | 39 | 37 | 37 |
| 45 | 42 | 34 | 45 | 47 | 47 | 42 | 41 | 43 | 32 |
| 37 | 31 | 25 | 33 | 14 | 41 | 46 | 46 | 38 | 39 |
| 39 | 43 | 42 | 44 | 37 | 41 | 39 | 38 | 33 | 28 |
| 47 | 35 | 40 | 34 | 33 | 37 | 41 | 43 | 41 | 34 |
| 41 | 36 | 38 | 31 | 41 | 44 | 48 | 41 | 35 | 40 |
| 57 | 43 | 33 | 33 | 49 | 46 | 44 | 48 | 44 | 31 |
| 32 | 35 | 40 | 43 | 45 | 45 | 38 | 45 | 37 | 44 |
| 42 | 38 | 44 | 45 | 33 | 49 | 55 | 40 | 34 | 33 |
| 41 | 38 | 38 | 50 | 40 | 50 | 37 | 35 | 44 | 18 |

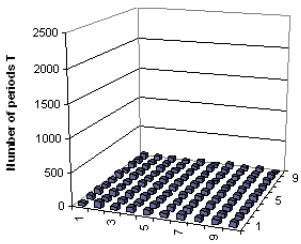



Figure 9. Sinks sojourn times at the different nodes of 10x10 grid network with Random scheme

The optimal sinks locations obtained for static sinks are the nodes which are almost at minimum distance i.e., number of hops to all other nodes. In Random scheme, the sinks sojourn time is variably distributed among all the nodes of the network. In the Periphery scheme, it is fairly distributed among the nodes in the perimeter of the network. For the Cluster scheme, the grid 10x10 was divided in 3 clusters 3x10, 3x10 and 4x10 in which the sinks move separately. We notice that the sink sojourns most of the times at the corners and the central grid area of its cluster. In the ILP scheme where the sinks move in the entire network, the same pattern is obtained, the sinks sojourn most of the times at the corners and the central grid area as shown in Figure 11.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 664 | 0 | 49 | 664 | 0 | 49 | 370 | 0 | 0 | 370 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 462 | 909 | 72 | 462 | 909 | 72 | 0 | 605 | 605 | 0 |
| 375 | 909 | 159 | 375 | 909 | 159 | 0 | 605 | 605 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 105 | 0 | 0 | 1 |
| 0 | 0 | 695 | 0 | 0 | 695 | 264 | 0 | 0 | 0 |
| 0 | 0 | 18 | 0 | 0 | 18 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 369 |
| 3x10 | | | 3x10 | | | 4x10 | | | |

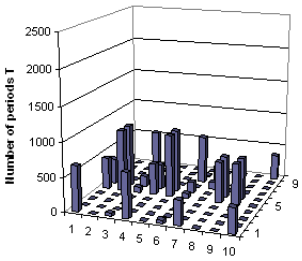



Figure 10. Sinks sojourn times at the different nodes of 10x10 grid network with Cluster scheme

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1193 | 165 | 99 | 186 | 241 | 232 | 177 | 84 | 145 | 1195 |
| 148 | 0 | 65 | 348 | 418 | 424 | 380 | 132 | 0 | 166 |
| 84 | 72 | 18 | 185 | 263 | 250 | 205 | 26 | 67 | 103 |
| 182 | 376 | 201 | 163 | 157 | 154 | 161 | 187 | 352 | 188 |
| 238 | 423 | 252 | 157 | 194 | 194 | 156 | 259 | 411 | 235 |
| 242 | 417 | 261 | 160 | 195 | 195 | 152 | 250 | 415 | 229 |
| 187 | 355 | 187 | 161 | 155 | 159 | 160 | 199 | 374 | 183 |
| 94 | 65 | 19 | 195 | 256 | 262 | 190 | 26 | 80 | 93 |
| 160 | 0 | 74 | 365 | 426 | 422 | 355 | 67 | 0 | 150 |
| 1194 | 152 | 90 | 180 | 238 | 240 | 181 | 95 | 158 | 1195 |

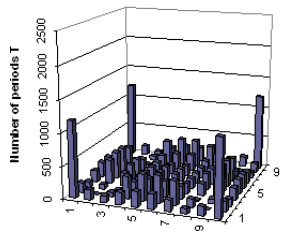



Figure 11. Sinks sojourn times at the different nodes of 10x10 grid network with ILP scheme

Independently of the size of the network and the number of the sinks as shown in Figure 12 and 13, the optimal locations of the sinks according to ILP solution are the four corners of the grid, then the central grid area. The sojourn of the sinks at the central grid area can be explained as follows. If one of the sinks is located in the center of the grid it will have four neighbors within the transmission range. Contrarily, if it is located in the perimeter or the corner, it will have respectively three and two neighbors. Obviously, the more neighbors within the transmission range the sink has, the better the traffic load balanced among nodes is and the higher lifetime is. However, independently of where the sinks are located, the sensors in the corners drain less energy on forwarding packets than the sensors in the center which are always along a routing path to reach the sinks. For this reason, the sinks sojourn more at the nodes in the corners than the nodes in the central grid area in order to consume the residual energy of the ”rich” sensors.

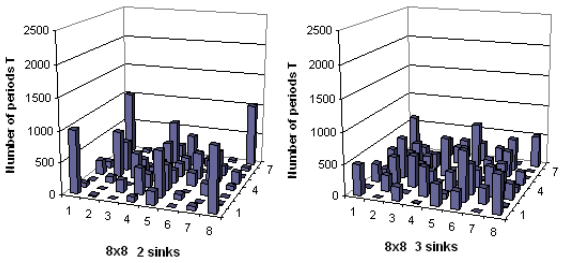



Figure 12. ILP sinks sojourn times at the different nodes of 8x8 grid network

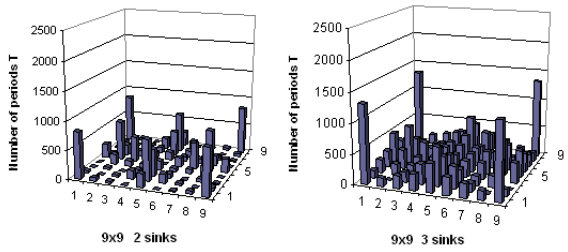



Figure 13. ILP sinks sojourn times at the different nodes of 9x9 grid network

### 5.3. The energy distribution

We analyzed the impact of the five schemes on the energy consumption at lifetime end in a network with 3 mobile sinks and 100 sensors. The distribution of energy consumption when the first sensor dies is depicted in Figures 14(a), 15(a), 17(a), 16(a) and 18(a). A light color means a higher percentage of energy consumption.

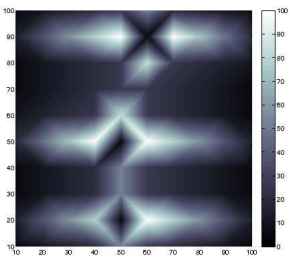
It is remarkable in all the figures that the energy consumption is highly variable and depends on the sinks locations. We notice that the nodes near the sinks have relatively higher energy consumption compared to most of the others because they have to receive and relay all other neighbors data in addition to their own data. This leads them to consume more energy.

In Figure 14(a), we observe that higher percentage of energy consumption is concentrated around three nodes in the grid which are the locations of the static sinks whereas the other sensors have a lower amount of energy consumption (dark color).

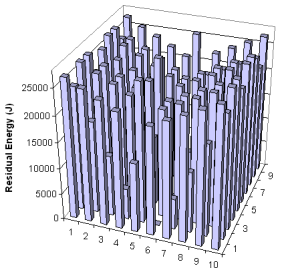
When the sinks move on the periphery of the network, the highest energy consumption occurs in nodes closest to the boundary of the network while the others nodes specially in the center consume less energy as seen in Figure 15(a).

In the Cluster scheme as shown in the Figure 16, the energy consumption is only balanced among the nodes of every cluster and not in the whole network. This is because of the restriction of sinks mobility to theirs clusters.

Figure 17(a) shows that Random scheme results in a better

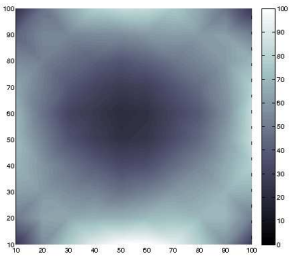


(a) Energy consumption

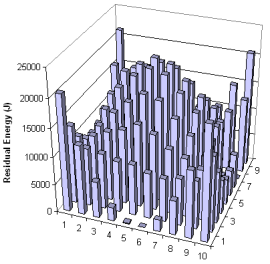



(b) Residual energy

Figure 14. Static scheme in 10x10 grid network

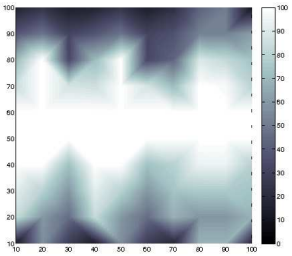


(a) Energy consumption

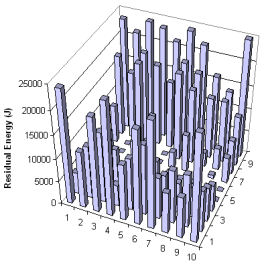



(b) Residual energy

Figure 15. Periphery scheme in 10x10 grid network

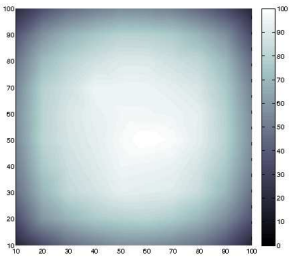


(a) Energy consumption

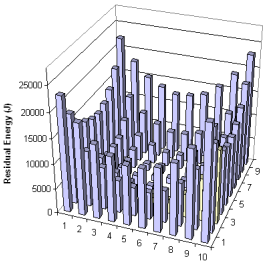



(b) Residual energy

Figure 16. Cluster scheme in 10x10 grid network

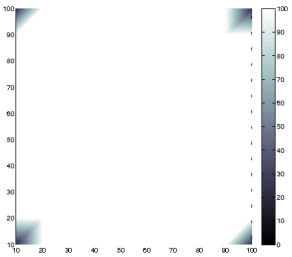


(a) Energy consumption

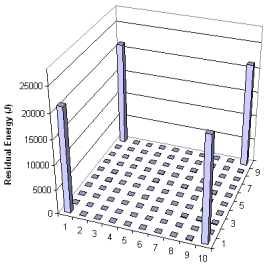



(b) Residual energy

Figure 17. Random scheme in 10x10 grid network



(a) Energy consumption




(b) Residual energy

Figure 18. ILP scheme in 10x10 grid network

balancing of energy consumption than Static, Periphery and Cluster schemes. In fact, we notice a larger area with light color.

However, ILP scheme balances almost perfectly the energy consumption among the nodes. In fact, the majority of the nodes depleted their energy at the same time except the four corners as shown in Figure 18(a).

The distribution of the residual energy at each sensor node in 10x10 grid with 3 mobiles sinks was also studied (see Figures 14(b), 15(b), 17(b), 16(b) and 18(b)).

With static sinks, the majority of sensors have more residual energy at the end of network lifetime than the schemes with mobile sinks which have their initial energies more depleted at the lifetime end.

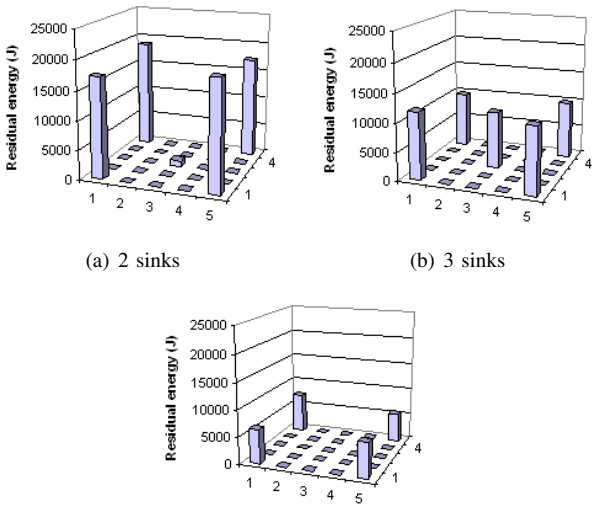
For the ILP scheme, sensor nodes have less residual energy than in the case of sinks moving in clusters. This is due to the fact that the mobility of sinks in the whole network changes the nodes acting as relays frequently and leads to balanced energy consumption among nodes. While, in the Cluster scheme, the movement of the sinks is restricted to their own clusters. So, this approach prevents to have global view of the entire network.

The ILP scheme results in a better distribution of residual energy among the nodes compared to Cluster, Periphery, Random and Static schemes. The results show that the percentages of the residual energy that remain unused at the network lifetime end for Static, Periphery, Random, Cluster and ILP schemes are respectively 71 %, 45 %, 31 %, 31 %, 3 %. Moreover, the number of sensors which have more than 50 % of their initial energies left at lifetime end is 80,

36, 20, 27 and 4 respectively for Static, Periphery, Random, Cluster and ILP schemes

It is remarkable in the ILP scheme, as shown in the Figure 18(b), that the sensor nodes in the corners have higher energy at the end of network lifetime than the others nodes. This can be explained by the fact that they do not drain their energy in forwarding neighbor's data in addition to their own data.

Moreover, we observe that at the end of network lifetime the distribution of residual energy in the grid is more balanced among the nodes when the number of sinks increases (See Figure 19). The use of more sinks results in the reduction of the average path length between the sensors and sinks thus enabling to achieve less traffic load to the nodes and increased network lifetime.




(a) 2 sinks (b) 3 sinks

(c) 4 sinks

Figure 19. Residual energy distribution of ILP scheme in 5x5 grid network with increasing number of sinks

These results are interesting since they are not pure theory. We tried to approach realistic buildings deployment parameters in our assumptions and especially with sinks that can not move frequently.

## 6. Conclusion and future Work

In this paper, we have explored the problem of positioning multiple mobile sinks in wireless sensor networks inside buildings, in order to avoid the energy hole problem and extend the network lifetime, which is really needed in practice.

As a solution, we have proposed an ILP which directly maximizes the network lifetime instead of minimizing the energy consumption or maximizing the residual energy, which is what was done in previous solutions. The proposed ILP determines the best way to relocate sinks by giving their optimal locations and the duration of their sojourn time.

A comparative study of the proposed solution with static sinks, mobile sinks moving in the periphery of the network, mobile sinks moving randomly and mobile sinks moving separately in different clusters was made. Relocating sinks with our solution and using realistic parameters assumptions results in the sensor network lifetime extension and the energy consumption more balanced among the nodes. The lifetime improvements achieved in our experiments by deploying 3 mobile sinks in the network with hundred sensors are almost 99 % against 3 mobile sinks moving separately in different clusters and almost 230 % against 3 static sinks.

The study of the pattern of the distribution of the sinks at different locations showed that the sinks sojourn most of times at the nodes in the central grid area and in the corners. This corresponds to very interesting feasible sites in buildings because the sinks can be easily relocated in the corridors which are often in the center of the buildings and provided with power and Internet access.

The weakness of the proposed approach is the scalability problem in networks with thousands of sensors due to the high ILP resolution complexity. In order to adopt this solution in a large scale wireless sensor network with more than hundreds of sensors, the network might be divided in several sensor sub-networks which will be deployed in the each part of the building. More than one sink can then be placed inside each zone to collect the information of the in-door environment. To implement the solution in a real environment and apply the model in real time conditions, technicians will be in charge of relocating the mobile sinks in the open areas of the buildings after long periods (i.e., months).

In our future work, we intend to improve the proposed ILP by including parameters and constraints that model more realistic requirements of an indoor environment (e.g., collisions) and take into account the energy consumption of sensing and data processing. We envisage also to introduce optimization in data routing by considering relevant metrics like latency, control overhead.

## References

[1] L. Ben Saad and B. Tourancheau, "Multiple mobile sinks positioning in wireless sensor networks for buildings," in *3 rd International Conference on Sensor Technologies and Applications SensorComm*, 2009.

[2] J. Pan, Y. T. Hou, L. Cai, Y. Shi, and S. X. Shen, "Topology control for wireless sensor networks," in *Proceedings of the 9th annual international conference on Mobile computing and networking MobiCom*, 2003, pp. 286–299.

[3] X. yang Li, W. zhan Song, and Y. Wang, "Topology control in heterogeneous wireless networks: Problems and solutions," in *Proceedings of the 23rd Joint Conference of the IEEE Computer and Communications Societies INFOCOM*, 2004.

[4] A. Sankar and Z. Liu, "Maximum lifetime routing in wireless ad-hoc networks," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM*, vol. 2, 2004, pp. 1089–1097.

[5] K. Fodor and A. Vidács, "Efficient routing to mobile sinks in wireless sensor networks," in *WICON '07: Proceedings of the 3rd international conference on Wireless internet*. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–7.

[6] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM*, 2004, pp. 629–640.

[7] S. R. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," *The IEEE Global Telecommunications Conference GLOBECOM*, 2003.

[8] W. Alsalih, S. Akl, and H. Hassanein, "Placement of multiple mobile base stations in wireless sensor networks," *IEEE International Symposium on Signal Processing and Information Technology*, 2007.

[9] X. Wu, G. Chen, and S. K. Das, "Avoiding energy holes in wireless sensor networks with nonuniform node distribution," *IEEE Transactions on Parallel and Distributed Systems*, 2008.

[10] ——, "On the energy hole problem of nonuniform node distribution in wireless sensor networks," *IEEE International Conference on Mobile Adhoc and Sensor Systems MASS*, 2006.

[11] J. Lian, K. Naik, and G. B. Agnew, "Data capacity improvement of wireless sensor networks using non-uniform sensor distribution," in *International Journal of Distributed Sensor Networks*, 2005.

[12] A. Bogdanov, E. Maneva, and S. Riesenfeld, "Power-aware base station positioning for sensor networks," in *Proceedings of the IEEE INFOCOM*, 2004, pp. 575–585.

[13] E. Oyman and C. Ersoy, "Multiple sink network design problem in large scale wireless sensor networks," *IEEE International Conference on Communications*, 2004.

[14] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon, "Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks," *Lecture Notes in Computer Science LNCS*, 2005.

[15] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," *In Proceedings 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM.*, 2005.

[16] K. Akkaya and M. Younis, "Sink repositioning for enhanced performance in wireless sensor networks," *Elsevier Computer Networks Journal*, 2005.

[17] B. Wang, D. Xie, C. Chen, J. Ma, and S. Cheng, "Employing mobile sink in event-driven wireless sensor networks," *IEEE Vehicular Technology Conference VTC*, 2008.

[18] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor networks lifetime," *In Proceedings of the 38th Annual Hawaii International Conference on System Sciences HICSS*, 2005.

[19] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang, "Controlled sink mobility for prolonging wireless sensor networks lifetime," *Wireless Networks*, 2007.

[20] Y. Bi, J. Niu, L. Sun, W. Huangfu, and Y. Sun, "Moving schemes for mobile sinks in wireless sensor networks," *Performance, Computing, and Communications Conference*, 2007.

[21] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: Modeling a three-tier architecture for sparse sensor networks," in *IEEE International Workshop on Sensor Network Protocols and Applications SNPA*, 2003, pp. 30–41.

[22] M. Marta and M. Cardei, "Improved sensor network lifetime with multiple mobile sinks," *Pervasive and Mobile computing*, 2009.

[23] Z. Vincze, R. Vida, and A. Vidacs, "Deploying multiple sinks in multi-hop wireless sensor networks," *IEEE International Conference on Pervasive Services*, 2007.

[24] A. Azad and A. Chockalingam, "Mobile base stations placement and energy aware routing in wireless sensor networks," *IEEE Wireless Communications and Networking Conference WCNC*, 2006.

[25] D. Maniezzo, K. Yao, and G. Mazzini, "Energetic trade-off between computing and communication resource in multimedia surveillance sensor network," *4th IEEE Conference on Mobile and Wireless Communications Networks MWCN*, 2002.

[26] *CPLEX http://www.ilog.fr/products/cplex/*.

[27] *Chipcon.* CC*2420 2.4* GHz IEEE *802.15.4 / ZigBee-ready RF transceiver*, 2004.

# Spatial Diversity Solutions for Short Range Communication in Home Care Systems using One Antenna Element

Markku J. Rossi, Jukka Ripatti, Fikret Jakupovic and * Reijo Ekman
*Mikkeli University of Applied Sciences, Finland*
** Turku University of Applied Sciences, Finland*
markku.rossi@mamk.fi

*Abstract-* ***It is becoming increasingly important for countries like Japan, Finland and Italy to enable as wide as possible home care for their elderly citizens because of the rapidly changing age structure of the population and high hospital costs. Wireless sensors hidden in furnishings ( as a reference to a Finnish ongoing project ) can provide necessary information for the high quality planning of home care visits. The relative positions of the radios are now fixed because of their relations to the furnishings. When conductive or metallic walls, floors or objects are found in the apartment, an antenna can hit a radio fade caused by shadowing or the summing up of waves. The Link Quality Indicator of a ZigBee® receiver can vary by up to 50 units within a short distance because of the floor, compromising the radio range. This paper describes the principles of a radio construction which is able to move the antenna element inside the enclosure of the radio unit along an X-shaped path along the major mechanical plane of the radio unit. Two electrical motors dynamically optimise the location of the internal antenna element to fit the current radio propagation environment. The paper also describes the feasibility and cost factors associated with the new structure. The achieved effect is a useful countermeasure against shadowing and multipath in the 2.44 GHz band but provides only a partial solution to the problem in the 868 MHz band, due to the space limitations in a typical radio unit. A prototype of the Diversity Adaptability radio was built and the capability to remarkably enhance the received power in difficult fading conditions was verified in an anechoic chamber. The observed side effects to the directional pattern of the radio were analysed to be removable by carefully selecting the route of the coaxial cable from the radio unit to the moving antenna element.***

*Keywords-home care; short range radio; sensors; ZigBee; shadowing; fading; spatial diversity; avoiding gain minima; diversity adaptability*

## I. INTRODUCTION

Mikkeli University of Applied Sciences ( MiUAS ) is situated in the South-Eastern district of Finland that has one of the most challenging predicted demographic structures for the 2020s in Europe. The eastern part of the district is expected to have 44700 inhabitants in 2020, 2130 of which will need daily care, 1580 because of memory disorders . The age structure of the population can be characterised by calculating what proportion of the population is children and of over 65 years old, divided by the number of people that are in the working age. The fresh population forecast gives a ratio 75,3% for Eastern Finland for 2020 /17/ . The situation will be more challenging for Eastern Finland than to the whole Finland. In Finland the ratio is caused by the unusually high birth rates in 1945 – 1955 and in Italy and Japan by the low birth rates after 1980. The forecasted population structures of these countries show challenges in financing the health care for the elderly [18].

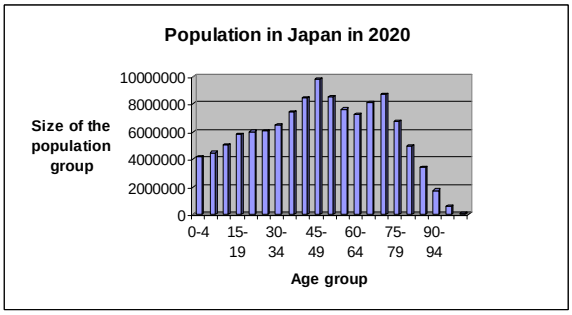



Figure 1. A forecast of the population structure of Japan for 2020 [18].

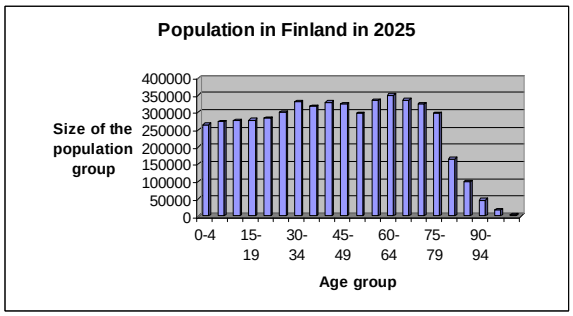



Figure 2. A forecast of the population structure of Finland for 2025 [18].

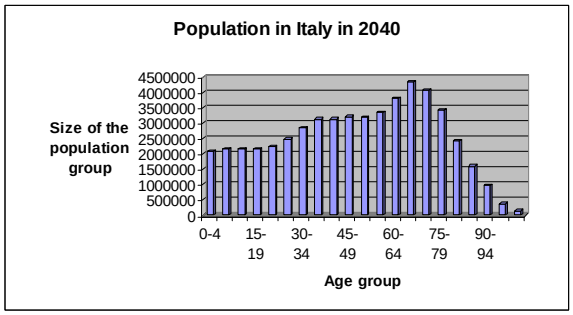



Figure 3. A forecast of the population structure of Italy for 2040 [18].

Data communication from homes has become feasible due to GSM and Flash OFDM ( at 450 MHz ) WWAN networks. Wireless communication has also become feasible at homes due to low cost short range radios such as ZigBee ®. The MiUAS generated an initiative to send to the hospital district home care visit recommendations generated at a call centre and based on real time event messages sent wirelessly from the homes of the elderly [14]. The project is now financed by the ERDF instrument of the European Structural Fund ( ESF ) and governed locally. The project built 37 test homes in 2 cities, 2 towns and 2 municipalities. We are aiming at 40% reduction in hospital days for a selected group due to increased home care, with corresponding cost savings. The main result will be the new work process. Recommendations of action based on real time sensor data have not been used in planning of home visits before. The decision rules for extracting recommendations of action from sensor data will also be new. The main technical challenges are how to maximise the time between replacing batteries to the sensor that detects motion in the living room and by providing countermeasures to the effects of RF shadowing and fading. Otherwise the ZigBee technology is already quite field proven, e.g. in the U.S. networks that read kWh-readings of consumed electricity from homes. The first challenge is related to different tactics to keep the receiver circuitry of the sensor device on as little as possible. The second challenge is related to the situation when the transmitter of the device is transmitting sensor data in an environment with shadowing. The battery life aspect was addressed in [2].

## II. MOBILE CARE IN EASTERN FINLAND PROJECT

The Mobile Care service will produce three kinds of products: alarms, emergency voice calls and recommendations of actions, like home care visits. The operators generating recommendations can view the following events:

- usage of a selected electrical appliance , such as a microwave oven
- motion in the living room
- pressure distribution on the bed
- whether the main door is opened or shut
- usage of the electrical cooker
- verification of taking medication pills out of a smart pill box
- whether the inhabitant is inside / outside of the apartment.

The homes have VoIP handsets with one button call initiation. The Central Unit has both a WWAN radio and a WLAN base station in addition to ZigBee. This also enables IP cameras. A ZigBee wrist device with an alarm button completes the set [14]. The recently founded company IsCom Oy continues the work of Riihonlahti Services Ltd in the project. The project started a field test with 37 homes in October 2009. The final project report is expected in April 2010.

A plan for the next phase includes a project that takes UML descriptions of the decision rules of creating action recommendations out of sensor data ( a result of the current project ) and develops an expert system that will deliver recommendations semi-automatically under the supervision of a nurse. MiUAS is now also a partner of a funding proposal to the EU COST programme for deep co-operation between organisations that have similar goals than MiUAS in the field of Smart Homes oriented to Health. The proposal was filed in September 2009 by Coventry University.

## III. RESEARCH NEEDS FOR REACHING THE REQUIRED LEVELS OF RELIABILITY

The error control methods: error control coding and retransmissions operate on a single wireless channel and add redundancy to the data packets to improve reliability. Their usefulness is however sometimes compromised over real fading RF channels. If the packet transmission both starts and passes the packet deadline in a deep fade, the methods mentioned before are usually not successful in delivering the packet [8].

The former reference is convinced that the incorporation of the spatial diversity approach into error-control is a very promising approach.

In the Mobile Care Eastern Finland concept some of the sensors can be invisibly embedded into furnishings. This eliminates the possibility of moving the sensor radio slightly so that heavy shadowing or a heavy slow fading can be avoided. We then decided to study spatial diversity to achieve as high as possible reliability in delivering the measurement results of the sensors. This is particularly important with the bed pressure distribution sensor. The later applications of this sensor can include e.g. an analysis of the breathing parameters during sleep.

## IV. RADIO PROPAGATION ENVIRONMENTS RELATED TO USAGE AT HOMES OF THE ELDERLY

A typical home where we are expecting to use the Mobile Care service in Eastern Finland is a wooden single family house. Typical dimensions for this house type are: floor area 8 x 8 m, a second floor with narrower rooms 2.7 m above the first floor and a fireplace downstairs built of bricks. The radio propagation here through the structures at 2.4 GHz has a fairly low level of shadowing. The findings in [7] support this conclusion. The metal casing above the stove and by the chimney typically cause some effects.

Another frequency available in Finland is 868.3 MHz according to the standard EN 300 220-1, with a longer range than for the ISM band with a 1 mW transmit power.

It is however also common that the inhabitant lives in a block of flats with steel reinforced concrete walls and floors. There is some heavy shadowing and reflections present particularly due to washing machines, water pipes and dish washers.

The reference [4] examined an office space of 30 x 15 m and found that the attenuation caused by a 0.14 m thick concrete wall was in the order of 5 dB.

The reference [5] uses the parameter Link Quality Indicator LQI to describe propagation effects. The LQI was interpreted there to represent well the chip error rate. In the case of a floor consisting of wood and polystyrene, sharp variations in LQI , 50 units out of the total range of 255 units along a 0.25 metre section of the propagation path were detected. The dielectric constant of polystyrene at 1 MHz is of the order of 2.75 [9]. The bending of radio waves is heavily correlated with the dielectric constant. It is evident that more studies would be needed on the effects of floor coverings made of different dielectric plastics materials.

In [6] measurements of RSSI ( Received Signal Strength Indicator ) were performed up to a distance of 32 metres in an indoor scenario. The Packet Error Rate ( PER ) was found to be low until the distance reached 25 metres. Severe degradation of 802.15.4 was however detected in the active space of a 802.11b/g ( WLAN ) base station, if the WLAN channels and the SRR ( Short Range Radio ) frequencies overlapped. There are however at least 4 pieces of 802.15.4 channels not severely overlapping with the 802.11b/g channels. The WLAN usage is at the moment very common in Finnish blocks of flats.

The rationale to study a solution that can move the antenna element to various positions inside the device is based on the characteristics of current two-element diversity antennas. These antennas have two elements at fixed positions from each other. The reference [3] points out that when the distance between the edges of the antenna elements is e.g. 41.7 mm ( very suitable for our purpose ), the gain of the combination has gain minima -22 dB .. -18 dB deep on the horizontal plane as can be predicted from the theory of antenna arrays. We want to avoid the existence of these gain minima.

## V. THE SPATIAL DIVERSITY SOLUTION

The design parameters of the 802.15.4 are well suited to our usage scenario because our expected required ranges as such are nearly always less than 25 metres. The challenge is mostly related to slow fading caused by the summing of the signals related to a direct path and reflections from metal surfaces.

The fitting of the sensor/radio devices discreetly into furnishings usually prevents the moving of the device with a built in antenna element which is required to avoid a deep fade. We decided to study the possibility of arranging spatial diversity that would be internal to the device and be arranged with only one

antenna element to minimise antenna, signal division and cabling costs in addition to avoiding gain minima at certain angles. No prior art studies about antenna elements moving inside the enclosure of a device, in connection with sensors for home care were found.

The typical device dimensions of a combined sensor, logic and RF circuitry are 50 x 50 mm without the battery compartment [10]. Willig [8] mentions that antenna movements of even a few centimetres are adequate to change the value of the spatial autocovariance function. The half wavelength in the ISM band is around 62 mm. If we move the antenna element along a path with a 45 degree angle to the sides of the enclosure, the available movement is of the order of 70 mm in an enclosure also housing a typical 2500 mm2 printed circuit board.

Benefits found in printed circuit antennas include: narrow bandwidth, mechanical strength, low cubic volume and low cost. A modern antenna design is presented in [3]. The bandwidth of a Hilbert inverted-F-antenna ( IFA ) for 2400 MHz is 3.3% or 80 MHz. This fits well with a scenario where we would use the $5^{th}$ and the $10^{th}$ channel of 802.15.4 so that they fall in between two wider WLAN channels. The WLAN channels are however this narrow only in the U.S., and in Europe wider WLAN channels are admitted, bringing more spectral overlapping to ZigBee and WLAN. These particular ZigBee channels are separated by 25 MHz. The Hilbert IFA would then reject the major parts of two WLAN channels and amplify only WLAN channel 6 and the two selected ZigBee channels. The dimensions of this Hilbert IFA are 2.7 x 5.6 x 1.5 mm and the gain is 1.0 – 1.5 dBi.

The mechanical construction consists of a plastic board with the movement route holes above the circuit board, with the flexible antenna coaxial cable attached to the centre of the circuit board.

Springs are attached to the 90 degree and 270 degree corners of the plastic board. Opposite to the springs there are two miniature electric motors. The Kysan Electronics FF-K10WA-5Z215 miniature motor has an 11.5 mm long chassis. It provides 0.06 mN/m torque at 14000 rpm and 0.09 W [11]. If the used pulley has a diameter of 3 mm, the required spooling for a 70 mm displacement is 7 full turns and the spooling time is 30 milliseconds. The other dimensions of the motor are: 6 x 8 mm. The antenna element slides along the X-shaped path drawn by one of the motors at a time. Whenever one motor is moving the antenna sledge the other motor is idle.

The antenna sledge is attached to both springs and both motors via lines. The basic orientation is with the sledge at the crossing of the two paths. The force against the spring is 0.04 N and the desired spring constant is 0.1 N/m, slightly less than the motor can stretch. The small pulleys for the motors can hold the line attached to the sledge.

When the antenna diversity module is 60 x 60 mm, the routes ( 2 ) of the sledge at 45 degree angles can be 70 mm long each, forming together an X-shaped movement area. This means that the length of the route from one end of the route to the centre is 35 mm and the maximum additional movement to any of the remaining three corners is also 35 mm .

When the sledge is moving, the motor that is not pulling the line, is releasing a maximum of 70 mm of line. The springs are capable of overcoming the sliding friction of the sledge. Therefore the sledge has also a micro sized servo in it to stop the movement wherever this is needed.

An example of a state-of-the-art servo is the Pico from GWS [12]. The dimensions are: 22.8 x 9.5 x 15.5 mm and the axle turns 60 degrees in 0.12 seconds. The activated servo will lock the sledge at the desired position against the pull of the springs.

When used together with the frequency of 868.3 MHz, the ratio of the dimension of the path of the antenna movement to the wavelength is only 0.2. It is then uncertain that the maximum improvement in the field strength would be as high as if operating at 2.44 GHz.

## VI. EFFECTS ON THE SENSOR DEVICES FROM THE ADDED DIVERSITY ADAPTABILITY FUNCTION

The coupling of the interference ( EMI ) from the digital circuits to the antenna is considerable in relation to the reception. The sharp edges of digital signals generate interference energy at high frequencies. The nonlinearities of the receiver can leak this energy into the received signal due to nonlinear mixing results. A sensitive receiver with fast digital circuits nearby is therefore usually a result of an iterative designing and prototyping process.

Now as we are moving the antenna unit above the logic circuits, the design of the electronic module is more challenging than ever. The complexity of the circuitry is however much lower than in a 3G cellular phone with fast processing and a GPS receiver.

Because of the Diversity Adaptability function the cubic volume of the wireless sensor device grows bigger. The effect from the empty space in between the electronics board and the plastic board is 36 cm3 if we allow 10 mm vertical space for the flexible coaxial cable for movement during the optimisation of the reception. The two motors consume 1.1 cm3 of space and the servo 3.4 cm3. The required space for the an-

tenna sledge compartment can be estimated to be 18 cm3. This can be compared with the volume of an AA sized primary battery, 8 cm3. Because our advanced device is now larger than a standard device, it is more difficult to hide it in certain kinds of furnishings.

The additional cost is: two motors 4.9 € , servo 10.8 € and cables plus additional mechanical gear and a larger enclosure 1 €, bringing the additional cost to 16.7 €. This means roughly doubling the component cost of the wireless sensor device. When compared to the new ability to dynamically minimise the packet error rate of ZigBee in changing propagation environments, the extra cost can sometimes be acceptable.

## VII. CONCLUSIONS FOR THE THEORETICAL CONCEPT

The critical nature of delivering sensor measurement results reliably in a service assisting the planning of medical home care visits generated a concept where one single antenna element is moving inside the wireless sensor device to minimise the packet error rate. The optimisation can adapt to a changing environment such as the later positioning of new home appliances in or out of the line-of-sight paths between the radios.

The concept shows theoretical promise to help in situations where the location of the sensor devices is fixed and cannot be fine tuned to avoid fades. The component cost of the device is doubled as is its cubic volume when compared to a conventional device.

A clear benefit of the solution compared to a competing scenario, the two-element diversity antenna, is the absence of gain minima at certain angles, associated with the pair of antenna elements.

I also recognised the need for more study on the effects of dielectric floor coverings to ZigBee propagation due to the possible bending of waves in the floor coverings.

## VIII. DESIGNING A PROTOTYPE FOR THE DIVERSITY AVAILABILITY ( DA ) FUNCTION

The design of the prototype was based on the principle presented at the 3rd International Conference on Sensor Technologies and Applications /1/ , ( FIG 4).

This article describes a real transceiver pair designed and built according to the concept from [1] and about its measured performance. From the wide selection of commercial ZigBee modules we chose devices from the product line of Dresden Elektronik Ingenieurtechnik GmbH in Germany. The unit providing an interface to the PC and to an external antenna is called the Sensor Terminal Board ( STB ). The dimensions are 100 x 75 mm. It has connectors for the actual radio board. The radio board is called the Radio Controller Board RCB230SMA. It has horizontal dimensions of 53 x 52 mm coming very close to our previous size assumptions.

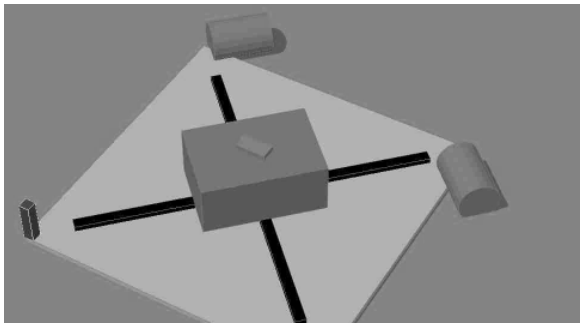


Figure 4. DA realised with two motors pulling a sledge against springs. The sledge has a servo for braking and an antenna element on it.

The radio Controller Board is capable of transmitting at +3 dBm. The transceiver chip is the Atmel AT86RF230. The board manufacturer gave us great co-operation and programmed the two STBs to use a constant frequency of 2,44 GHz and a transmit power of -10 dBm. These parameters would be suitable for the later tests in an anechoic chamber. In addition, the STB that would be selected to be the network device and receiving continuous temperature measurement data from the coordinator device was capable of outputting its own RSSI ( Received Signal Strength Indicator ) measurement results continuously and at suitable 1 second intervals to the USB connector.

The prototype of the network device uses a flat chip antenna instead of a higher quarter wave stub as originally planned. Samples of the interesting Hilbert IFA [3] are not available anymore so we selected a very small chip antenna from the manufacturer Yageo Corporation. Again, we received excellent co-operation and got samples of the Yageo 3216 PIFAs ( Planar Inverted-F ) antennas. Instead of the 2,7 x 5,6 mm dimensions from the original plan we now moved to the 1,6 x 3,2 mm range. The mean effective gain for Yageo is +0,14 dBi, slightly less than for the Hilbert IFA of [3]. The directional pattern is not perfectly omnidirectional as the maximum gain to a space angle is +3,7 dBi [13].

We then built the printed circuit board for the PIFA exactly according to the manufacturer's recommendations ( including the matching components ) and connected the antenna unit to the SMA coaxial connector of the Dresden RCB used as a network device with a 50 ohm cable of the type RG 178 B. The structure of the prototype network device can be seen from FIG. 5. A good feature of the construction is the 20 x 30 mm

printed circuit board for the PIFA, having a large ground plane visible on the picture against the electronics situated underneath. For the measurements the antenna unit was turned 180 degrees vertically compared to this picture.

The servo is the same unit as selected earlier, the GWS Pico [12]. The motors of the prototype are Motraxx K10WAs from Motraxx Elektrogeraete GmbH. The specifications are very close to the ones from Kysan [11].

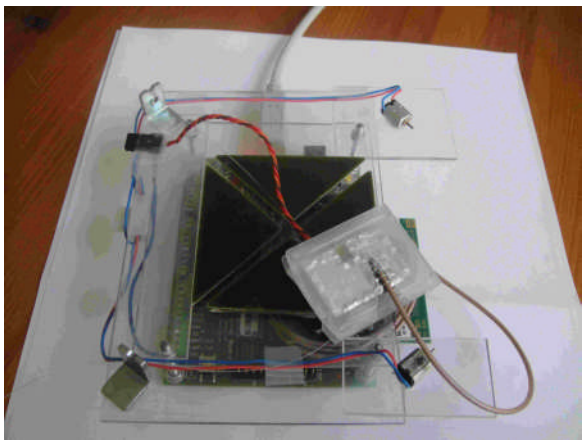


Figure 5. The DA antenna construction on Dresden RCB. The two motors are visible. The servo for the braking function is underneath the board which has the X-shaped hole.

## IX. LABORATORY TEST ARRANGEMENTS FOR THE PROTOTYPE

After investigating different service providers for anechoic chambers we selected the Turku University of Applied Sciences' chamber. Their facility is situated in SW Finland and uses a RFD-FA-100 room by ETS-Lindgren. The internal dimensions are : 6,4 m x 3,44 m x 3,0m (h). The level of internal reflections compared to the direct wave is -20 dB. The overall appearance of the anechoic chamber can be seen in FIG. 6.

The distance between the transmitter and the DA receiver was chosen to be 3,27 m. The height of the transmitter antenna from the floor was 1,15 m and 1,00 m at the receiver. The slight angle of 2,6 degrees for the line connecting the antennas compared to horizontal does not affect the received power. The transmitter antenna was an omnidirectional quarter wave stub from Dresden Elektronik.

The room has a non-reflective turning table that can be remote controlled from outside the chamber.

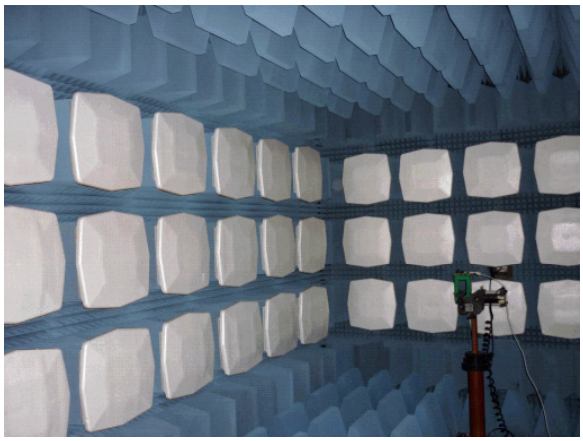


Figure 6. The anechoic chamber in Turku.

## X. LABORATORY TEST RESULTS

The simulated condition of a severe signal minimum caused by multipath was created with one 0,6 x 1,2 m aluminium plate hanging vertically and at a 90 degree horizontal angle compared to the signal path. The achieved weakening of the signal was 17dB.

The ZigBee radio working as a coordinator is seen in FIG. 7.

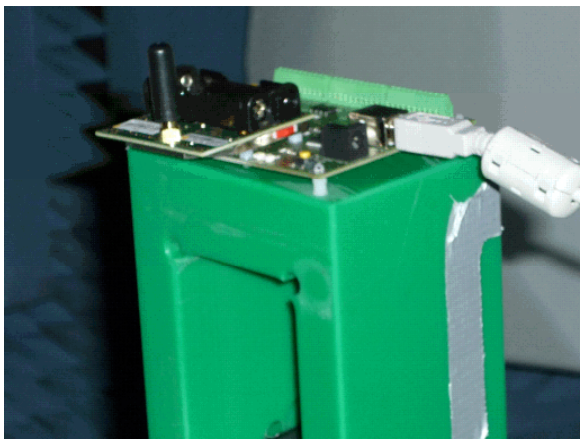


Figure 7. The ZigBee coordinator radio in the anechoic chamber.

The DA effect was achieved by selecting 8 adaptation positions for the DA antenna unit that is capable of moving along the X-shaped path. The direction "North" was at the receiver directly away from the transmitter antenna. The DA positions were named as NE ½ , NE1, SE ½ , SE1, SW ½ , SW1 , NW ½ and NW1. The positions " ½ " were halfway from the cen-

tre of the X towards the ends of one of the four 35 mm long DA paths. The positions " 1 " were 35 mm away from the centre of the X.

The results of the test for achieving escape from the fade are in TABLE 1 :

| Measurement no. | DA position | RSSI ( dBm ) |
|---|---|---|
| DA0 | middle of the X | -78 |
| DA1 | NE ½ | -67 |
| DA2 | NE1 | -70 |
| DA3 | SE ½ | -64 |
| DA4 | SE 1 | -63 |
| DA5 | SW ½ | -59 |
| DA6 | SW1 | -67 |
| DA7 | NW ½ | -79 |
| DA8 | NW1 | -62 |
| DA9 | middle of the X | -74 |

Table 1. The RSSIs when using the DA function. The unrotated RSSI before providing the back reflector was -59 dBm.

The elapsed time between DA0 and DA9 was 10 minutes. The usage of only one reflector caused a hysteresis of 4 dB during the DA test ( -74 dBm - -78 dBm).

The DA function was able to raise the RSSI by at least 15 – 19 dB. The best position for this test setting was the DA position SW ½ . The rotation of the antenna element during the linear movement was negligible and as we can see from the second measurement described below, the angular response at the angles close to the angle used in the test above was nearly flat.

The second test was the examination of the directional pattern of the DA unit antenna gain against the horizontal rotation angle.

The ZigBee network device on the electrically rotatable measurement table can be seen in FIG. 8.

The measurements were taken every 5 degrees, rotating clockwise when seen from above the DA unit with the starting point RSSI-0 pointing directly away from the transmitter antenna. The average RSSI over 72 measurements over the angles was -62,3 dBm. The average of the standard deviations of the powers at angles was 0.45 dB when the sample size at every angle was 20 samples. The dBm-readings were then rounded to the nearest 0.5 dBm reading to the table, because the radio gave the individual readings to the terminal emulation software only at a resolution of 1 dBm. The typical 90% confidence level of the power readings at the angles from the t-distribution with the sample size 20 is + - 0.78 dB.

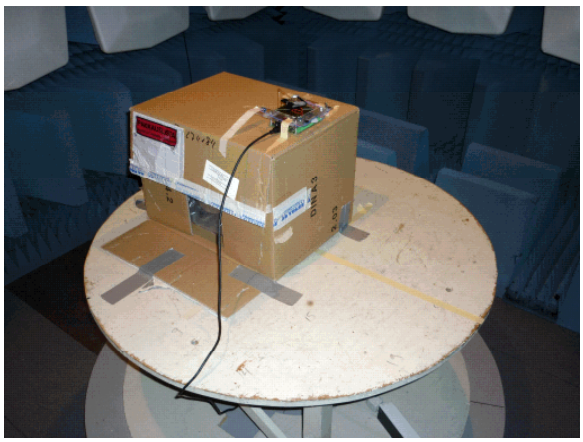


Figure 8. The ZigBee network device on the electrically rotatable measurement table.

The results of the measurement of the horizontal angular response of the prototype are in TABLE 2 :

| rotation angle, degrees | received power, dBm | **std deviation, dB** | rotation angle, degrees | received power, dBm | **std deviation, dB** |
|---|---|---|---|---|---|
| 0 | -59 | .497 | 185 | -66 | .51 |
| 5 | -58 | .497 | 190 | -64 | .384 |
| 10 | -58 | .357 | 195 | -64 | .447 |
| 15 | -56 | .384 | 200 | -62 | .384 |
| 20 | -55 | .447 | 205 | -61 | .384 |
| 25 | -55 | .447 | 210 | -61 | .384 |
| 30 | -54.5 | .49 | 215 | -60 | .384 |
| 35 | -54 | .6 | 220 | -59 | .316 |
| 40 | -54 | .477 | 225 | -58 | .384 |
| 45 | -54 | .316 | 230 | -58 | .384 |
| 50 | -54.5 | .497 | 235 | -59 | .384 |
| 55 | -55 | .384 | 240 | -58 | .447 |
| 60 | -58.5 | .49 | 245 | -58 | .436 |
| 65 | -61 | .447 | 250 | -59 | .447 |
| 70 | -66 | .548 | 255 | -60 | .384 |
| 75 | -70 | .775 | 260 | -58 | .384 |
| 80 | -67 | .632 | 265 | -58 | .384 |
| 85 | -65 | .539 | 270 | -58 | .218 |
| 90 | -63 | .316 | 275 | -60.5 | .497 |
| 95 | -61 | .316 | 280 | -61 | .477 |
| 100 | -61 | .384 | 285 | -61.5 | .589 |
| 105 | -62.5 | .49 | 290 | -62.5 | .592 |
| 110 | -64 | .447 | 295 | -64 | .384 |

| | | | | | |
|---|---|---|---|---|---|
| 115 | -67 | .316 | 300 | -64 | .436 |
| 120 | -70 | .539 | 305 | -64 | .6 |
| 125 | -82 | .853 | 310 | -62 | .384 |
| 130 | -74.5 | .589 | 315 | -62.5 | .477 |
| 135 | -67 | .447 | 320 | -64 | .497 |
| 140 | -67 | .316 | 325 | -67 | .548 |
| 145 | -69.5 | .654 | 330 | -64 | .316 |
| 150 | -67 | .384 | 335 | -62.5 | .477 |
| 155 | -64 | .384 | 340 | -61.5 | .583 |
| 160 | -66 | .384 | 345 | -61 | .384 |
| 165 | -67 | .384 | 350 | -61 | .477 |
| 170 | -67 | .51 | 355 | -61 | .384 |
| 175 | -70 | .624 | 360 | -59 | .316 |
| 180 | -66.5 | .497 | | | |

Table 2. The RSSIs of the DA prototype at different horizontal rotation angles.

These results can be seen also in polar format on FIG. 9.

The diagram shows the numbers of the rotation measurement stops from RSSI-0 to RSSI-73 with RSSI-0 and RSSI-73 at the same angle.

When compared to the specification of the Yaego 3216 PIFA which itself is not perfectly omnidirectional, we observed one new gain minimum caused by the construction of the DA unit prototype. The gain minimum corresponding to the rotation angle of 125 degrees caused an RSSI of -82 dBm. With the rotation angles of 120 and 130 degrees, the gain was already more than 10 dB higher.

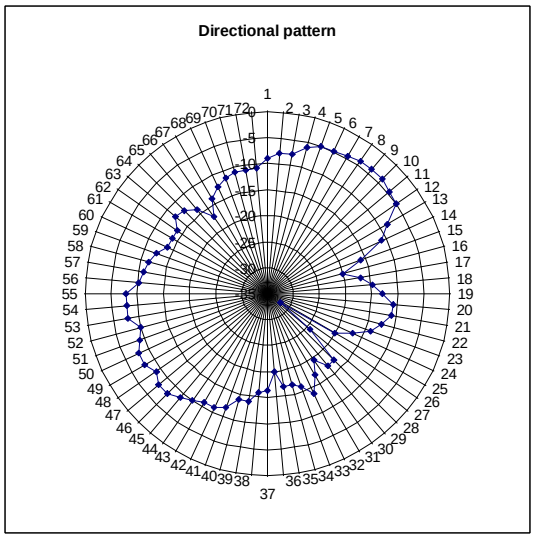



Figure 9. The directional pattern of the DA unit. The reference level 0 dB corresponds to an RSSI of -50 dBm.

The average received power was at -62,3 dBm. When the parameters :

- Tx transmit power -10 dBm
- Tx antenna gain 0 dBi
- average Rx antenna element gain +0,14 dBi
- distance between the Tx and Rx antenna elements 3,27 metres
- frequency of 2,44 GHz

are applied to the Friis free space equation

$$P_{received} = P_{transmitted} * G_{Tx} * G_{Rx} * \lambda^2 / (4 * \pi * r)^2$$

we get $P_{received}$ = 923E-9 mW = -60,3 dBm .

When also taking into account the loss in the RG 178 B coaxial cable between the Dresden RCB and the Yaego antenna element of 1,6 dB, we notice that the received power was only 0,4 dB less than the theoretical prediction. Actually the received power was the same as the predicted power within the measurement accuracy of the Atmel radio chip.

## XI. ADDITIONAL MECHANICAL PRINCIPLES

In addition to the current approach with one movement plane with one X-shaped path, the function can be realised with two movement planes. The lower plane has one straight hole in the horizontal direction of 45 degrees and is attached to an upper plane. The whole lower plane can move around a knob situated below. The second, upper plane has a corresponding straight hole at an angle of 135 degrees and can move along the hole that is at a 90 degree angle with the lower hole.

The hole can also be circular in shape, e.g. from an angle of 3 degrees to 357 degrees. The antenna element would be attached to a pole corresponding to the hand of a clock. The hand is attached to only one motor that is situated at the centre of the circle.

If the original one-plane with two motors construction is used, one additional release mechanism is needed close to one of the springs. This mechanism selects the required one of two possible directions at 90 degree angles to what the spring is pulling the antenna sledge.

For practical control of the position with good long term stability, a measurement of the position of the

sledge would be required for control feedback. The measurement can be optical, with an LED on the lower side of the sledge and photodiodes attached to the radio board under the movement plane.

## XII. ENERGY MINIMISATION ASPECTS AND SHADOWING

The main addition to the energy consumption in this concept compared to traditional solutions is during the running of the two Motraxx electrical motors one at a time. The usual starting position of the antenna sledge is from the centre point of the x-shaped antenna path. The efficiency of the motors is at a maximum with a 3.0 V supply voltage and with a 0.12 A current, providing rotation at 13800 RPM. The maximum distance travelled by the sledge during the measurement phase is 4 times 70 mm, or 280 mm. After this, the sledge simply moves to the most optimal measured position in terms of the packet error rate. The average distance travelled then is 17.5 mm. The end stage of the motor driver circuit is a switching transistor with a high efficiency.

With the planned pulley each 10 mm sledge movement takes one turn of the axle and lasts for 4.3 ms. The length of the steps is 10 mm. To find a new optimum antenna position would take an average of 30 steps. The duration of the whole procedure is dominated by the time to measure the new Received Signal Strength or the packet error rate and not by the movement periods of the antenna sledge.

The energy consumption of the motors during one position search is 3.0 V * 0.12 A * 30 * 0.0043 s = 0.046 Ws. This can be compared to a typical battery configuration with two pieces of CR123A Lithium primary batteries containing 9.0 Wh or 32400 Ws of energy. The position optimisation procedure is usually not a significant factor in the overall energy consumption.

Instead, the key question when designing a ZigBee radio network is the co-operation mode between the coordinator and the device of a star network. With the beaconed mode, the device can be asleep until the active period begins after a selected inactive period. During the inactive period the device can be asleep and only providing power to a low power real time clock. The power consumption of e.g ZigBee devices is nearly the same during a reception operation and a transmission operation. If the beaconed mode is used the energy consumption can be predicted quite well.

To reduce the energy consumption further the unbeaconed mode can be selected instead. Now the device can be asleep until it detects itself new sensor data to be reported, wakes up and performs a carrier sense operation. If no other device in the network is transmitting at that moment, the device can then send the sensor report.

In the case the service however wants to receive sensor reports on demand, a novel principle was introduced in [2]. If the beaconed mode is used and the inactive period is very long the coordinator cannot get sensor reports according to the needs of the higher system. Instead of keeping the device in the beaconed mode with periodically turning on the receiver circuitry, the waking up of the device is now arranged via an UV detector that has a negligible energy consumption. The coordinator emits bursts of UV light to wake up the device via an interrupt line of the sleeping microprocessor. This concept needs an optical path from the coordinator to the device.

The Diversity Availability system presented in here can run into problems in the energy consumption if there is a lot of movement of shadowing objects in the path between the coordinator and the device. When using the 2.4 GHz band the wavelength is of the order 0.13 metres. From the Limits of Exposure to RF Radiation we know that the human body is not especially much absorbing RF energy in the exact 2.4 GHz band in general. That is why an external power density of 10 W/m2 is accepted in Finland at this band. However, the used frequency is indeed used for heating food in microwave ovens. The heat is produced by the vibration of the various dipoles of the molecules in the EM field. There is no sharp absorption band for the water molecule in this band against the common assumption and e.g. also 915 MHz is used for heating food.

Therefore the presence of a human body close to the device in between the coordinator and the device can be seen as a normal shadowing case. When the dimension of the body is of the order twice the wavelength, the scattering cross section of the body is remarkable, or approx. 0.3 – 1 times the body dimension. When seen from the wave penetration point of view, the skin depth of a 2.4 GHz plane wave into the muscles and organs of a typical human body is close to 1 cm. This means that severe shadowing is experienced when a human is standing close to a ZigBee device.

The effect to the energy consumption is dependent on the frequency of the human movement related to the antenna position optimisation cycle. Because many RSS measurements are needed for each optimisation cycle, the cycle lasts at least 20 seconds. The cycle should be started only when shadowing has been observed for at least 20 seconds. The worst case would then be that the human changes the position in front of the device slightly every 40 seconds. A remarkable negative effect to the operational time would be observed if 20% of the battery energy would have been consumed to the mere continuous optimisation of the position of the antenna sledge. The dominating factor

would be the need to keep the receiver on during the optimisations, not the energy used by the motors. The continuous receiver-on status would consume 20% of our battery energy in just 3 hours. This should clearly be taken into account when selecting the position of the sensors at homes.

To optimise the overall combination of the data transmission quality and energy consumption in a star network one should analyse the needs of the higher system using the services of the sensor network. The energy consumption coming from the active usage of the receiver circuitry is important in the overall energy consumption [16]. The power consumption of the example transceiver IC, Atmel AT86RF230 is 46 mW when the receiver is on and the transmitter is off. The difference between the power consumptions at the RF output powers +3dBm and -17 dBm ( the whole range that can be controlled by sw ) is in comparison only 21 mW.

The key question is how often does the higher system need new sensor data. If sensor data updates are needed at any moment according to the demand, long periods of inactive time cannot be realised in the beaconed mode. The inactive period must be set to a lower value than the maximum permissible delivery time for a sensor data report. An alternative to this is to use an optical turn-on activation principle together with unbeaconed mode for the receiver, presented in [2] with an erratum in [1]. This principle minimises the time with receiver on.

When active, the IC can usually provide functions that can be used to minimise the used energy. The main functions are the RSSI measurement, the ED ( Energy Detection ) calculation and the LQI ( Link Quality Indicator ) calculation. RSSI indicates the received power of an individual data frame. The ED calculates an average of several RSSIs. These are related to the received power only. The LQI studies what the estimate for the PER ( Packet Error Rate ) is from analogue parameters. The LQI then takes into account also errors generated by possible RF interferers. The higher the PER is, the more often we must retransmit the data, consuming energy in the process. When the LQI shows 150, the PER is around 0.1 and when LQI is 220, the PER is already negligibly small.

The DA functionality can be used to achieve the best possible energy consumption in a given geometrical configuration and multipath and interference environment. If the radio channel is selected first out of the 16 alternatives so that the LQI is maximised we have avoided the interferers, including our own sensors, as much as possible. After this, the DA function can be used to find the best position for the antenna inside of the sensor device. After that, the LQI can be used to drop the transmit power until the PER is still acceptable together with the retransmission rate following from it.

## XIII. CONCLUSIONS FOR THE PROTOTYPE MEASUREMENTS

The capability of the Diversity Adaptability function to increase the received signal power was verified by measurements. The DA unit won extra link margin for home care and other similar applications where the position of the receiver cannot be chosen at the level of +17 dB. If the sharply shaped space with high fading would have had even deeper fading, even more benefit could be possible. Here the depth of the fade was limited by the characteristics of the anechoic chamber.

The observed new minimum at the directional pattern requires some examination. The coaxial cable from the Dresden RCB runs in a horizontal position beside the Yaego antenna element. The incoming field was amplified less than with other angles, when the first motor was at an angle of 195 degrees from the Tx-antenna line. The starting angle of the first motor before rotation was +70 degrees. The horizontal coaxial cable started from the angle +30 degrees. It was at 155 degrees when the minimum was detected.

The interaction with both the horizontal coaxial cable and the first motor caused a strong deformation to the angular response. The direction to the transmitter antenna was 180 degrees when related to the set null of the rotation experiment. There was deformation of the field due to the shield of the coaxial cable whose potential is close to the potential of the reference electrical ground and which was now in the path between the transmitter and the receiver antennas.

The second motor was respectively first situated at -60 degrees from the Rx antenna element. It was between the Tx and the Rx antenna elements at the rotation angle of 240 degrees. The angular gain of the antenna construction was however good ( RSSI = -58 dBm ) in this case, at the rotation angle of 240 degrees. The mentioned angles take into account the fact that the Yaego element was not in the middle of the printed circuit board of the antenna element.

The horizontal coaxial cable is the main cause for the drop in sensitivity at the rotation angle of 125 degrees, corresponding to the measurement angle no 26 on the diagram. A motor alone did not remarkably disturb the field. The diameter of the motors is only of the order 7 mm.

To overcome this side effect, the coaxial cable should be led through the centre of the sledge when designing an operational product. We concluded that when this principle is applied, the observed gain minimum will disappear. To verify this postulation, we

selected later, after the chamber measurements an open field with low ground conductivity to test a modified device. Because the RSSI function of the radio does not work with powers lower than -91 dBm and the notch detected at the rotation angle 125 degrees was 20 dB deep, we selected 7.0 meters to be the antenna distance in the field test. The free space power expected with this arrangement is -69 dBm overall and -89 dBm at the rotation angle of 125 degrees.

The measured power was -65 dBm indicating some reflections from the ground. The receiver was again rotated to find the notch angle. The notch was found at the expected angle and the depth of the notch was now 18 dB.

The receiver was now modified so that the coaxial cable comes to the antenna unit directly from the down direction. The receiver was again installed on the field and it was rotated to find out if the notch was still existent. The lowest detected power was now - 73 dBm instead of the -83 dBm with the coaxial cable oriented horizontally. The received power was still around the level -65 dBm at most of the angles indicating the performance of the antenna unit was now the same but more even than before the modification.

The hysteresis experienced with the back reflector arrangement suggests that very local and deep fades might not be very common in practical situations.

Also another phenomenon could at least theoretically reduce the number of situations where DA would be mandatory in practical systems. This phenomenon is the limited cross-polarisation ratio CPR of commonly used antennas. CPR is the gain of the cross-polarised component divided by the gain of the copolarised component /15/. For a short vertical dipole the CPR is larger ( less perfect ) at large elevation angles than at small angles. The CPR ranges commonly around -20 … -10 dB and is caused especially by the physical thickness and slight mechanical errors of the constructions. After polarisation changes due to interaction with conductive obstacles in the apartment a part of the signal will leak into the receiver antenna also via the polarisation that was not the original design criteria for the antenna.

To study the level of probability that a DA receiver would be mandatory in apartments a large simulation set up could be arranged. Another possibility would be to compare conventional and DA receivers in a real and demanding multipath environment.

## ACKNOWLEDGEMENTS

We wish to thank Mr. Mike Ludwig of Dresden Elektronik for providing the embedded software that sent the RSSI measurements from the radio chip to the USB port.

We wish to thank Mr. Remco Hollemans of Yaego Corporation for sending samples of their small Planar Inverted-F antennas.

We wish to thank Mr. Elmar Bernhardt of the Elektroniikan 3K-tehdas of the city of Savonlinna for designing and building the printed circuit board for the Yaego antenna element.

## REMARKS

ZigBee ® is a trademark owned by ZigBee Alliance, Inc.

## REFERENCES

[1] Rossi, M.J. **,** *Spatial Diversity for Short Range Communication in Home Care Systems using one Antenna Element.* SENSORCOMM 2009. The 3rd International Conference on Sensor Technologies and Applications, IARIA , 2009. 978-0-7695-3669-9/09 , IEEE 2009 , DOI 10.1109/SENSORCOMM.2009.35 .

[2] Rossi, M.J. **,** *Energy Efficient Motion Detection of Elderly Living at Home.* SENSORCOMM 2008. The 2nd International Conference on Sensor Technologies and Applications, IARIA , 2008.

[3] Azad, M.Z. and Ali, M., A New Class of Miniature Embedded Inverted-F Antennas ( IFAs ) for 2.4 GHz WLAN Application.. IEEE Trans. on Antennas and Propagation, Vol. 54, No. 9, 2006, pp. 2585 – 2592.

[4] Lee, J. et al. , *RF Propagation Simulation in Sensor Networks.* The 2nd International Conference on Sensor Technologies and Applications. IEEE , 2008.

[5] Machedon-Pisu, M. et al. , *Efficient Data Propagation Techniques and Security Concerns in Low Rate Wireless Personal Area Networks in Outdoor and Indoor Scenarios.* Optimization of Electrical and Electronic Equipment - OPTIM 2008. IEEE , 2008.

[6] Petrova, M. et al. , *Performance Study of IEEE 802.15.4 Using Measurements and Simulations.* WCNC 2006 Proceedings. IEEE , 2006.

[7] Tang , L. et al. , *Channel Characterization and Link Quality Assessment of IEEE 802.15.4-Compliant Radio for Factory Environments.* IEEE Transactions on Industrial Informatics, Vol. 3, No. 2, 2007. pp. 99 – 110.

[8] Willig, A. , *Recent and Emerging Topics in Wireless Industrial Communications: A Selection.* IEEE Transactions on Industrial Informatics, Vol. 4, No. 2, 2008. pp. 102 – 124.

[9] *Electrical Properties of Plastic Materials. Professional Plastics Inc.* http://www.professionalplastics.com/professionalplastics/ElectricalPropertiesofPlastics.pdf , *accessed in Feb. 2010.*

[10] *Rihotec Detector with ZigBee Communication. Personal communication, Pertti Harju, Riihonlahti Services Ltd., Puumala, Finland, 2008.*

[11] *DC Motor FF-K10WA-5Z215 . Data sheet. Kysan Electronics, Mountain View, CA, 2003.*

[12] *Sub micro sized Servo GWS Pico. Data sheet. Grand Wing Servo-tech Co., Ltd., Taipei, Taiwan.* http://www.gwsus.com/english/product/servo/sub%20micro.htm , *accessed in Feb. 2010.*

[13] 3216 *Ceramic Chip Antenna in PIFA Mode for Bluetooth/WLan Application. Preliminary. Yageo Corporation, Taiwan, 2009.*

[14] Rossi, M.J., Pärnänen, A. and Luukkainen, S. , *Mobiilihoiva turvallisen kotihoidon tukena. Täydennetty projektisuunnitelma.* Project Plan to the ERDF funding programme, Mikkeli University of Applied Sciences, August 2008, in Finnish.

[15] Yau, K.S.B. **,** *Ionospheric Propagation , The Experimental System.* http://digital.library.adelaide.edu.au/dspace/bitstream/2440/50413/4/03chapter4.pdf , *accessed in October 2009.*

[16] *AVR Low Power 2.4 GHz Transceiver for ZigBee, IEEE 802.15.4 , 6LoWPAN, RF4CE and ISM Applications. AT86RF230 Data Sheet. Document 5131E. Atmel Corp. , 2009.*

[17] Population Projection 2009-2060, *Statistics Finland,* http://www.tilastokeskus.fi/til/vaenn/2009/vaenn_2009_2009-09-30_tie_001_en.html, *accessed in Feb. 2010.*

[18] International Data Base, *U.S. Census Bureau,* http://www.census.gov/ipc/www/idb/informationGateway.php , *accessed in Feb. 2010.*

# Analysis and Experimental Evaluation of Network Data-Plane Virtualization Mechanisms

Fabienne Anhalt and Pascale Vicat-Blanc Primet
INRIA, LIP - ENS Lyon
{fabienne.anhalt, pascale.primet}@ens-lyon.fr

## Abstract

*Combining end-host, server and router virtualization could offer isolated and malleable virtual networks of different types, owners and protocols, all sharing one physical infrastructure. However, the virtualization of the data plane may lead to performance degradation and unpredictability. These arise not only due to additional processing, but also from the sharing of physical resources like memory, CPU and network devices. This article analyses network virtualization from the data plane's perspective. We explore the resulting network performance in terms of throughput, packet loss and latency between virtual machines, as well as the induced CPU cost. The virtual machines act as senders or receivers, or as software routers forwarding traffic between two interfaces in the context of Xen and KVM. Our results show that the impact of virtualization with Xen has become smaller with its successive versions. Moreover, compared to KVM, Xen gives better network performance with less processing overhead. Therefore, Xen currently seems to be one of the most interesting software data-plane virtualization technologies.*

***Keywords*-Network virtualization; data plane; performance; Xen; KVM**

## I. Introduction

System-virtualization technology has been recently redefined in the context of distributed systems and enterprise servers. By running several virtual servers on one physical machine, organizations and companies can better exploit the physical resources of one machine in terms of CPU and energy consumption: often, a server uses only a small part of the available hardware resources. Moreover, server virtualization allows increasing security, due to the isolation factor. Also, migration and mobility enhance reliability.
To introduce these benefits into the network itself, an emerging idea is to implement virtualization not only on the end-host servers or at link level, but also on network devices, like routers. This technology could make coexist networks or autonomous systems of different types, owners, and protocols over one physical network. Such virtual IP networks are an emerging approach to provide virtual network infrastructures as a service, allowing users to create customized networks with personalized IP routing paradigms over a shared physical infrastructure. They can be used for research (implementation and test "in the wild" of new protocols) and further to decouple the service from the infrastructure in the real Internet, rethinking the backbone architecture.
However, if virtualization could potentially solve the main issues of the current Internet (security, mobility, reliability, reconfigurability), it would nevertheless introduce an overhead due to the additional layers inserted between the virtual machines and the hardware. In particular, if the data plane is virtualized, which allows the greatest customization, all network traffic has to cross these virtualization layers, which impacts performance. Furthermore, in addition to the routing and bandwidth-allocation problems of traditional routers, we have to take into account the problem of local-resource sharing by different virtual networks. Considering this sharing of resources—e.g., the network interfaces, the processors, the memory(buffer space), the switching fabric—it is a challenge to get a predictable, stable and optimal performance.
To have a better insight into these issues, this article analyzes the current network data-plane virtualization solutions. We evaluate their properties and potential in terms of network performance like throughput, packet loss, latency and the induced CPU cost. End-host network performance is first analyzed to evaluate virtualization in a simple sending or receiving scenario with a single network interface, using different system configurations. We propose our design of a virtual router prototype with a virtualized plane, and then evaluate virtual forwarding.
This article is an extension to our previous work on Xen's network performance [1]. It details the formerly announced

results and proposes a comparison to the recent KVM virtualization technology.
The rest of this article is organized as follows. The next section describes the context and background of network virtualization. Section III analyses the technology of virtualizing the data plane. Section IV discusses experimental results on network performance. Finally, section V compares our results to related work, and section VI concludes this article.

## II. Background

### A. Towards network virtualization

Virtualization came up to cross the barrier of physical hardware, and to share a resource between several users, giving each one the illusion that the resource belongs entirely to him. It was introduced by IBM in 1973 [2] and became very popular with the arrival of solutions like Xen [3] and VMware [4]. This approach is very useful to enhance isolation, mobility, dynamic reconfiguration and fault tolerance. It is now widely used by companies and institutions using only a few physical servers to host a multitude of virtual servers. Multiplying their servers this way allows them to isolate services, facilitate reconfiguration and maximize fault-tolerance and flexibility. They can easily duplicate servers, and thus deal with hardware problems, as virtual servers can be migrated from one physical machine to another if necessary. Thus, virtualization makes institutions less dependent on the hardware while optimizing their utilization. The emerging cloud concept currently pushes the level of abstraction from the hardware even a step further. Commercial cloud providers (e.g., [5][6][7]) propose to host the company's data-centers and services inside virtual machines on the servers they manage. Hence companies do not need to maintain hardware servers anymore. By externalizing their data and services, they can considerably reduce costs, and enjoy more flexibility, as they can rent the necessary amount of server resources on demand.
While contents and services moved to a virtual environment, the network has to take the same direction, to get the same benefits. By virtualizing not only end-host servers, but also network nodes and links, the physical network infrastructure can be shared by several virtual networks, thus optimizing resource utilization. Users of virtual networks have the illusion to manipulate a dedicated network, and virtual-node and link-migration simplicity can improve fault tolerance and optimize resource allocation. Hence, the network becomes a flexible resource which can be booked on demand, for example by providers, just like companies rent clouds today. This gives virtual-network providers the flexibility to allocate resources dynamically over the physical network and have end-to-end control on their virtual paths [8]. In this scope, regarding the utilization of virtual networks on the Internet, isolation is very important to guarantee confined virtual environments with performance guarantees to each virtual network operator who wants to offer a continuous service to its clients.
In this context, research was conducted on infrastructure virtualization, including virtual routers as a new resource. As an example, VINI [9] allows several virtual networks to share a single physical infrastructure. Researchers can run experiments in isolated virtual network slices with real routing software they can personalize. Trellis [10] is a network-hosting platform deployed on the VINI facility. It is a virtual-network substrate that can run on commodity hardware. It is built using VServer [11] container-based virtualization. In this implementation, only the control-plane is virtualized which enables only limited isolation between virtual networks. However, these solutions show interesting new functionalities, hosting several virtual networks with customizable routing paradigms on a single physical infrastructure. Therefore, data-plane virtualization is required in addition to control-plane virtualization to add the right level of confinement and enable fairness.

### B. Data-plane virtualization techniques

Virtualizing the data plane means having a separate and independent data plane and virtual hardware in each virtual machine. To manage the virtual hardware, each virtual machine runs a separate OS and the virtualized hardware is shared among the virtual machines. Basically, this sharing can be performed by software emulation of the hardware, exposing emulated hardware to each virtual machine [12] or using hardware with virtualization extension [13][14]. As emulation requires the translation of all the instructions, it has a very important performance overhead. Using hardware virtualization, virtual machines instructions are executed directly on the hardware which offers accelerated performance. Nonetheless, generally not all hardware supports virtualization. If modern CPUs are provided with hardware virtualization support, most of the time I/O devices like network interfaces need to be emulated in software to be used in fully virtualized systems.
An alternative to emulation is *paravirtualization*, where each virtual machine uses special virtual drivers to access the hardware devices. This requires patching the virtual machine's OS to include the virtual drivers, but device access is more efficient because the virtual driver's design is adapted to the virtualization mechanism. For this reason, paravirtualization seems to be the most promising solution for network-intensive virtual machines—assuming the virtual machine kernels can be patched, which is now possible for the major OSes. Figure 1 illustrates the difference

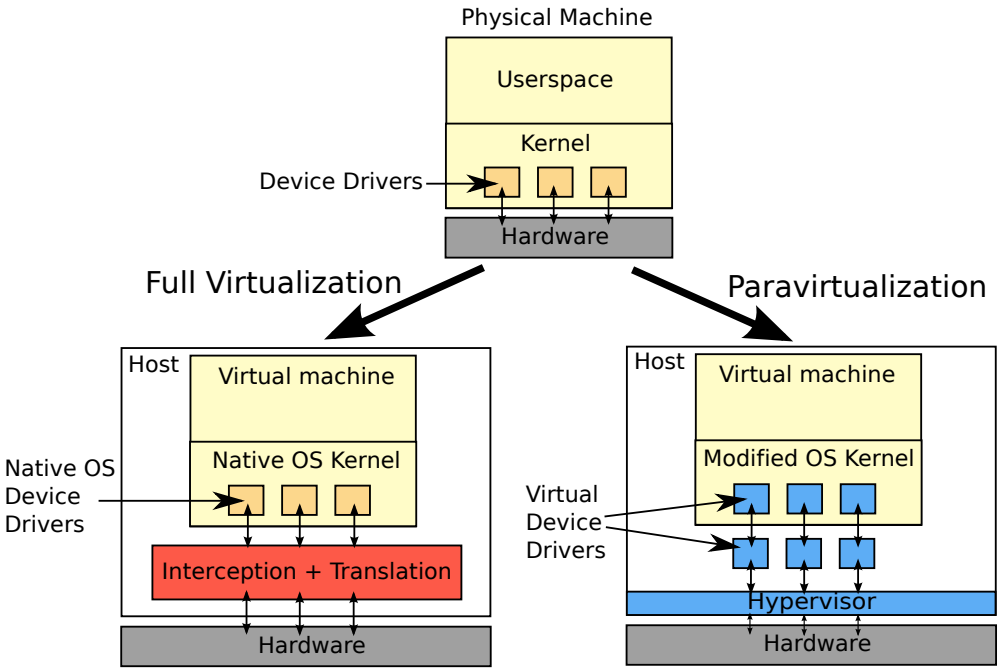



**Figure 1. Full virtualization and paravirtualization.**

between full- and paravirtualization.

The most commonly used paravirtualization solution is presently Xen [3]. A more recent virtualization technology is KVM [15], which initially supported only full virtualization with hardware-assisted CPU virtualization and emulated I/O device drivers. When the virtual machines are network-intensive, an important performance overhead is expected, as emulation is a very costly procedure. With the appearance of the *virtio* [16] tools proposing a set of virtual I/O drivers, KVM became able to perform paravirtualization as well, to leverage the I/O performance. In the following, to shed light on these technologies, Xen's and KVM's network mechanisms are described in detail, and their performance is evaluated.

## III. Network Virtualization

This section details the mechanisms used by paravirtualization techniques to virtualize the network devices. In Xen [3], each virtual machine resides in a so-called guest domain. Among the guest domains, only a driver domain, domain 0 (dom0) by default, has direct access to the hardware. All the other domains, called domain U (domU) for "Unprivileged", need to use virtual interfaces. A virtual-machine monitor, called hypervisor, manages all the virtual machines and the access to the hardware devices. In KVM, the virtual machines are created as device nodes, and the host system operates as the hypervisor. It runs two KVM kernel modules, a modified QEMU [12] module performing hardware-device emulation, and a processor-dependent module to manage hardware virtualization.

### A. Data path in Xen

The virtual machines in Xen access the network hardware through the virtualization layer. Each domU has a virtual interface for each physical network interface of the physical machine. This virtual interface is accessed via a *split-device driver* composed of two parts, the front-end driver in domU and the back-end driver in dom0 [17]. Figure 2 illustrates the path followed by the network packets emitted on a virtual machine residing inside a domU. The memory page where a packet resides in the domU kernel is either mapped to dom0 or the packet is copied to a segment of shared memory by the Xen hypervisor from where it is transmitted to dom0. Inside dom0, packets are bridged (path 1) or routed (path 2) between the virtual interfaces and the physical ones. The reception of packets on a domU is similar. To receive a packet, domU gives a grant to dom0 so that dom0 can access the grant page and copy the packet to domU's kernel space [18].

The additional path a packet has to go through due to virtualization is marked by the dashed line. A significant processing and latency overhead can be expected due to the additional copy to the shared memory between domU and dom0. Moreover, the multiplexing and demultiplexing

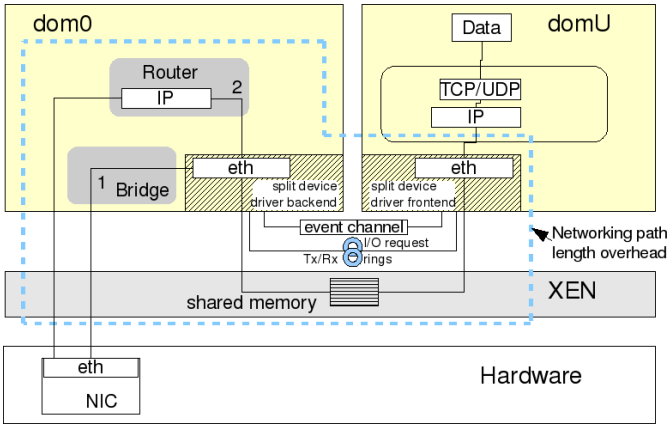



**Figure 2. Path of a network packet with Xen, from a domU to the NIC.**

of the packets in dom0 can be expensive.

## B. Data path in KVM

Similar to Xen, KVM uses a virtual driver in paravirtualization mode. This virtual driver is part of the virtio [16] I/O drivers used within the Linux kernel. Virtual machine kernels also use a front-end driver with particular code to communicate with a back-end driver which interfaces with the KVM module inside the Linux kernel [19]. This architecture is represented on Figure 3. To communicate

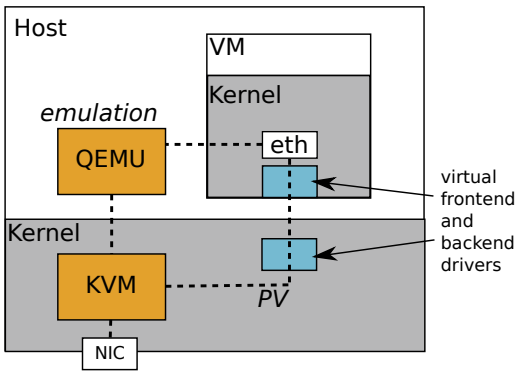



**Figure 3. Path of a network packet in KVM using paravirtualization (PV) or full virtualization (emulation).**

between front-end and back-end, ring-buffers are used for the implementation of net channels, like in the Xen virtual driver.
In its original full-virtualization mode, KVM emulates I/O drivers using a QEMU [12] module in the userspace.

## C. Routers data plane virtualization

The described virtualization techniques can be used for fully (i.e., control-plane and data plane) virtualized software routers, to implement virtual network infrastructures as described before. In particular, we used Xen, to implement such a virtual router, due to its more promising performance it showed in our end-host experiments, compared to KVM. Figure 4 shows an example of such an architecture with software routers uploaded (control- and data-path) into virtual machines to create virtual routers. In this example, two virtual routers share the resources (NICs, CPU, memory) of a single physical server. The governing principle inside such virtual routers

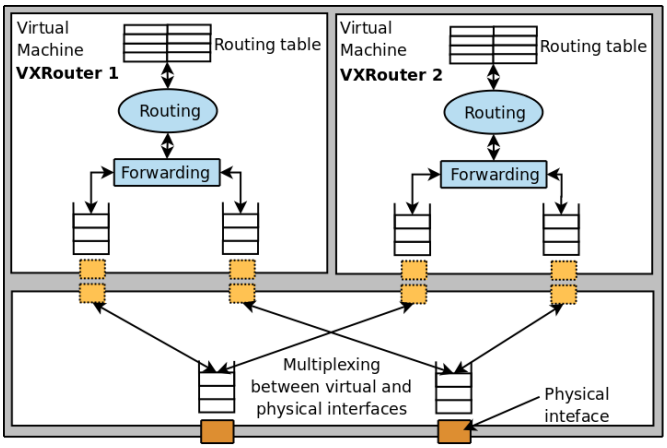



**Figure 4. Machine with 2 virtual routers sharing the 2 NICs.**

is the same as inside a standard software router, except that the virtual machines do not have direct access to the physical hardware interfaces. The packets are forwarded between the virtual or emulated interfaces and the corresponding physical interfaces thanks to a multiplexing and demultiplexing mechanism. This is implemented in an intermediate layer located between the hardware and the virtual machines, which corresponds to the hypervisor and the host's operating-system kernel in the virtualization techniques described before. For example in Xen, this layer corresponds to the hypervisor and the driver domain or dom0. As a consequence, there is additional computing in this model whose impact on the network performance needs to be analyzed.

## D. Performance problem statement

An efficient usage of virtual machines for networking requires a certain number of non-functional properties like *efficiency*, *fairness* in resource sharing and *predictability*

of performance. We define and evaluate these properties using the following metrics. Given $N \in \mathbb{N}$ the number of virtual machines inside the considered physical machine, the defined metrics are listed in Table I.

| | |
|---|---|
| $R_i$ | Throughput of virtual machine $i$, $i \in [1, N]$ |
| $R_{aggregate}$ | $\sum_{i=1}^{N} R_i$: aggregate throughput |
| $R_{aggregate}/N$ | Effective fair share of the rate |
| $C_i$ | CPU cost on virtual machine $i$, $i \in [1, N]$ |
| $C_{aggregate}$ | $\sum_{i=1}^{N} C_i$: total CPU cost |
| $L_i$ | Latency on virtual machine $i$, $i \in [1, N]$ |
| $R_{classical(T/R)}$ $R_{classical(F)}$ | Throughputs for sending/receiving and forwarding on classical Linux without virtualization |
| $C_{classical(T)}$ $C_{classical(R)}$ $C_{classical(F)}$ | CPU costs of sending, receiving and forwarding on a classical software router |
| $L_{classical(F)}$ | Latency on a classical Linux router |

**Table I. Metrics.**

For our proposal, we define the efficiency in terms of throughput by (1)

$$E_{throughput} = \frac{R_{aggregate}}{R_{classical}} \quad (1).$$

The fairness of the inter-virtual machine resource sharing is derived from the classical Jain index[20], defined by (2).

$$Fairness(x) = \frac{\left[\sum_{i=1}^{n} x_i\right]^2}{n \times \sum_{i=1}^{n} x_i^2} \quad (2)$$

where $n$ is the number of virtual machines sharing the physical resources and $x_i$ the metric achieved by each virtual machine $i$, for example the throughput or the CPU percentage.
Predictability and scalability are evaluated analyzing the performance according to the number of virtual machines.

## IV. Experiments and Analysis

In this section, we examine the impact of the virtualization layer on the networking performance. In this context, at first, virtual end-host performance is evaluated with Xen. Then it is compared to results obtained on KVM. Finally, a virtual router is implemented with Xen, which showed better performance in virtual networking than KVM. This virtual router's forwarding performance is then evaluated.

### A. Experimental setup

The experiments were all executed on machines reserved on the fully controlled, and reconfigurable French national testbed Grid'5000 [21]. The machines used for the initial experiments on Xen end-hosts were IBM eServers 325, with 2 AMD Opteron 246 CPUs (2.0 GHz/1 MB) with one core each, 2 GB of memory and a 1 Gb/s NIC. Virtual routers were built with Xen on IBM eServers 326m, with 2 AMD Opteron 246 CPUs (2.0GHz/1MB), with one core each, 2 GB of memory and two 1 Gb/s NICs. The latest experiments on KVM were executed on more recent machines provided with hardware virtualization enabled processors. These were Dell PowerEdges 1950 with two dual-core Intel Xeon 5148 LV processors (2.33 GHz) and 8 GB of memory. We tested with Xen that performance was equivalent in these two different hardware configurations. In each experiment, all the machines were located inside one LAN interconnected by a switch.
The software configurations used were Xen 3.1.0 and 3.2.1 with respectively the modified 2.6.18-3 and 2.6.18.8 Linux kernels. Comparative experiments were performed on KVM 84 with the Linux 2.6.29 kernel in the host system as well as in the virtual machine. Measurement tools were *iperf* [22] for the TCP throughput, *netperf* [23] for the UDP rate, *xentop* and *sar* for the CPU utilization, and the classical *ping* utility for latency.

### B. Evaluation of virtual end-hosts

In the following experiments, network performance on virtual end-hosts implemented with Xen 3.1 and Xen 3.2 was evaluated. As some results with Xen 3.1 were not satisfying, dom0 being the bottleneck, a second run of the experiments on Xen 3.1 was performed, allocating more CPU time to dom0 (up to 32 times the part attributed to a domU). This choice was made as dom0 is in charge of forwarding all the network traffic between the physical and the virtual interfaces. This setup will be called *Xen 3.1a*. In addition, the obtained results were compared to KVM virtual end-hosts. KVM was used either in its native full hardware virtualization setup or using lightweight paravirtualization.

*1) Sending performance:* In this first experiment, the TCP sending throughput on 1, 2, 4 and 8 virtual hosts inside one physical machine, as well as the corresponding CPU overhead, were evaluated. The throughput per virtual machine and the aggregate throughput with Xen are represented on Figure 5. In both Xen configurations,

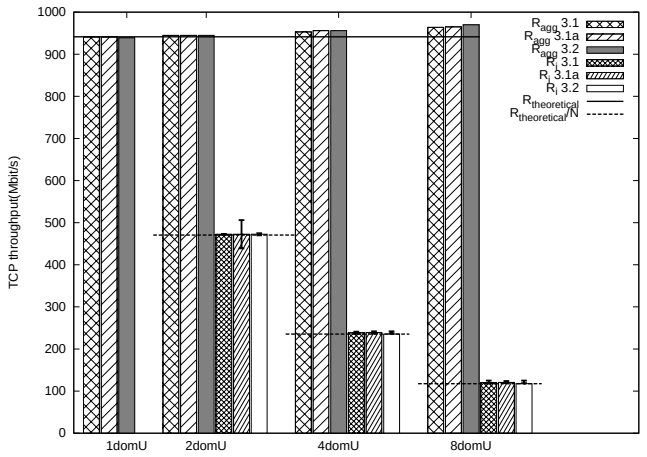



**Figure 5. TCP Sending throughput on respectively 1, 2, 4 or 8 VMs with Xen versions 3.1 and 3.2.**

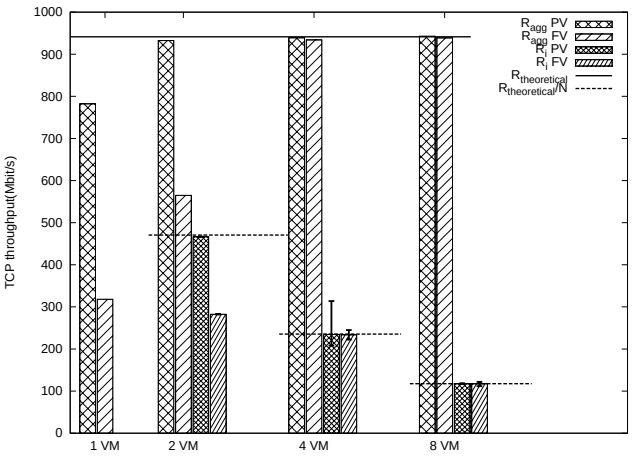



**Figure 7. TCP Sending throughput on 1, 2, 4 or 8 VMs with KVM 84 using paravirtualization (PV) or full virtualization (FV).**

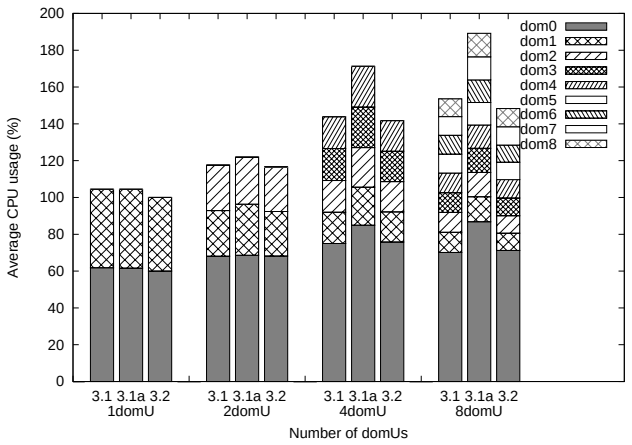



**Figure 6. Average utilization of the two CPUs during TCP sending on 1, 2, 4 or 8 VMs with Xen.**

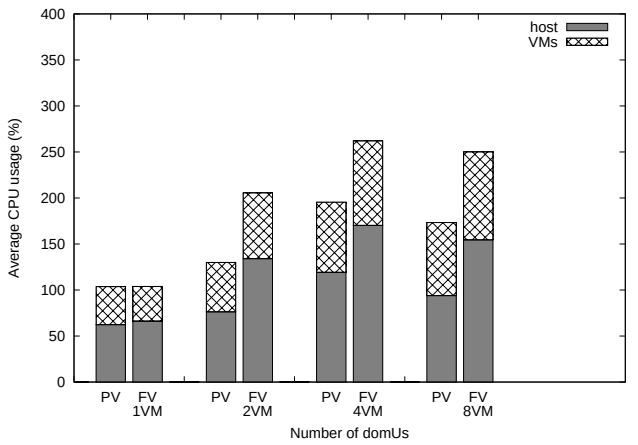



**Figure 8. Average utilization of the four CPU cores during TCP sending on 1, 2, 4 or 8 VMs with KVM.**

3.1 and 3.2, performance was close to classical Linux throughput $R_{classical(T/R)} = 938\ Mb/s$. In the 3.1a and 3.2 setups, the aggregated throughput obtained by all the virtual machines barely reached more throughput than on 3.1. We conclude that in the three cases (3.1, 3.1a, 3.2), the system is efficient and predictable, in terms of sending throughput. Indeed, the throughput per virtual machine corresponds to the fair share of the available bandwidth of the link ($R_{theoretical}/N$).

The associated average CPU utilization for each Xen guest domain is represented on Figure 6. For a single domU, around half the processing power of the two CPUs was used in the three setups (Xen 3.1, 3.1a and 3.2), whereas on a native Linux system without virtualization, we measured that only $C_{classical(E)} = 64\%$ of the two CPUs, out of $200\%$, was in use running the same network benchmark. In the experiment with 8 domUs, the CPUs were used at over 140%. The overall CPU overhead did not differ much between 3.1 and 3.2 setups. However, by increasing dom0's CPU weight (setup 3.1a), the overall CPU cost also increased while leveraging the throughput insignificantly. We notice that even though virtualization introduced a processing overhead, two processors like the ones used in these experiments achieved a throughput equivalent to the maximum theoretical throughput on 8 concurrent virtual machines, sending TCP flows of default maximum-sized packets on a 1 Gb/s link. Here,

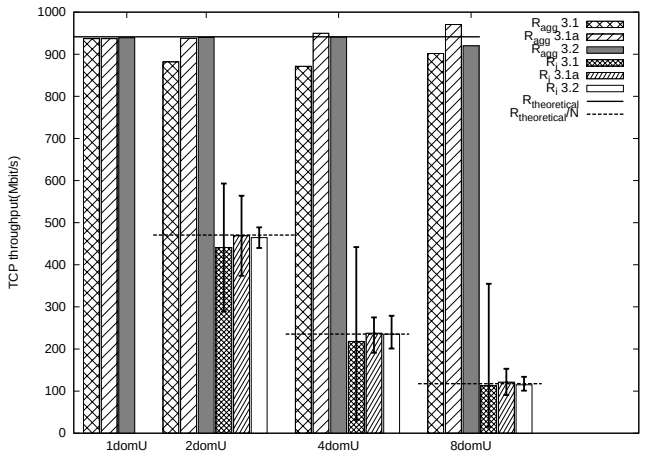



**Figure 9. TCP Receiving throughput on respectively 1, 2, 4 or 8 VMs with Xen versions 3.1 and 3.2.**

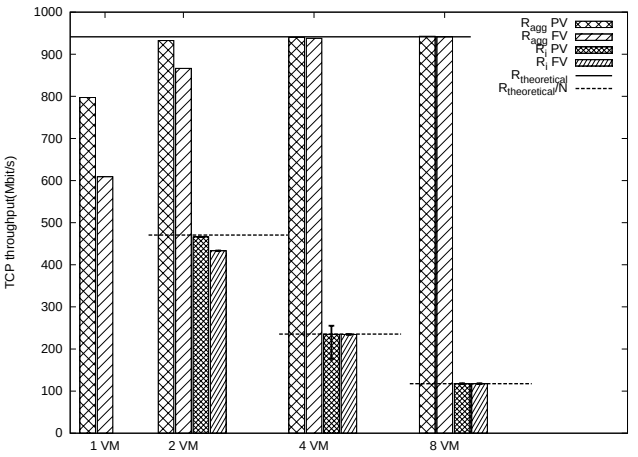



**Figure 11. TCP Receiving throughput on 1, 2, 4 or 8 VMs with KVM 84 using paravirtualization (PV) and full virtualization (FV).**

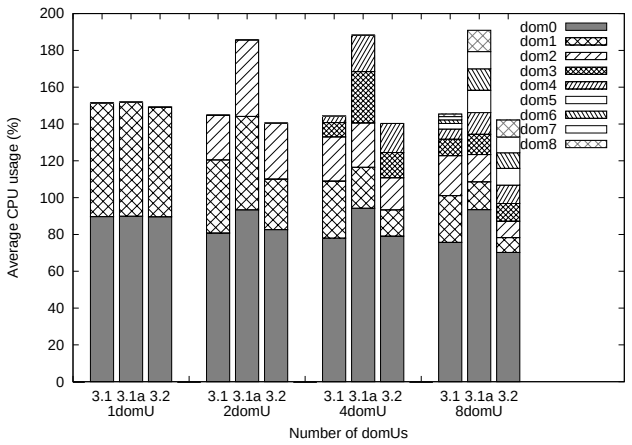



**Figure 10. Average utilization of the two CPUs during TCP receiving on 1, 2, 4 or 8 VMs.**

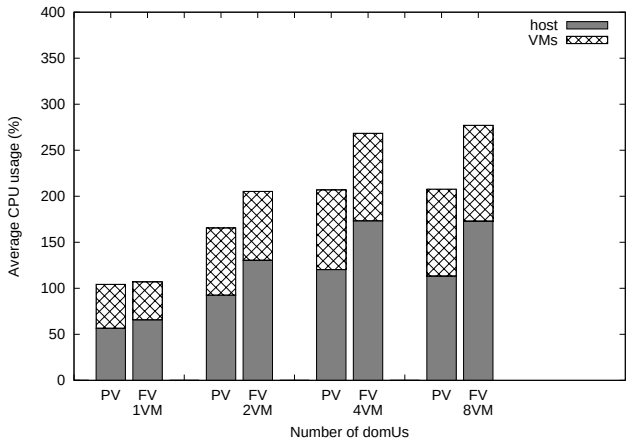



**Figure 12. Average utilization of the four CPU cores during TCP receiving on 1, 2, 4 or 8 VMs with KVM.**

the fairness index was close to 1, bandwidth and CPU time were fairly shared between the different domUs.

The same experiment was executed on KVM virtual machines. KVM was used in two different configurations: paravirtualization (PV) using virtual drivers from *virtio*, and native full hardware virtualization (FV) where network drivers are emulated. The results in terms of throughput are represented on Figure 7. The first point to notice is that paravirtualization clearly outperformed full hardware virtualization with network driver emulation. For a single virtual machine, KVM with full virtualization reached only around 30% of the native Linux throughput. In this case, the bottleneck was clearly the CPU utilization. Figure 8 shows that the entire CPU core assigned to the evaluated virtual machine was used. In the case of several virtual machines, where each one was assigned a different CPU core, the throughput increased, as more CPU cores were involved.

*2) Receiving performance:* In this experiment, the TCP receiving throughput on 1, 2, 4 and 8 concurrent virtual machines and the corresponding processing overhead were evaluated. Figure 9 represents the results of this experiment in terms of TCP throughput per domU and aggregate throughput with Xen. We notice that the aggregate throughput decreased slightly according to the number of virtual machines on Xen 3.1. It reached only 882 Mb/s on a single domU and only 900 Mb/s on a set

of 8 concurrent domUs, which corresponds to about 96% of throughput $R_{classical(T/R)} = 938\ Mb/s$ on a classical Linux system. Efficiency $E_{throughput}$ varied between 0.96 for 8 domUs and 0.94 for a single domU. By changing the scheduler parameters (Xen 3.1a), we managed to improve the aggregate throughput to reach about 970 Mb/s on 8 virtual machines. Also, setup 3.1a improved fairness in Xen 3.1, but increased CPU consumption, leading to a trade-off between determinism and scalability. Xen 3.2 showed similar trends, leveraging throughput and removing unfairness (improving the event channel mechanism [24]). Moreover, CPU utilization decreased slightly in Xen 3.2, while being more efficient than Xen 3.1, achieving even better throughput. We conclude that important improvements have been implemented in Xen 3.2 to decrease the excessive dom0 CPU overhead.
[24]. This problem was fixed in Xen 3.2. Providing dom0 with more CPU time simply, as it was done in the 3.1a setup, allowed also to improve fairness in Xen 3.1 by giving dom0 enough time to treat all the events before the scheduler ran out of credits and started switching unnecessarily between dom0 and domUs. The resulting fair resource sharing made performance much more predictable. The measured aggregate receiving throughput in Xen 3.2 was more similar to the Xen 3.1a results with the modified scheduler parameters. The throughput increased by about 6% compared to the default 3.1 version. Figure 10 gives the CPU time distribution among the guest domains. The total CPU cost of the system varied between 140% and 150% in the default Xen 3.1 and 3.2 versions, which represents an important overhead compared to a Linux system without virtualization, where a network reception takes $C_{classical(R)} = 48\%$ with the same benchmark. We notice that on the default Xen 3.1, the efficiency in terms of throughput decreased to around $E_{throughput} = 0.91$, while the available CPU time was not entirely consumed. Also, the distribution of the CPU time consumption among the domUs followed the same unfairness pattern than for the throughput. This shows that the virtual machine scheduler on the CPU looses efficiency when stressed with networking. The fairness index decreased until only 0.46 on 8 concurrent domUs on Xen 3.1 because of the described scheduling problem.

In comparison, KVM using the virtualized network driver achieved very similar results to sending: Using a single CPU core, as in the case of one virtual machine, is not enough to achieve maximal Linux throughput as shows Figure 11. Figure 12 shows that the CPU overhead for receiving is slightly more important than for sending using virtio paravirtualization driver. This is similar to the results obtained with Xen 3.2, which nevertheless used between 10% and 30% less processing to achieve the same throughput. In the case of full virtualization, KVM's receiving mechanism is more efficient than its sending mechanism, but it still does not reach Xen's performance, needing three of the available CPU cores to achieve maximum Linux throughput.

## C. Evaluation of virtual routers

To figure out the forwarding performance of virtual routers with 2 NICs, the UDP receiving throughput over virtual machines sending maximum-sized packets at maximum link speed over the virtual routers, and the TCP throughput was measured. Further the latency over virtual routers was measured. For this experiment, only Xen 3.2 was used, which was the best performing technology in the previous experiments. The results were obtained on Xen 3.2 in its default configuration and with increased weight parameter for dom0 in CPU scheduling (32 times the part attributed to a domU). We call this setup *Xen 3.2a* in the following.

*1) Forwarding performance:* To determine the performance of virtual routers, UDP traffic was generated with either maximum- (1500 bytes) or minimum- (64 bytes) sized packets over one or several virtual routers (from 1 to 8) sharing a single physical machine. All the flows were sent at the maximum rate from distinct physical machines to avoid bias. Then, end-to-end TCP throughput was also evaluated.
Figures 13 show the obtained UDP bit rate with maximum-sized packets and the TCP throughput. The corresponding CPU cost is represented in Figure 14. Table II details the UDP packet rates and the loss rates per domU with maximum- and minimum-sized packets. With UDP, the

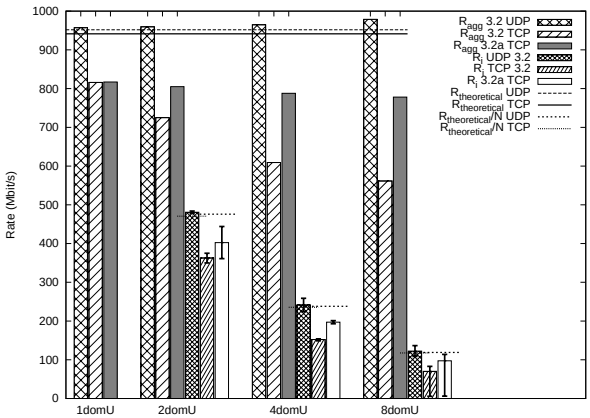



**Figure 13. Receiver side throughput over 1, 2, 4 or 8 virtual routers with Xen 3.2 forwarding 1500-byte packets.**

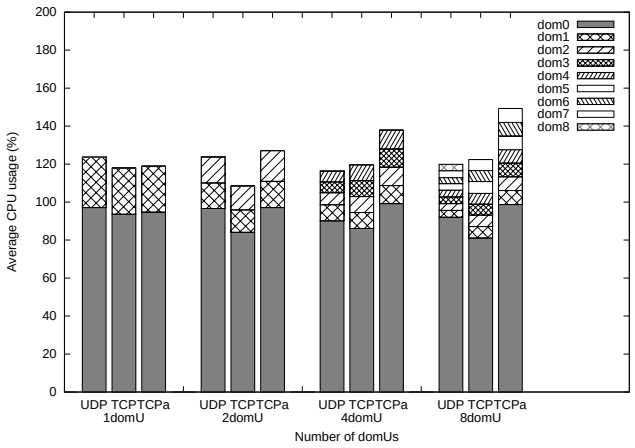



**Figure 14. CPU cost of 1, 2, 4 or 8 virtual routers with Xen 3.2 forwarding 1500-byte packets.**

packet loss rate with maximum-sized packets on each virtual machine corresponded to $1 - R_{theoretical}/(N \times R_{theoretical})$.
The bandwidth was fairly shared between the virtual routers. The results showed efficiency, the throughput corresponded to the value obtained on a classical Linux router $R_{classical(F)} = 957\ Mb/s$. The aggregate UDP throughput was in some cases a bit higher than the theoretical value due to little variation in the start times of the different flows. Resource sharing was fair: in this case, performance is predictable. With maximum-sized packets, dom0 used an entire CPU, forwarding at maximum rate with UDP and only 80% of the maximum throughput with TCP, having an efficiency of $E_{throughput} = 0.80$. With minimum-sized packets on 4 or 8 virtual routers, dom0 became overloaded, not being able to forward on all virtual routers anymore.
Regarding the processing of the router, dom0 used much more CPU resources than the domUs, compared to the simple sending or receiving scenario. This is due to the fact that it has to forward the traffic of twice as many virtual interfaces than before (16 in the case of 8 virtual routers). In the case of UDP and TCP with the modified scheduling parameters (named TCPa), the usage of domU's CPU was pushed to its maximum. It used an entire CPU. Hence, the TCP throughput was obviously limited by dom0's processing limitation. In the case of TCP forwarding with the default CPU scheduling parameters, dom0 did not get enough processing resources, especially when the number of virtual routers increased, in turn increasing the number of virtual interfaces to treat, which caused the throughput to decrease. With the important overload with 8 virtual routers, the last domU was not even able to forward packets anymore, and thus used no CPU.

| | 1500 byte packets | | 64 byte packets | |
|---|---|---|---|---|
| | pps/VM | loss/VM | pps/VM | loss/VM |
| 1 VM | 81284 | 0 % | 109685 | 60 % |
| 2 VM | 40841 | 50 % | 12052 | 96 % |
| 4 VM | 20486 | 75 % | / | / |
| 8 VM | 10393 | 87 % | / | / |
| Linux | 81277 | 0.00 % | 356494 | 0.06 % |

**Table II. Average UDP packet-forwarding rate and loss rate per domU.**

With TCP, the throughput throttled down, especially with an increasing number of virtual routers. This might be related to an increasing latency, discussed in the next paragraph.

*2) Virtual router latency:* In this experiment, the latency on one virtual router (VR) was measured, while concurrent virtual routers (1, 3 or 7) sharing the same physical machine were either idle or stressed forwarding maximum-rate TCP flows. Table III represents the results in both cases. The latency over a virtual router sharing

| Latency(ms) | Linux | 1 VR | 2 VR | 4 VR | 8 VR |
|---|---|---|---|---|---|
| idle | 0.084 | 0.147 | 0.150 | 0.147 | 0.154 |
| stressed | | | 0.888 | 1.376 | 3.8515 |

**Table III. Latency over one VR among 1, 2, 4 or 8 VRs idle or stressed with TCP forwarding.**

the physical machine with other idle virtual routers was about 0.150 ms, no matter the number of virtual routers, which was almost the double of the latency on a classical Linux router $L_{classical(F)} = 0.084ms$. In the case of a stressed system, the latency on the considered virtual router increased with the number of concurrent virtual routers forwarding maximum throughput TCP flows. The average latency reached nearly 4 ms on a virtual router sharing the physical machines with 7 virtual routers forwarding TCP flows. The more virtual machines asking for the scheduler, the more the latency on the virtual router increased. For TCP, this can lead to timeouts. In this case, frequent retransmissions throttle the throughput down.

## V. Related work

The performance of virtual packet transmission in Xen is a crucial subject and was treated in several papers. Table IV lists the main network issues of the successive Xen versions, discussed in this section. On Xen 2.0, Menon et al. [25] measured default transmit and receive TCP throughput on a domU below 1000 Mb/s using four 1-Gigabit NICs, which is much less than the throughput we measured on Xen 3.2 (940 Mb/s on a single NIC). The

| | Key problem | Key improvement |
|---|---|---|
| Xen 2.0 | End-host throughput (≅25% of CL)[25] | VNIC offloading I/O channel modif. |
| Xen 3.0 | Forwarding rate <25% of CL, CPU bottleneck[26] | SMP load balancing CPU scheduler |
| Xen 3.1 | Unfairness | Event channel modif.[24] |
| Xen 3.2 | Small packets I/O (memory bottleneck)[27] | |
| KVM | CPU overhead | virtio virtual driver |

**Table IV. Comparison of Xen versions and KVM.**

authors improved Xen's networking performance by modifying the virtual network interfaces to include hardware NIC features and optimize the I/O channel between dom0 and domU. After optimization, they obtained results for transmissions similar to what we obtained on Xen 3.2: 3310 Mb/s on four NICs which corresponds to around 830 Mb/s per NIC, but still less for receptions (only 970 Mb/s on four NICs) which corresponds to only 26% of Xen 3.2's receiving throughput.

A study about scheduling on Xen 3 unstable (changeset 15080) [24] confirms our result, with Xen 3.1, of the unfair bandwidth distribution among the domUs in the default configuration, with the credit scheduler [28]. The authors measured an aggregate throughput of less then 800 Mb/s on 7 domUs, varying from less than 25 Mb/s to around 195 Mb/s per domU. They proposed event-channel improvements which enhanced the fairness in the sharing of the bandwidth between the virtual machines, to vary only about $\pm 25$ Mb/s on the different domUs. We noticed this improved fairness in the new Xen 3.2 version.

McIlroy and Sventek in [29] proposed the virtual router concept to separate traffic into virtual forwarding planes. In this case, data-plane virtualization offers QoS with individual packet forwarding. An evaluation of Xen 3.0 for routers [30] shows that the aggregate forwarding throughput on two to six domUs reaches less than 25% of what is achievable on classical Linux for 64-byte frames.

In a recent paper [27], the authors showed that memory access time is the main system bottleneck with Xen. They finally proposed a system to map forwarding paths to CPU cores so that as many packets as possible can be kept in the closest CPU cache to limit costly memory accesses [31]. This system is an interesting step to improve the performance of forwarding in software, but there is no real data-plane virtualization as packet data remains in dom0 and only routing is performed in domU. It is an interesting trade-off between performance and the level of virtualization in software virtual routers.

In parallel to these evaluations and optimizations of Xen, KVM appeared as a new full virtualization solution. To leverage it's low I/O performance, due to the emulation of the drivers, paravirtualization was also included within the *virtio* drivers. Nevertheless, as our results showed, Xen outperforms KVM with virtio when it comes to network virtualization. While virtio is a sort of trade-off between genericity and performance, Xen's virtual drivers are tailored to the Xen technology and, as described before, have gone through several optimizations. This can explain its better performance to date.

Another very recent software network virtualization solution is Crossbow [32], which appeared on OpenSolaris. Crossbow allows to build virtual NICs using so-called virtualization lanes. It works with hardware-virtualization-enabled NICs, and packets are directly classified on layer 2 into the right virtualization lane. In the absence of hardware virtualization of the NICs, virtual lanes are implemented entirely in software, but giving no guarantees on the fairness in the resources sharing. In both cases, Crossbow is a solution to virtualize the data plane and is an interesting technology to study as future work.

## VI. Conclusion and perspectives

In this article, we evaluated the network performance of virtual end-hosts and a virtual router we designed and implemented with Xen. The results were also compared to KVM virtualization technology. Virtualization mechanisms like additional copy and I/O scheduling of virtual machines sharing the physical devices are costly. Nevertheless, our results show that virtualizing the data plane by forwarding packets inside the virtual machines becomes a more and more promising approach with the successive versions of Xen, improving those mechanisms. We show that end-host throughput improved in Xen 3.2 compared to 3.1. Also, previous fairness issues have been corrected. Xen, with its virtual network drivers, outperforms KVM, offering higher network transmission rates, as it requires significantly less processing power. Virtual routers built with Xen act similarly to classical Linux routers, while forwarding big packets. The evaluation of successive Xen versions shows that the technology is constantly improving and gives a promising perspective to network virtualization. However, our results showed an important overhead, due to additional processing needed, since packets travel through virtual drivers. For this reason, the best solution to leverage performance would probably be to use virtualization-enabled network interfaces. Our next goal is to evaluate the impact of executing large-scale applications on concurrent virtual network topologies with virtual routers performing network control in terms of routing and bandwidth sharing.

## Acknowledgement

This work has been funded by the French ministry of Education and Research and the ANR, INRIA, and CNRS, via ACI GRID's Grid'5000 project, ANR HIPCAL grant and INRIA Aladdin ADT. We thank Olivier Mornard for his assistance in setting up the testbed.

## References

[1] F. Anhalt and P. Vicat-Blanc Primet, "Analysis and experimental evaluation of data plane virtualization with Xen," in *ICNS 09 : International Conference on Networking and Services*, (Valencia, Spain), Apr. 2009.

[2] G. J. Popek and R. P. Goldberg, "Formal requirements for virtualizable third generation architectures," in *SOSP '73: Proceedings of the fourth ACM symposium on Operating system principles*, (New York, NY, USA), p. 121, ACM, 1973.

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, (New York, NY, USA), pp. 164–177, ACM, 2003.

[4] J. Sugerman, G. Venkitachalam, and B.-H. Lim, "Virtualizing I/O devices on VMware workstation's hosted virtual machine monitor," in *Proc. 2001 Usenix Annual Technical Conference*, pp. 1–14, Usenix Assoc., 2001.

[5] J. Varia, "Cloud architectures," in *NSF Data-Intensive Scalable Computing Workshop*, 2008.

[6] http://www.3tera.com/.

[7] http://www.microsoft.com/azure.

[8] N. Feamster, L. Gao, and J. Rexford, "How to lease the internet in your spare time," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 1, pp. 61–64, 2007.

[9] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, "In vini veritas: realistic and controlled network experimentation," in *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 3–14, ACM, 2006.

[10] S. Bhatia, M. Motiwala, W. Muhlbauer, V. Valancius, A. Bavier, N. Feamster, L. Peterson, and J. Rexford, "Hosting Virtual Networks on Commodity Hardware," tech. rep., Georgia Tech Computer Science Technical Report GT-CS-07-10, Jan 2008.

[11] "Linux VServers Project." http://linux-vserver.org.

[12] F. Bellard, "Qemu, a fast and portable dynamic translator," in *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, (Berkeley, CA, USA), pp. 41–41, USENIX Association, 2005.

[13] G. Neiger, A. Santoni, F. Leung, D. Rodgers, and R. Uhlig, "Intel virtualization technology: Hardware support for efficient processor virtualization," tech. rep., Intel Technology Journal, 2006.

[14] "AMD-V Nested Paging," tech. rep., Advance Micro Devices, Inc., July 2008.

[15] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "kvm: the Linux Virtual Machine Monitor," in *Linux Symposium*, 2007.

[16] R. Russell, "virtio: towards a de-facto standard for virtual i/o devices," *SIGOPS Oper. Syst. Rev.*, vol. 42, no. 5, pp. 95–103, 2008.

[17] D. Chisnall, *The Definitive Guide to the Xen Hypervisor*. Prentice Hall, 2007.

[18] J. R. Santos, G. J. Janakiraman, Y. Turner, and I. Pratt, "Netchannel 2: Optimizing Network Performance," in *Xen summit*, 2007.

[19] D. Laor, "KVM Para-Virtualized Guest Drivers." KVM Forum 2007, Aug. 2007.

[20] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *ACM Transaction on Computer Systems*, Sept. 1984.

[21] F. Cappello, P. Primet et al., "Grid'5000: A large scale and highly reconfigurable grid experimental testbed," in *GRID '05: Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing*, pp. 99–106, IEEE Computer Society, 2005.

[22] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "iperf : testing the limits of your network," *http://dast.nlanr.net/Projects/Iperf*.

[23] http://www.netperf.org/netperf/.

[24] D. Ongaro, A. L. Cox, and S. Rixner, "Scheduling i/o in virtual machine monitors," in *VEE '08: Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, pp. 1–10, ACM, 2008.

[25] A. Menon, A. L. Cox, and W. Zwaenepoel, "Optimizing network virtualization in xen," in *ATEC '06: Proceedings of the annual conference on USENIX '06 Annual Technical Conference*, pp. 2–2, USENIX Association, 2006.

[26] F. Anhalt and P. Vicat-Blanc Primet, "Analysis and evaluation of a XEN based virtual router," Tech. Rep. 6658, INRIA Rhône Alpes, Sep 2008.

[27] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy, "Fairness issues in software virtual routers," in *PRESTO '08*, pp. 33–38, ACM, 2008.

[28] http://wiki.xensource.com/xenwiki/ CreditScheduler.

[29] R. McIlroy and J. Sventek, "Resource virtualisation of network routers," in *HPSR 06: 2006 Workshop on High Performance Switching and Routing*, June 2006.

[30] N. Egi et al., “Evaluating Xen for Router Virtualization,” in *ICCCN*, pp. 1256–1261, 2007.

[31] N. Egi, A. Greenhalgh, M. Hoerdt, F. Huici, P. Papadimitriou, M. Handley, and L. Mathy, “A Platform for High Performance and Flexible Virtual Routers on Commodity Hardware.” SIGCOMM 2009 poster session, Aug. 2009.

[32] S. Tripathi, N. Droux, T. Srinivasan, and K. Belgaied, “Crossbow: from hardware virtualized nics to virtualized networks,” in *VISA ’09: Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures*, (New York, NY, USA), pp. 53–62, ACM, 2009.

# Self-Organizing ZigBee Network and Bayesian Filter Based Patient Localization Approaches for Disaster Management

Ashok-Kumar Chandra-Sekaran, Christophe Kunze
FZI Research Center for Information Technology, Karlsruhe, Germany
e-mail: chandra@fzi.de, kunze@fzi.de

Klaus D. Müller-Glaser, Wilhelm Stork
Institute for Information Processing Technology (ITIV), Karlsruhe Institute of Technology (KIT), Germany.
e-mail: kmg@itiv.uka.de, stork@itiv.uka.de

***Abstract*— A self-organizing, scalable, heterogeneous and location aware WSN architecture called Disaster Aid Network (DAN) for assisting the responders to provide efficient emergency response was proposed in our previous work. The two main aspects of DAN are the communication and localization aspect. As part of the DAN communication aspect, in this paper we have undergone empirical investigations to identify the suitability of ZigBee's 2.4 GHz operation for DAN. A new self-configuring mechanism for patient data access by the doctor at the disaster site is also proposed. As part of the DAN localization aspect, in this paper we have implemented two new Bayesian filter based algorithms called Improved Range-Based Monte Carlo Patient Localization and Range-Based Unscented Kalman Filter Patient Localization for real time localization of large number of patients at the disaster site. The close-to-reality simulations of both these algorithms is done using a disaster management mobility model to identify their suitability for patient tracking. The new localization solution in tandem with the emergency response system shall facilitate efficient logistic support at the disaster site.**

***Keywords- Emergency response; ZigBee; self-configuration; patient localization; Bayesian filter based algorithms; close to reality simulation***

## I. INTRODUCTION

Wireless Sensor Networks (WSN) offer great opportunities and numerous applications are imaginable. But they also impose some new challenges that have to be dealt with. The main WSN challenges that researchers and developers are currently dealing with include heterogeneous network, scalability, self-organisation, self-sufficient operation, multi-hop communication, ad-hoc networks and localization. Self-Organising sensor network are WSN built from sensor nodes that may spontaneously create impromptu network, assemble the network themselves, dynamically adapt to device failure and degradation, manage movement of sensor nodes, and react to changes in task and network requirements. Self-organization can be classified into four aspects as follows.

- Self-configuration: The ability of WSN to automatically and seamlessly configure sensor nodes.
- Self-healing: The ability of WSN to automatically detect, diagnose, and repair localized software and hardware problems.
- Self-optimization: The ability of WSN to continually seek opportunities to improve their own performance and efficiency.
- Self-protection: The ability of WSN to automatically defend themselves against malicious attacks or cascading failures. A WSN should use early warning to anticipate and prevent system wide failures.

Some of the short range wireless communication standard based technologies that can be considered for WSN are Bluetooth, ZigBee, RFID, etc. ZigBee is a standard developed by the ZigBee Alliance that defines a set of communication protocols for low-data-rate short-range wireless networking based on the Open System Interconnect (OSI) basic reference model [20]. Its goal is to provide the means for low-cost implementation of low-data-rate wireless networks with ultra-low power consumption. The ZigBee standard distinguishes between three different device roles for the nodes of the network namely the coordinator, router and end device. It supports star, mesh and tree topology. *RFID* (Radio Frequency Identification) is a technology for contactless and automated identification. It joins other methods for identification like the bar code. However, the advantage of RFID is that it doesn't require a visual contact. The most familiar form of RFID is the RFID tag and a reader.

The potential problems faced in the aftermaths of a disaster are: response capabilities of the local jurisdiction may be insufficient, large-scale evacuations from the disaster site, complications in implementing evacuation management strategy, disruption of critical infrastructure (energy, transportation, telecommunications, etc.), tens of thousands of casualties, response activities must begin without a detailed situation and critical assessment as its time consuming to obtain an initial common operating picture [21].

A new emergency response system based on a Disaster Aid Network (DAN) was proposed by us to improve emergency response [3]. The two main aspects of DAN are the communication and localization aspect.

In the DAN communication aspect the goal is to develop a scalable and robust communication system that supports low power and self-organizing mechanisms. In this paper the focus is on the self-organizing mechanisms for the DAN communication aspect.

In the DAN localization aspect the goal is to track the patients at the disaster site. The task of tracking a patient [18] can be split into range estimation and position estimation. In

this paper we focus on the position estimation part. We have developed two new Bayesian filter based position estimation algorithms and compared their performance to find their suitability for optimal patient tracking. The position estimation algorithm in tandem with the emergency response system shall facilitate efficient logistics at the disaster site by providing real time information about the patients' locations to the On-site Organizational Chief (OOC).

The paper is organized as follows: Section II explains the the Disaster Aid Network system. Section III details the empirical investigations for ZigBee 2.4 GHz operation. Section IV describes the self organization techniques for DAN. Section V mentions the state of the art of localization in WSN. Section VI explains our patient localization method and the two algorithms. Section VII shows the simulation results of these algorithms and Section VIII concludes the paper.

## II. DISASTER AID NETWORK

We focus on the disaster management strategy followed in Germany [2] called "Mass Casualty events" (MANV) but our system can also be adapted to other disaster management strategies. At the beginning of MANV, the disaster site organization chief designates the disaster site into four care zones as follows: The danger zone where the disaster itself happens, injured deposition zone where the patients are prioritized (triaging), treatment zone and transport zone. The patients are shifted from one zone to another before being evacuated [4].

DAN is a self-organizing, scalable, heterogeneous sensor network (see figure 1) [3] of 30-200 nodes comprising of:

- Patient nodes with electronic triage tag and optional continuous vital sign monitoring. They are also called blind nodes because their positions are unknown and have to be estimated.
- Pseudo anchor nodes are patient nodes whose positions are already estimated.
- Doctor nodes (mobile anchor nodes) are mobile nodes (Tablet PC) whose locations are known.
- The monitor station is a collector node which collects the patients' locations and visualizes them for the onsite organization chief (OOC).
- Static anchor nodes are nodes placed at fixed positions whose locations are already known.
- Server: A server running a database for data collection and aggregation is placed at the management centre (or data acquisition centre).

The specifications that the DAN communication aspect should satisfy are: robustness, scalability and self organizing communication system, support heterogeneous network, low power, the use of standard based technology and support distributed communication. Several standard wireless technologies (ZigBee, WLAN, etc.) were considered and due to the low-power, low data rate properties and the possibility to mesh network hundreds of nodes, the ZigBee standard is chosen for investigation to identify its suitability for DAN.

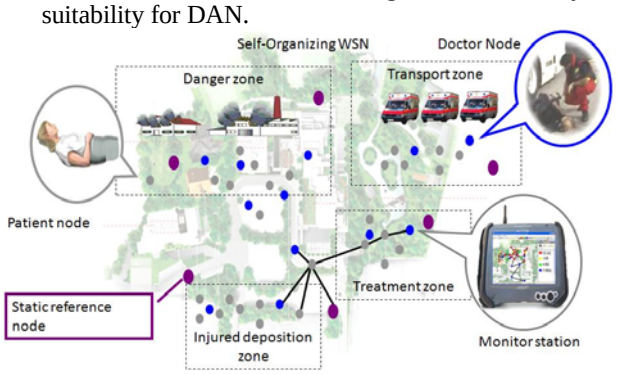



Figure 1. DAN system

The results of our investigations in [22] to find ZigBee's suitability for DAN in terms of power consumption, scalability, mobility and number of routers are summarized as follows

- Power consumption of ZigBee ready sensor nodes imply a battery lifetime of at least a day.
- Scalability: In a static scenario the network scales well up to 150 nodes in a 300m x 300m area. In a mobile scenario, the network does not scale as well as in the static scenario due to the high PLR (packet loss ratio).
- Influence of routers in the network: In a static scenario using more routers doesn't improve the network performance.
- Mobility: The performance of ZigBee network is affected when the nodes are mobile, especially when the mobile node density is high. However, the performance could be improved if the number of routing-capable devices is increased.

These results show that ZigBee is basically suitable for DAN even though detailed investigations will have to be done. In this paper the suitability of ZigBee's 2.4 GHz operation is analyzed and a self-configuring mechanism for DAN is proposed.

## III. ZIGBEE 2.4 GHZ OPERATION

The 2.4 GHz operation of ZigBee is analyzed by considering three main factors which are the node transmission range, RF attenuation and coexistence.

### A. Transmission Range Testing

The maximum transmission range of the ZigBee nodes are measured in different situations using CC2430DB demonstration boards (See figure 2(a)) from Texas Instruments (TI) [24]. The technical data of the CC2430DB is as follows: CC2430 System on Chip with an 8051 core and 2.4 GHz transceiver, a PCB antenna, a data rate of 250 kbps, transmission output power of 0 dbm, and receiver sensitivity of -92dbm. During the measurements the CC2430DB nodes are mounted in a quadrate form of plastic boxes. This simulates the housing case that will be used for

DAN nodes to protect them during the emergency response process.


a) CC2430DB board[1] b) Reference node and triage tag setup

Figure 2. Transmission range testing setup

When both the nodes are placed at the ground level with grass, the maximum transmission range measured is 12 m. This shows that a grass height of even 5 to 10 cm heavily attenuates the signals. When two nodes are elevated at a height of 1.5m using statives' in a line of sight field, the maximum transmission range between them is measured as 205m.

An experiment is undergone to test the transmission range between two nodes with one node acting as a reference node and the other acting as a triage-tag (patient node). In this experiment a CC2430DB node elevated at a height of 1.5 m acts as a reference node and a test person wearing a triage-tag (CC2430DB) are setup in a line of sight environment as shown in figure 2(b). The maximum transmission distance measured between the reference node and the test person (such that a communication is ensured) when the triage-tag is placed at different positions are mentioned in the table 1.

| Setup | Maximum Transmission Distance |
|---|---|
| Test person stands facing the reference node and wears the triage-tag on his chest | 205m |
| Test person stands facing the reference node and wears the triage-tag on his back | 95m |
| Test person lies in the ground and wears the triage-tag on his chest | 105m |
| Test person lies in the ground and wears the triage-tag on his back | 8m |

TABLE I. REFERENCE NODE – TRIAGE TAG MAXIMUM TRANSMISSION DISTANCE MEASUREMENT

### B. Coexistence and RF Attenuation

Since ZigBee operates in the 2.4 GHz ISM band which is also the operating frequency for several other technologies like WLAN and Bluetooth, it's often questionable if ZigBee can coexist. ZigBee network can access up to 16 separate 5MHz channels in the 2.4GHz band, several of which do not overlap all the time with US and European versions of IEEE 802.11 or Wi-Fi. A state of the art analysis related to coexistence indicates that that IEEE 802.15.4 suffers heavy packet loss when interfered by an IEEE 802.11 network specifically if both the technologies are running at overlapping channels. A new feature called 'channel agility' has been introduced in the 2007 version of ZigBee specification. This provides a method for the network to change channels in the event of interference. The network coordinator detects potential interference on a channel and directs the devices on the network to change to a better channel. Kupri's [25] experimental results show that a WLAN (IEEE 802.11 g and n) will not frequently block ZigBee / IEEE 802.15.4 traffic and the 802.15.4 system can co-exist with a WLAN network. Their test results show that even under the most severe 802.11 interference (IEEE 802.11n at 40 MHz bandwidth) by incorporating channel agility, IEEE 802.15.4/ ZigBee is a viable and stable solution for home control applications. They conclude that the implementation of channel agility improves the overall situation and increases the robustness of ZigBee networks.

In practice, a radio signal operating in a disaster site may encounter many objects (fixed, mobile, and transient objects) in its transmission path and undergoes additional attenuation depending on the absorption characteristics of the objects. In [26] they have measured the ZigBee 2.4GHz signal attenuation through the following objects: metal door-6dBm, human body-3dBm, glass wall with metalframe-6dBm, metal door in brick wall- 12dBm.

### C. Summary

The ZigBee 2.4 GHz ranging experiments indicate that during the deployment of nodes in DAN the reference nodes should be elevated at a height of 1.5m for a better transmission range and the maximum distance achievable between two reference nodes in LOS is 205m. The reference nodes can also be provided with range-extenders to increase their range and thereby reduce the number of infrastructure nodes. The maximum transmission distance measurements between a reference node and triage-tag shows that 2.4 GHz ZigBee is suitable for DAN except for the condition when the injured person lies in the ground and wears the triage-tag on his back, wherein the signal strength is strongly attenuated due to the influence of the human body.

However, the empirical investigations of the following topics will be part of future work: coexistence of ZigBee with other ISM band technologies, the effect of RF signal attenuation by obstacles at the 2.4 GHz operation, effect of antenna propoerties and verification of maximum data rate to identify the extent of application data that can be sent.

## IV. SELF ORGANIZATION OF DAN

Self-organization procedures associated with the communication network are necessary for reliable system operation. In DAN during the initial setup, the nodes (coordinator, static anchor nodes, and doctor nodes) should automatically form a network. Also during the emergency response process new patient nodes appear and leave the site (after evacuation). So DAN should automatically reconfigure nodes in the order of 200. The self-configuration characteristics [19] of ZigBee supports automatic

[1] Source: graphic from [24]

establishment of network and association of new nodes joining the network.

In DAN when a mesh network is established, a routing path has to be formed such that the patient nodes can send their data via routers to the server at the management centre of the disaster site. If any node in this route becomes faulty or if the patient nodes move, the routing paths have to be adapted dynamically such that the patient data is relayed in an optimal way to the server. The self-healing characteristics of ZigBee mesh networking [19] can optimally route, as the nodes move and can select alternative route if any node in the routing path stops functioning.

Even though the self-configuring and self-healing mechanisms of ZigBee are useful for DAN, they are not sufficient and new self-organization mechanisms to satisfy DAN communication aspect functionalities have to be developed. In the next subsection one such new self-configuring mechanism for seamless patient data access by the doctor node at the disaster site is explained.

### *A. Self-Configuration for Patient Data access by Doctor*

A doctor provides each patient with a patient node either at the danger zone or at the triage zone. The patient node is worn around the neck of the patients and switched on by the doctor. A doctor node can communicate with a patient node at the disaster site to fulfil any of the following functionalities

- In the triage zone the doctor uses his tablet PC to configure the patient node with triage and personal information.
- During the emergency response at the triage and treatment zone the doctor's tablet PC might have to often read the patient status (vital signs, triage information) from the patient node.

The following problems might occur during the doctor-patient node interactions

- Considering the large number of patients and fewer doctors present at the disaster site, it can be complicated and time consuming for the doctor to manually configure his tablet PC for accessing or configuring the data of a patient node.
- Thus, automatic configuration of patient-doctor nodes is required. But during automatic configuration there can be large numbers of patients at close proximity and the doctor node must be sure that it talks to the right node i.e. patient identification can be an issue.

In order to solve the above mentioned problems, a self-configuring mechanism using RFID and ZigBee is demonstrated in the next subsection to provide a simple, automatic and safe data access between the patient node and the doctor node.

#### *1) ZigBee-RFID based Self-configuring Mechanism*

A doctor who needs to configure or access a patient data at the disaster site brings his Tablet PC in close proximity (in the order of cm or a meter) to that particular patient node and with a single click in his Tablet PC he can access the right patient's data safely, easily and automatically. This self-configuring mechanism is described below.

The patient node consists of a ZigBee transceiver and RFID passive tag. The RFID passive tag is used for patient identification and stores a unique id which in our case is the 64 bit IEEE address of the corresponding ZigBee-ready patient node. The RFID passive tag has a very short range. The doctor node is a Tablet PC enabled with a ZigBee module and an RFID reader and runs an application for patient data access.

When the doctor needs to access the patient's data he brings his Tablet PC in close proximity to that particular node and clicks a button on the Java application. Considering that the proximity of the tag is in the order of cm, the RFID reader of the doctor node can only read the tag of that particular node. Of course it is assumed that no two patients are present in proximity of cm range in the disaster site. The communication flow between the patient node and the doctor node is depicted in figure 3.

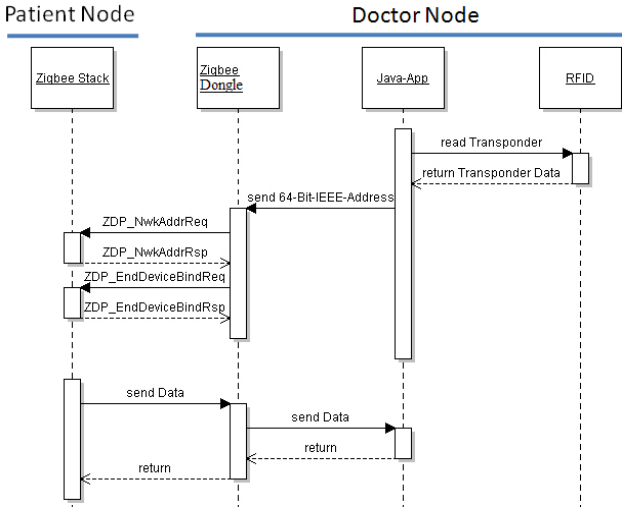



Figure 3. Communication flow diagram of ZigBee-RFID based self-configuring mechanism

The RFID reader reads the unique id of the patient node which is the 64 bit IEEEE address of that node and provides it to the application. The application then connects to the ZigBee stack of the dongle and sends this 64 bit IEEEE address via ZigBee communication and requests the network address of that node. Once the network address is received by the dongle it performs a ZigBee application layer binding between the doctor node and that patient node. After this a mobile ad-hoc network is formed and the doctor can safely access the patient data.

#### *2) Demonstrator*

In order to verify the self-configuring mechanism for seamless data access between patient node and doctor node explained above, a demonstrator is constructed.

The ZigBee-ready temperature sensor node [22] attached with a RFID passive tag is used as the patient node. During the time of development of this demonstrator there was no Tablet PC available with a ZigBee module and RFID reader integrated. So the doctor node is represented in this demonstrator by a laptop connected to a ZigBee dongle and a

RFID reader. A ZigBee dongle which offers ZigBee 2003 compatible interface for the Plug & Play connection to a PC is used. The RFID reader is S4100 MFR from TI which has an ISO 15693 conformal RFID reader and connects via RS232 interface to a PC.

RFIDZigBee is a Java application with a simple GUI for patient data access and runs in the laptop. The ZigBee dongle is connected to the RFIDZigBee via a Python based intermediate layer developed by us that interfaces ZigBee stack of the dongle to RFIDZigBee. The demonstrator setup is depicted in figure 4. It consists of two patient nodes and a doctor node.

The doctor node is brought in close proximity to patient node1 and the "connect button" is clicked in the RFIDZigBee software. The RFID reader of the doctor node is able to read only the 64 bit IEEE address of patient node 1 followed by which the ZigBee dongle binds the doctor node to the patient node1 to form a mobile ad-hoc network. Now the doctor node safely received the test data from the patient node 1.

Thus the demonstration verifies that it is possible for the doctor node to automatically record the patient data from the patient node with a single button press. Besides the patient identification using RFID passive tag makes sure that the doctor receives data from the right patient and increase security.

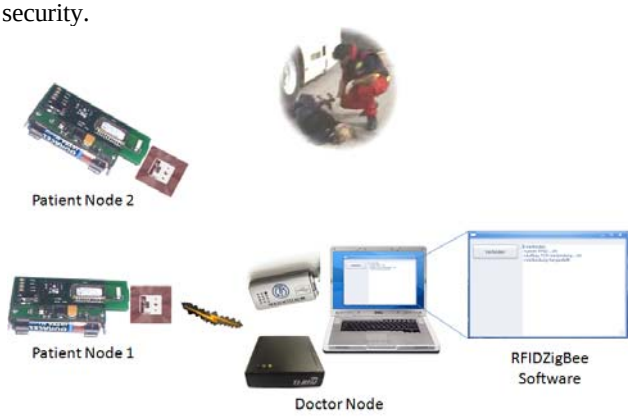



Figure 4. Self-Configuration for Patient Data access by Doctor-Demonstrator

### B. *Self-Protection*

In DAN the patient data should be securely sent to the server of the management centre and should be automatically protected against any malicious attack.

There can be two main types of security concerns in WSN: data confidentiality and data authentication [19]. Any message transmitted can be received by an intruder and cause confidentiality problems. Encrypting the message sent with a security key can solve confidentiality problems. The second security concern is the data authentication wherein an intruding node can modify and resend even an encrypted message. Including message integrity code (MIC) for each outgoing message can allow the recipient to check whether the message is corrupted.

Even intrusion of any device can be prohibited through device authentication technique. If the nodes are not tamper-resistant then an intruder can access the security key from the memory of the device. So tamper-resistant mechanisms for DAN nodes will have to be formulated such that once tampering is detected the node should automatically erase sensitive information.

ZigBee supports the AES standard for data encryption and also supports measures for device and data authentication. However testing the AES encryption, device and data authentication of ZigBee for DAN and devising tamper-resistant techniques are open questions and are not addressed within the scope of this paper. Also during an emergency response process different work groups (emergency doctors, etc) need to access the patient data and each group is hierarchical. So the data access has to be restricted based on the type of workgroup and the hierarchy of its members.

### C. *Fault Tolerance Mechanism*

Since DAN should operate in a hostile environment it should be fault tolerant to provide reliable service which is an open question. For example when the DAN is in operation the entrance of fire engine at the site can down one part of the network due to severe RF attenuation by the metal objects of the fire engine. This can be handled if a link degradation mechanism which can detect faulty links on the fly and inform the coordinator which can make alarm signals.

A simple fault tolerant approach is to resort to redundant deployment of sensor nodes and replication of information between sensor nodes can be adopted to overcome some of the related problems. Another approach is to provide the whole system self-healing capability in a cooperative way. The self-healing feature of sensor networks provides the ability to adapt to unforeseeable situations, diverse environments, and dynamic changes.

### D. *Summary*

In the self-organization of DAN section the need for self-organization with respect to the DAN communication aspect is mentioned. The self-configuring and the self-healing feature of ZigBee can be useful for the DAN network formation and for the healing of routing paths respectively. A self-configuring mechanism for safe and easy access of the patient data by the emergency doctors is proposed and evaluated using a demonstrator.

As part of the future work, new self-organization ideas to satisfy the DAN communication aspect functionalities will have to be developed. For instance, self-organizing techniques for the new patient nodes joining the DAN to automatically know their destination address (collector node) can be developed. Fault tolerance techniques leading to robust communication in DAN will have to be developed. Secure mechanisms for communicating patient data to the management centre and self protection techniques against malicious attacks will have to be developed.

## V. RELATED WORK FOR LOCALIZATION IN WSN

Current localization systems like Active Badge [17], Cricket [16], RADAR [12], SpotON [11] and other RFID based systems like LANDMARC [18] require a lot of infrastructure while GPS [6] is not suitable for indoor scenarios.

Due to the non-linear, non-Gaussian properties of our scenario and the unpredictable movement of the patients, Monte Carlo Localization (MCL) based methods are investigated to solve our problem. Hu and Evans [10] proposed a range-free localization algorithm called MCL that only works in mobile sensor networks. Dil et al. proposed a range-based version of the MCL algorithm [13]. In [14] Baggio et al. introduced MCB as a variant of MCL.

In [15], Rudafshani et al. proposed MSL (Mobile Static Localization) as a range-free algorithm based on MCL, that improves localization accuracy by using the location estimates of all the anchor nodes and pseudo anchors present in first and second hops. Each node is assigned with a closeness value which indicates the accuracy of that node's location estimate. During initialization, samples are drawn from the entire area. A weight is then assigned to every sample depending on the closeness value of its neighboring nodes. MSL then computes a new location estimate (the weighted mean of samples) and a new closeness value. After initialization, the new samples are drawn during prediction within a circle centered at the current sample location and a radius of the maximum node speed.

## VI. RSSI BASED PATIENT LOCALIZATION METHOD

The goal is to track the patient nodes at the disaster site and provide their real time locations to the monitor station. The requirements [7] for patient tracking, that the new algorithm and DAN must comply with, are: handle the different environments (both outdoor and indoor) since disasters can happen at different locations; use minimum or no special infrastructure (static anchor nodes) due to the lack of deployment time; track 30-200 patient nodes moving with varying speed (0 to 3m/s); attain an accuracy of around 10m; be scalable and robust; have low computation and communication overhead. The main challenge here is to handle the varying mobility and different environments with adverse RF conditions and also use minimum or no infrastructure. Based on these requirements a Received Signal Strength Indicator (RSSI) based [23] patient localization methodology is proposed for DAN.

At the beginning of the emergency response, a portable monitor station (typically a notebook) gets online; the static anchor nodes are deployed manually covering the disaster area; the emergency doctor nodes (can be a PDA with GPS [6]) act as mobile anchors; once a patient is found the doctor provides him with a wearable patient node. Each patient node runs a decentralized localization algorithm to estimate its real time location and sends it to the monitor station.

In DAN the only information about the patient node is its maximum speed $v_{\max}$, so our system equation is modeled as shown in equation (1).

$$\underline{x}_k = \underline{x}_{k-1} + \underline{w} \tag{1}$$

where $\underline{x}_k = (x_k, y_k)^T$ is the position of the patient node at time unit $k$ and $\underline{w}$ is a random variable which is uniformly distributed within a circle centered around the zero vector with radius of the maximum speed value of the node $v_{\max}$.

After each time unit, a patient node requests for the position, the RSS and the closeness value (a measure of the accuracy of the neighboring node's position estimate) of its entire one-hop neighbors. The patient node collects these values and estimates the ranges (distances) to these neighbors based on the RSS. Each range measurement is modeled according to the measurement equation (2)

$$z_k = \left\| \underline{x}_k - \underline{\tilde{x}}_k \right\| + v \tag{2}$$

where $\underline{x}_k = (x_k, y_k)^T$ and $\underline{\tilde{x}}_k = (\tilde{x}_k, \tilde{y}_k)^T$ are the positions of the patient node and the one-hop neighbor after time unit $k$, respectively, $z_k$ is the range measurement and $v$ is a Gaussian random variable with mean $\mu_v$ and standard deviation $\sigma_v$. The values for $\mu_v$ and $\sigma_v$ are given by the systematic and random error of the measurement model that is used for simulating the distance estimations of the patient node in our simulator. However, in a real scenario these values are deduced from the environmental factors i.e. all anchor nodes within the transmission range of each other compute the error between their actual distance and their estimated distance. All these values are collected in a single node and the mean $\mu_v$ and the standard deviation $\sigma_v$ are calculated.

Once the patient node estimates the distances to its one-hop neighbors it runs a new decentralized position estimation algorithm. None of the algorithms mentioned in Section V exactly meets the specific requirements for patient position estimation, in their current form. In terms of accuracy error, computation cost and communication cost, MSL from Rudafshani et al. acts as a good base for our scenario because it works well with any number of static and mobile nodes in an irregular shaped sensor network. But since MSL is a range-free algorithm it is unable to reach the required accuracy of our scenario. So we used MSL as base and developed the Range-Based Monte Carlo Patient Localization algorithm (MPL) [27]. MPL gave encouraging results with a Gaussian measurement model but we want to test our patient tracking solution with a realistic non-Gaussian measurement model and improve its performance. So in this paper we have developed the 'Improved Range-Based Monte Carlo Patient Localization' (IMPL). Also, to compare IMPL's performance with other non-linear filter approaches and to find its suitability for patient tracking we have developed a new unscented Kalman filter based algorithm called 'Range-Based Unscented Kalman Filter Patient Localization' (UPL). Either IMPL or UPL runs in

each patient node to estimate its location and communicates it to the monitor station.

### A. Improved Range-Based Monte Carlo Patient Localization (IMPL)

IMPL [1] maintains a weighted sample set in order to estimate the patient node's position. Our new additions in IMPL are:

- Transmission range-based anchor boxing during initialization.
- Removing the usage of two-hop neighbors and conditional selection of one-hop pseudo anchors to gain higher accuracy and reduce computational costs
- A new method for sample weighting.

IMPL mainly consists of three steps: prediction, weighting and resampling.

*1) Prediction:* The prediction step depends on whether there's already an established sample set (after initialization) or not (during initialization).

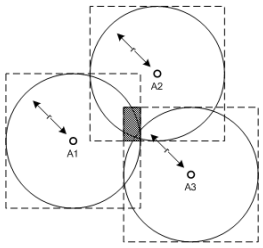



Figure 5. Forming the anchor box

In prediction during initialization, first the area to be sampled from is constrained to an anchor box. The region covered by the transmission range $r$ of each one-hop anchor is approximated to a box as shown in figure 5. The overlapping area of all the boxes (shaded area in figure 5) forms the anchor box. Then 50 uniformly distributed samples are drawn from the anchor box. In prediction after initialization, we take each sample from the previous time step and form a circle of radius $v_{\max} + addition$ centered at that sample's position. From every circle one new sample is drawn.

*2) Weighting:* A weight is calculated for each sample (based on the range measurements) to know if the sample is good or bad. In order to weight the sample $i$ all range measurements to one-hop anchors and to one-hop pseudo anchors with a closeness value less than the current blind node's closeness value are selected. If the range measurement $rm_j$ to one-hop neighbor $j$ is selected, a partial weight $wp_i^j$ will be computed for it. Therefore the range measurement is projected onto a Gaussian distribution (see figure 6), which has $\mu = d + \mu_v$ as mean, where $d$ is the distance between the sample and the one-hop neighbor, and $\sigma_v$ as standard deviation (see (2)).

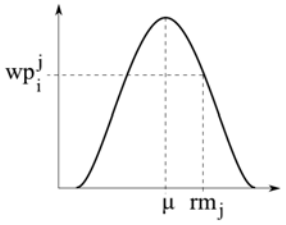



Figure 6. Calculating the partial weight

So the partial weight $wp_i^j$ for sample $i$ and range measurement $rm_j$ is calculated as

$$wp_i^j = \left(1/\sigma_v\sqrt{2\pi}\right)\cdot e^{-0.5\cdot\left(rm_j-\mu\right)^2/\sigma_v^2}. \quad (3)$$

The total weight $w_i$ for sample $i$ is the product of all the partial weights.

*3) Resampling:* After normalizing the weights of the sample set, samples are redrawn from the normalized sample set with a probability proportional to their weights. The size of the sample set remains the same and the same weight is assigned to all new samples such that the weights are normalized.

The position estimate $(x, y)$ of the blind node is calculated as the weighted mean of the sample set. The closeness value for blind node $p$ with $N$ samples is computed as

$$closeness_p = \sum_{i=1}^{N} w_i \sqrt{\left(x_i - x\right)^2 + \left(y_i - y\right)^2} / N \quad (4)$$

where $(x_i, y_i)$ denotes the position of sample $i$, $w_i$ denotes the weight of the sample $i$ and $(x, y)$ is the current location estimate of node $p$. The anchor nodes always have a closeness value of 0.

### B. Range-based Unscented Kalman Filter Patient Localization (UPL)

UPL [1] is based on an unscented Kalman filter [8] which only approximates the mean $\hat{\underline{x}}_k^+$ and the covariance $P_k^+$ of the posterior after time unit $k$. The trace of the covariance matrix provides the closeness value and indicates the position estimate's accuracy. In UPL distance measurements (measurement vector $\underline{z}_k$) to all one-hop anchors and to all one-hop pseudo anchors whose closeness value is below a threshold value (100) are used.

The mean $\hat{\underline{x}}_0^+$ and the covariance $P_0^+$ at time unit 0 are initialized with the mean and the covariance of a uniform distribution on a rectangle representing the area of the network, because there is no information about the patient

node's position at the beginning. Since the system equation is linear the prediction step of the Kalman filter is used to calculate the mean $\hat{\underline{x}}_k^-$ and the covariance $P_k^-$ of the prior distribution after time unit $k$ as shown in (5).

$$\hat{\underline{x}}_k^- = \hat{\underline{x}}_{k-1}^+ \,,\; P_k^- = P_{k-1}^+ + Q_k \tag{5}$$

where $Q_k$ is the covariance of a uniform distribution on a circle with radius $v_{\max}$.

In the filtering step a set of four so-called sigma points $\underline{s}_k^i$ is deterministically chosen (see (6) and (7)) whose ensemble mean and covariance are equal to $\hat{\underline{x}}_k^-$ and $P_k^-$, respectively [8].

$$\underline{s}_k^i = \hat{\underline{x}}_k^- + \left(\sqrt{2 \cdot P_k^-}\right)_i^T, \quad i = 1,2 \tag{6}$$

$$\underline{s}_k^i = \hat{\underline{x}}_k^- - \left(\sqrt{2 \cdot P_k^-}\right)_{i-2}^T, \quad i = 3,4 \tag{7}$$

For every selected one-hop neighbor all sigma points are transformed according to the corresponding measurement equation. If $\tilde{\underline{x}}_k^j = \left(\tilde{x}_k^j, \tilde{y}_k^j\right)^T$ is the position of one-hop neighbor $j$, then one obtains (8).

$$t_k^{ij} = \left\| \underline{s}_k^i - \tilde{\underline{x}}_k^j \right\| + \mu_v \,, \tag{8}$$

where $t_k^{ij}$ is the transformation of sigma point $\underline{s}_k^i$. All transformed sigma points, which correspond to the same sigma point, form a transformed sigma point vector $\underline{t}_k^i$.

The predicted measurement vector $\hat{\underline{z}}_k$ is computed as

$$\hat{\underline{z}}_k = (1/4) \cdot \sum_{i=1}^4 \underline{t}_k^i \tag{9}$$

The covariance of the predicted measurement vector and the cross covariance between $\hat{\underline{x}}_k^-$ and $\hat{\underline{z}}_k$ are obtained as shown in (10) and (11).

$$P_k^y = (1/4) \cdot \sum_{i=1}^4 \left(\underline{t}_k^i - \hat{\underline{z}}_k\right)\left(\underline{t}_k^i - \hat{\underline{z}}_k\right)^T + R_k \,, \tag{10}$$

$$P_k^{xy} = (1/4) \cdot \sum_{i=1}^4 \left(\underline{s}_k^i - \hat{\underline{x}}_k^-\right)\left(\underline{t}_k^i - \hat{\underline{z}}_k\right)^T \tag{11}$$

where $R_k$ is a diagonal matrix with all its main diagonal entries equal to $\sigma_v^2$.

Now the filtering step of the Kalman filter can be applied to calculate the mean and the covariance of the posterior.

$$K_k = P_k^{xy}\left(P_k^y\right)^{-1} \tag{12}$$

$$\hat{\underline{x}}_k^+ = \hat{\underline{x}}_k^- + K_k\left(\underline{z}_k - \hat{\underline{z}}_k\right), \; P_k^+ = P_k^- - K_k P_k^y K_k^T \tag{13}$$

## VII. PERFORMANCE EVALUATION

We used the simulator used by Rudafshani et al. [15] to develop and test our algorithms. In order to do simulations close to reality we added the following new features: disaster management mobility model, range measurement model, anchor position error.

Disaster management mobility model: In order to create a mobility model that replicates the MANV scenario (see Section II) and test our algorithm a new trajectory based mobility model is developed. Since the nodes move in a random fashion in the random waypoint mobility model it cannot be used for modeling our scenario. This new mobility model is used to setup a MANV-disaster management scenario (see figure 7) within an area of 500m x 500m with 100 nodes that include 15 static and 35 mobile anchor nodes. The maximum speed of all nodes is set to 3m/s. At the beginning of the simulation the doctor nodes appear dynamically at the site followed by the patient nodes. Most of the patients are moved from one zone to another accompanied by at least one doctor. When a patient node arrives at the transport zone it leaves the site after predefined time units. Similarly the doctor nodes leave the site, too.

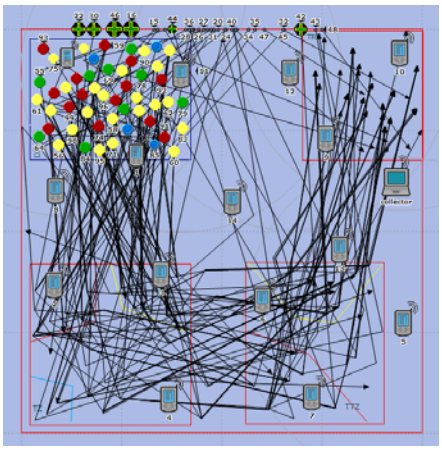


Figure 7. Disaster management mobility model

The simulations are also done for the random waypoint mobility model [5] in order to find the influence of certain parameters (transmission range, etc.) on the algorithms' performance.

Anchor position error: We incorporate erroneous anchor positions into the simulation. An additive zero-mean Gaussian error is used with a 3 sigma bound of 2m for static anchors and 10m for mobile anchors.

Range measurement model: We create a close to realistic measurement model. An outdoor area with adverse RF conditions, is setup with a sensor network and real RSS data is collected to obtain the distance estimations. A frequency distribution of the distance estimation error is plotted. The frequency distribution of error is analyzed along with the evolution of error over time and two factors are noticed: First, most of the time the actual distance is underestimated and second, the error is time-correlated. In order to account for the time-correlation the simulator keeps a state variable between all pairs of nodes which need to perform a distance measurement. Every state corresponds to one certain base error value in the range between -70m and 30m. Accessorily, a zero-mean Gaussian random error with a standard deviation of 5m is added to the base error. After a range measurement the state of the correspondent pair is updated according to a Markov chain. It may remain the same or a transition into the state with the next base error value below or rather above may occur. The transition probabilities are empirically determined to fit the error distribution of the outdoor experimental data. The mean and the standard deviation of the outdoor error distribution are used as systematic and random error of the measurement model, respectively. Since MSL is a range-free position estimation algorithm it is not affected by the range measurement model.

In all simulations we used an ideal transmission range of 200m for all nodes.

### A. *Simulation results for random waypoint mobility model*

The simulation setup is made considering an area of 400m x 400m with a total of 100 nodes that include 10 static and 20 mobile anchor nodes. All nodes move according to the random waypoint mobility model with their maximum speed set to 3m/time step.

*1) Accuracy using a static model:* To simulate a static model the maximum velocity of all nodes is set to 0m/time step. IMPL has an average error of 5.22m while UPL has an average error of 9.10m (see figure. 8). IMPL can therefore handle a static scenario better than UPL. The average error of MSL is 14.41m.

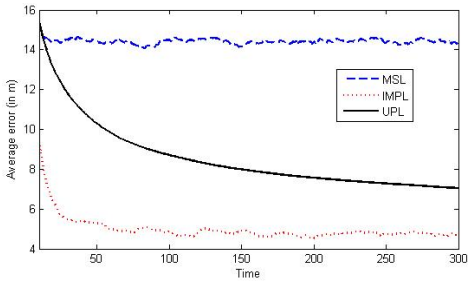



Figure 8. Accuracy using a static model

*2) Accuracy over time:* The average error of IMPL is similar to the average error of UPL but IMPL needs less time to converge during simulation start (see figure 9). The average error of MSL is constantly high.

*3) Accuracy over varying transmission range:* figure 10 shows that as the transmission range increases, the average error of both IMPL and UPL reduces. This is because with increasing transmission range the number of neighboring nodes (anchors or pseudo anchors) also increases. The accuracy error of MSL worsens with high transmission ranges which can be due to its weighting step.

*4) Accuracy over varying number of blind nodes:* As the number of blind nodes increases, both IMPL and UPL improve their accuracy slightly, showing that the usage of pseudo anchors has a positive effect. Compared to IMPL the accuracy error of MSL is very high (see figure 11).

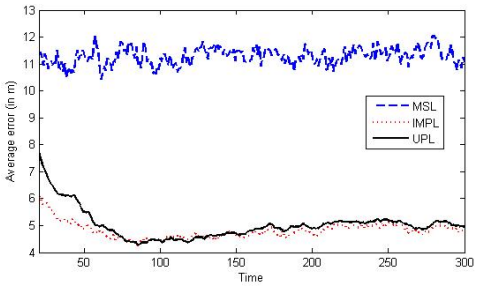



Figure 9. Accuracy over time in the random waypoint mobility model

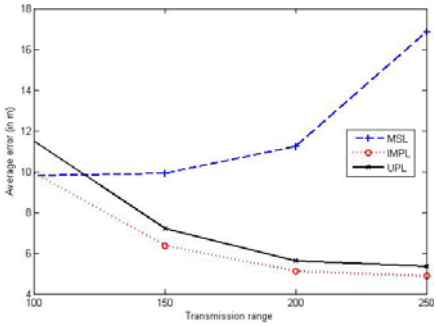



Figure 10. Accuracy over varying transmission range

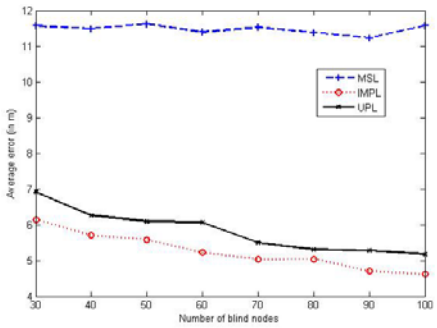



Figure 11. Accuracy over varying blind nodes

*5) Accuracy over varying number of anchor nodes:* The error of MSL stabilizes only when there are 20 anchor nodes. When the number of anchor nodes are less IMPL performs better than UPL. As the number of anchors increase both IMPL and UPL have similar accuracy error (see figure 12).

In Fig. 11 and 12 as the number of nodes increases (blind nodes and mobile reference nodes), both IMPL and UPL attain an accuracy of 5m to 10m implying that the algorithms scale with network node size in terms of accuracy.

*6) Accuracy over varying sample set size:* IMPL needs at least 20 samples to attain an average error of around 5m. MSL also converges at around 20 samples but its average error is significantly higher. After around 50 samples the average error of IMPL does not improve much so we chose a maximal sample set size of 50 (figure 13).

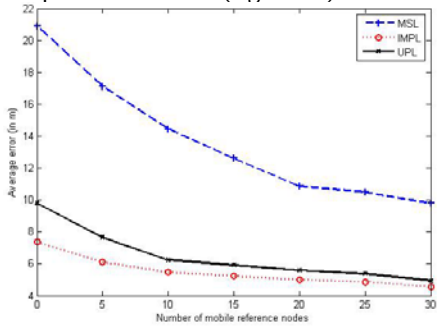



Figure 12. Accuracy over varying anchor nodes

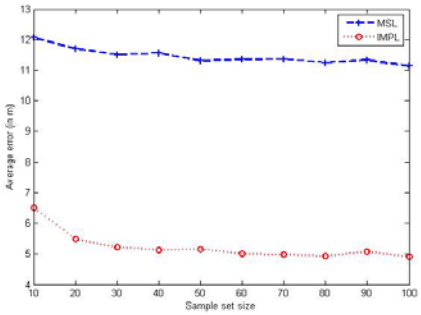



Figure 13. Accuracy over varying sample set size

## B. Simulation results for disaster management mobility model

Here IMPL and UPL are simulated for the disaster management mobility model described in Section VII.

*1) Accuracy over time:* IMPL achieves an average accuracy of 6.78m whereas UPL has an average accuracy of 7.87m (see figure 14). In between 600s and 1600s the accuracy of IMPL is stabilized. Before 600s and after 1600s the average error is less stable. This is due to the dynamic addition and removal of doctor and patient nodes which act as anchors and pseudo anchors, respectively.

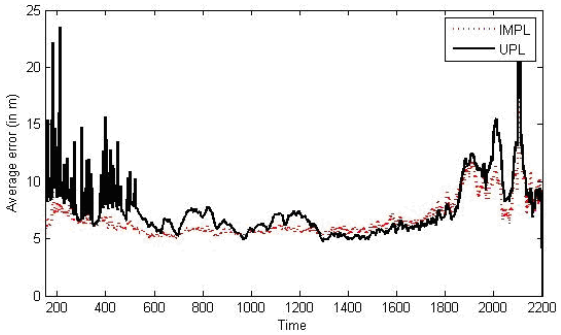



Figure 14. Accuracy over time in the disaster management mobility model

*2) Number of bad localized nodes for IMPL over time:* figure 15 shows that between 400s and 1200s almost all doctor and patient nodes are present. Around 20 patient nodes are localized with an error less than 5m and another 20 nodes with an error between 5m and 10m. The achieved accuracy is within the requirements defined in Section VI. The number of bad localized nodes for UPL was similar to that of IMPL.

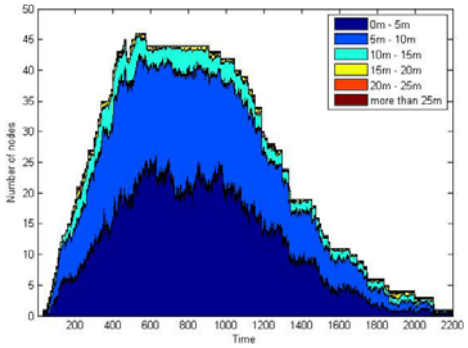



Figure 15. Bad localized nodes for IMPL in the disaster management mobility model

*3) Computation Cost:* The average arithmetic operations required to run IMPL or UPL once, at a particular time step are counted to form the computation cost as shown in figure 16. Although theoretically the computational costs of a particle filter based solution are higher than that of an unscented Kalman filter based approach [8], Fig. 13 shows the opposite for IMPL and UPL. This is because we have a smaller sample set size for IMPL in comparison to general particle filters used in Robotics [9]. UPL's high computation cost is due to its handling of high dimensional measurement vectors (because of large number of neighbouring nodes) which involves complex matrix operations.

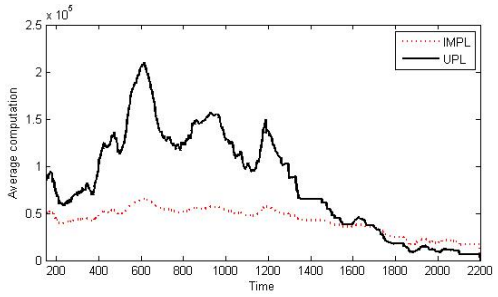



Figure 16. Computational cost in the disaster management mobility model

## VIII. CONCLUSION AND FUTURE WORK

In this paper ZigBee's 2.4 Ghz operation is tested using ranging experiments and the results show that ZigBee is basically suitable for DAN even though further empirical investigations for co-existence and RF signal attenuation will be future work. The self organizing features of ZigBee are are useful for DAN even though not enough. A new self-configuration mechanism for seamless patient data access by the doctor at the disaster site is proposed and verified by us using a demonstrator.

We have also proposed two Bayesian filter based position estimation algorithms - IMPL and UPL for tracking patients during emergency response. We have simulated IMPL and UPL using two mobility models - disaster management mobility model and random waypoint mobility model under realistic simulation setups (an outdoor experiment based range measurement model and anchor error inclusions). The simulation results for the random waypoint mobility model show that the average accuracy error over time is almost same (around 5.6m) for both IMPL and UPL while MSL has a high accuracy error (11.3m). With less number of anchor nodes and low transmission range, the accuracy error of IMPL is better, but as these values increase both IMPL and UPL have a similar performance. The simulation results for the disaster management mobility model show that both IMPL and UPL satisfy the requirements for patient tracking in terms of accuracy. Half the number of patient nodes has a localization error less than 5m and the other half (almost) is between 5m-10m. In terms of computation cost IMPL is better than UPL, but this is an indication obtained from a simple computation model. The results show that IMPL is better than UPL for patient tracking and in tandem with the emergency response system facilitates efficient logistics management at the disaster site. A demonstrator with IMPL running in each node will be implemented and tested as part of future work.

## REFERENCES

[1] A. Chandra-Sekaran, P. Weisser, C. Kunze, and K. Mueller-Glaser, A Comparison of Bayesian Filter Based Approaches for Patient Localization During Emergency Response to Crisis, in *The Third International Conference on Sensor Technologies and Applications (SENSORCOMM 2009 )*, Athens, Greece, 2009.

[2] Strategische Neukonzeption der ergänzenden technischen Ausstattung des Katastrophenschutzes im Zivilschutz - Dr. Meyer-Teschendorf Projekt-Ref: P. Rechenbach

[3] Chandra-Sekaran, A., et al.: Efficient Resource Estimation During Mass Casualty Emergency Response Based on a Location Aware Disaster Aid Network. EWSN,vol.4913/2008. ISSN0302-9743, 2008.

[4] Organisationsrahmenplan für erste Maßnahmen am Unfallort, den Transport und die stationäre Versorgung Schwerbrandverletzter bei einem Massenanfall- Stuttgart, 1982. - 24 S.; (dt.)

[5] Tracy Camp et al.:. A Survey of Mobility Models for AdHoc Network Research.Wireless Communication&Mobile Computing'02.

[6] Global Positioning System http://www.gps.gov/

[7] Lechtleutner, Prof.:„MANV Anforderung Fragebogen Antwort Fachhochschule Köln, Rescue Engineering.

[8] Simon, Dan: Optimal State Estimation John Wiley & Sons, 2006.

[9] Probabilistic Robotics by sebastian Thrun, Burgard and Dieter Fox.

[10] Hu and Evans: Localization for Mobile Sensor Networks. Conference on Mobile Computing and Networking (MobiCom), 2004.

[11] Hightower, et al. SpotON: An indoor 3D location sensing technology based on RF signal strength, Tech. Report'00: Univ. of Washington.

[12] Bahl, P. et al. RADAR: An inbuilding RF based user location and tracking system, Proc. of IEEE Infocom. vol. 2, pages 775-784, 2000.

[13] Dil, Bram, et al.: Range-based localization in mobile sensor networks. Proc. of the 3rd European Workshop on WSN. pp. 164-179, 2006.

[14] Baggio, Aline and Langendoen, Koen: Monte-Carlo localization for mobile wireless sensor networks Technical Report TU Delft. 2006

[15] Rudafshani, et al.: Localization in Wireless Sensor Networks. Proc. of Information processing in sensor networks'07 conference. pp. 51-60.

[16] Priyantha, et al.: The Cricket Location Support System, Proc. of Mobile computing and networking (MobiCom'00). pages 32-43.

[17] Want, R., et al., The active badge location system, ACM Trans. Inf. Syst., pages 91-102, 1992.

[18] Muthukrishnan, et al.:Towards Smart Surroundings:Enabling Techniques and Technologies for Localization. Proc. of LoCA 2005.

[19] S. Farahani, ZigBee Wireless Networks and Transceivers. Newnes, 2008.

[20] ZigBee-Alliance, ZigBee Specification 2006, 053474th ed., 2006.

[21] R. R. Rao and J. Eisenberg, Improving Disaster Management: The Role of IT in Mitigation, Preparedness, Response, and Recovery. National Academies Press, 2007.

[22] A. Chandra-Sekaran, A. Nwokafor, L. Shammas, C. Kunze, and K. Mueller-Glaser, A Disaster Aid Sensor Network using ZigBee for Patient Localization and Air Temperature Monitoring, *International Journal on Advances in Networks and services*, no. ISSN: 1942-2644, pp. 68–80, 2009.

[23] A. Chandra-Sekaran, P. Dheenathayalan, P. Weisser, C. Kunze, and W. Stork, "Empirical Analysis and Ranging using Environment and Mobility Adaptive RSSI Filter For Patient Localization during Disaster Management," in The 5th International Conference on Networking and Services (ICNS 2009), Valencia, Spain, 2009.

[24] Texas-Instruments, *CC2430DB Demonstration Board User Manual*, 2007.

[25] G. Kupris, Channel Agility and ZigBee / WLAN (IEEE 802.11 g and n) Coexistence, Freescale Semiconductor, Tech. Rep., 2008.

[26] Rmoni-Wireless, ZigBee- Do we still need wires? FEE Measurements in Automotive, June 2007.

[27] A. Chandra Sekaran, G. Stefansson, C. Kunze, K. Mueller-Glaser, and P. Weisser, "A Range-Based Monte Carlo Patient Localization during Emergency Response to Crisis," in The Fifth Advanced International Conference on Telecommunications (AICT 2009), Venice, Italy, 2009.

# Automated Dependability Planning in Virtualised Information System

Marco D. Aime
m.aime@polito.it

Paolo Carlo Pomi
paolo.pomi@polito.it

Marco Vallini
marco.vallini@polito.it

Politecnico di Torino
Dipartimento di Automatica e Informatica
Corso Duca degli Abruzzi 24, 10149 Torino, Italy

## Abstract

*Virtualisation technologies widely affect information dependability practices, and suggest novel approaches for dependable system configuration. We analyse how to exploit these peculiarities within a tool for semi-automated configuration of virtual information systems. We present the general architecture of the tool, which is the product of previous work focused on traditional information systems, and discuss how to update its configuration strategies when moving to virtual environments. We present the stepwise process that, starting from the model of the target service, computes a set of configuration plans that determine which virtual machines should be deployed, their internal configuration, and their intended interactions. Our tool also generates re-configuration templates to switch between different configuration plans in case of dependability problems or changed requirements. Actually, reaction plans are heavily affected by virtualisation technologies, which permit fast re-allocation and reconfiguration of virtual machines. We finally discuss the integration of our process with virtual machine management systems, in order to perform configuration and re-configuration in fully automated way.*

**Keywords:** automatic system configuration; dependability ontology; virtualization.

## 1 Introduction

Virtualisation technologies are changing traditional architecture in data center environments [8, 17, 26]. They widely affect the trade-off between costs and benefits of standard dependability mechanisms and suggest the introduction of novel approaches. In previous work [6] we addressed the policy-driven, automatic configuration of information systems, taking care of dependability requirements at the business service level. [6] describes process, algorithms, and tools for semi-automatic generation of dependability configurations for information systems not relying on modern virtualisation technologies. In [7], we have first analysed how to adapt our approach to virtualised information systems. In this paper, we investigate how virtualisation changes the best practices of dependability planning, and how we have updated our configuration framework accordingly. In particular, after a quick reference to virtualisation concepts, we summarise the relevant background from previous works in Sec. 2, discuss virtualisation-based dependability configuration strategies in Sec. 3, describe the configuration generation algorithms implemented by our tool in Sec. 4, and give guidelines to integrate with virtual machine management systems in Sec. 5.

The heart of modern virtualisation technologies is the Virtual Machine Monitor, or hypervisor, that allows multiple operating system instances to run on the same physical host. The first technology class is *full virtualisation*, like VMware[28], which introduces a layer that traps and emulates all privileged instructions. On the other hand, *para virtualisation* [9] uses a modified operating system to let virtual machines directly use hardware resources. This technique introduces lower overhead and best fits data centres.

Hypervisor offers isolation among hosted virtual machines: if a service running on a guest machine is tampered, security threats are confined and do not affect the other virtual machines. Virtual machines (VMs) are connected together by virtual connections, managed by the hypervisor, with advantages in terms of performance and security [16]. Actually, process and connection isolation assumes the hypervisor as trusted. The set of techniques to achieve trusted hypervisors, e.g. Trusted Computing [25], are out of the objectives of this paper.

Hypervisor can also freeze and restore VMs, but the migration of a VM requires taking a snapshot of *all* its resources at a given time, sending this snapshot to the destination hosting device, and rebuilding there the VM from the snapshot. Although migration is extremely promising for dependability, migration functions, as currently provided by

the hypervisor, make hard assumptions: source and destination hosts must share the same hardware architecture; the source and destination hosts must share the same network segment; the VM image and virtual storage must be already available at the destination host. Moreover, management of failures in the migration process is generally incomplete [19]. Therefore, safe VM migration requires in practice a complex process to be carefully planned.

Our process and tools allow minimising the risks of virtual systems by planning hot/cold standbys for critical VMs, configuring distributed storage, and protecting data transfer across the system. Starting from service-level models of the target dependability requirements, our tools generate multiple configurations featuring different dependability solutions. Configurations may be further selected according to other constraints (e.g. performance, cost, energy). We instead exploit these alternative configurations to construct re-configuration templates, intended in terms of switching between different configurations to react to dependability problems or changed requirements.

## 2 Planning Framework

Our framework implements a model-based approach to progressive policy refinement and configuration generation. We developed a tool, able to generate set of alternative configurations for complex ICT systems, considering the dependability requirements at business level.

Approaches for automatic configuration were proposed by [23], using policy refinement techniques widely adopted also in our work. The author focused his attention on configuration of devices, but did not consider in detail the chance to have different deployment plans. Another interesting work on policy refinement is [10] but it is mainly focused on authentication and authorisation rules.

In [11] authors present an approach very similar to ours, but limited to J2EE based applications, whereas ours presents a technology independent solution, in fact, in this paper we study its application at virtualized environments. Furthermore, they do not present a clear methodology for modelling information system nor a classification of the system components from the dependability point of view. [22] shows a promising model-driven approach to achieve security requirements, but it does not consider dependability problems. Another interesting project is [12], but is more concerned on software development rather than on configuration and management of an existing system.

This tool generates an allocation plan, for alternative hosting of service components, then for every plan generates the configuration for the devices involved in the hosting, in the communication enabling and protection, and the protection of hosts.

Our approach is based on a system of rules that act on the ontology representing the system. Add the beginning the ontology is separated in service and requirements on one side, and on the other side the hardware and software able to host and protect the service. Every rule reads the information from the ontology and adds some information, reducing the gap among the two side of the ontology. The rules executed after can use all the information already generated, since some rules can add the connection between the two sides. Such approach permits to fully track the automatic generation process, and insert other rules to exploit different scenarios or dependability solutions.

More details of the framework are presented in [6]: we resume here its building blocks as long as this work is layered upon them. Three steps compose the process: model construction, configuration generation and configuration ranking. During *model construction*, we describe the information system, composed by business services, virtual machines, physical resources, and dependability requirements. We model business services using a profile of W3C's *Web Service Choreography Description Language* [2]. We consider services as a set of interactions between different service components, from a choreographic point of view. This approach handles every business service component as a black box, and focuses on the role inside the global business process. To model the virtual infrastructure, we use the *System Description Language* [3] based on DMTF's CIM[24]. This model is based on a multi-layered graph of physical and software elements in the network (nodes), and their structural and logical dependencies (edges). All our models have an XML serialisation. These models are then represented as ontology, to be read by the generation process.

The *configuration generation* process produces a set of alternative virtual network topologies (composed by virtual machines and virtual networks) driven by templates plus corresponding per-device abstract configurations. Every configuration provides the business services with varying degrees of compliance with the given dependability constraints. We perform two main refinement steps in cascade: (1) generate the virtual machines to host and protect the business service components, considering the required capabilities; (2) permit and protect interactions among service components, generating an abstract configuration for every involved network node (e.g. firewalls, load balancers). The tool is designed as an ontology-based model transformation engine. Transformation rules are written using XSL and executed by a standard XSL engine. This technological choice has been taken because XSL allows writing rules using set operation discouraging the usage of internal models: since its support to procedural programming approach is minimal. In this manner, rules are naturally forced to be simple and have to exploit ontology models as intermediate data repository, by adding artifacts instance to be used by the rule ex-

ecuted after. Finally, the integration that we did within an XML database permits a distributed approach with few efforts. The engine retrieves the model processed by the previous steps from the XML database and executes the set of transformations for the current level extending the internal model. In this paper, we analyse the main changes required to make the configuration rules virtualisation aware.

# 3 Dependability Techniques in virtualisation environment

## 3.1 Service provisioning

This section describes how virtualisation technology impacts the dependability techniques used for configuring servers and technical services to maintain business service continuity.

Our tool originally computed allocation of services on physical machines. When migrating to the virtualisation world, we calculate the allocation of virtual machines while considering the security requirements determined by the hosted services. In other words, the maximum of the hosted service for every dependability requirements is considered, in order to guarantee the correct level of support for all them. This choice must consider that, for example, internal resources are scarce (few machines available) and less efficient (provider is supposed to have a better connection to the Internet), but grant higher trust levels. Outsourcing VMs instead of services implies that the configuration of VMs' internals is not delegated to the hosting provider. Our tool computes such configurations, allowing stronger control on the system.

Depending on the service allocated, we classify VMs in two classes: *working machines*, performing operations on external data (e.g. application servers), and *storing machines*, saving data on their own disk images e.g. DBMS. In accordance to this classification, the configurations also describe whether each VM should be spawned in 'persistent' (their disk images are modified) or 'non-persistent' mode (when we shut down and restart the VM all the modification on disk images are lost). As detailed in the next sections, this approach reduces the impact in case of vulnerability, and helps in backup strategies.

**Service isolation and packaging** The configuration rules first isolate different classes of services to limit fault propagation, including exploitation of vulnerabilities. In traditional systems, isolation implies different servers, generating big additional costs. When our tool produces plans for virtual environment, it enforces isolation more aggressively due to reduced costs of VMs. Nevertheless, it considers the different strength between such logical isolation and a traditional physical isolation.

Splitting a service onto different machines allows minimising permissions. Imagine to have two services differentiated either by roles (one for restricted users, one for public), permissions (one just reading some contents, the other modifying them), or resources (one accessing the stock, the other using customer data). With no alternative allocation, we must expose both services to each other, delegate isolation to the operating system, and increase risks of violations of the more restricted service. In similar cases, our tool spawns two different machines: a machine accessible from the public, the other implementing session-level access control (TLS or VPN); a machine accessing read-only data, the other with read-write access to data. Similarly, imagine allocating two web applications on the only available Apache web server, one requiring a module that presents a well-known vulnerability. Our tool generates two different VMs, each one with the proper set of software dependencies.

In traditional systems, our tool considers service isolation costly from the communication performance point of view, due to the additional network interactions. When adopting VMs, the tool decreases this cost, since the interactions between VMs on the same physical device are generally faster than network communications. Naturally, to exploit this feature our tool calculates the VM groups, in order to minimise the communication delay w.r.t the business service model. As result, typically, these groups collect service components belonging to the same business service [13].

Some services reside on isolated hosts by design (e.g. , firewalls of different brands, front-ends and application servers). Our planner seeks server consolidation, implementing them as different VMs on the same physical machine, to speed up network communications, while preserving most of the security advantages deriving from isolation.

**Virtual machine replication** Replication of resources, typically with the aid of a central manager (e.g. load balancer), is a pillar of dependable configuration. Our planning tool exploits the chance to spawn dynamically virtual machines to implement countermeasures and mitigations to problems like vulnerability exploitations, denial of service attacks, failures and QoS degradation due to misuse or spikes in demand.

Menaces deriving from possible vulnerabilities are quite common in information systems. Typically, there are three different approaches to such problems: change the software implementation with an immune one, apply a security patch, or take the risk of a possible exploitation. We model software packages with their dependences and we can tag whether a software package is vulnerable or not. Thanks to this model, the planner mitigates the impact of vulnerabilities affecting service availability exploiting "diversity implementation" concepts [18]. In other words, our tools plan

a replication of critical service components, using different operating systems, with different software implementations, and subject to different vulnerabilities. Since in many cases the vulnerability affects a library imported and not directly the software package, this approach is effective when the two implementations do not share any library. Actually, this approach is most effective when service availability is crucial, but problematic for confidentiality requirements. In fact, the chance to get confidential data increases, due to the possibility that at least one replica has unknown vulnerabilities. In this case, the planner considers a solution based on cold/hot standby as more effective: the spare system with the alternative implementation is exposed only in case the standard one has become vulnerable. In particular, the tool prescribes the storage of disk images of VMs with alternative implementations (e.g. a Linux firewall and an OpenBSD one, a Linux web server and a Windows one).

**Data management** In a standard system, DBMS servers provide persistent data services to applications. In virtual environment, our tool determines at configuration time how many DBMS virtual machines are needed and why. In fact, since our models manage also the nature of the data of the applications, our tool plans to spawn different DBMS instances and split data belonging to different applications. This isolation allows enforcing access control also at network level by selectively authorising service component interactions. For example, we can suppose that applications $X$ and $Y$ need two different data sets, $L$ and $K$. Instead of allowing $X$ and $Y$ to communicate with the same DBMS (that contains $L$ and $K$), our tools spawn two different DBMS, hosting $L$ and $K$ respectively. Our tool will then compute the network configuration rules allowing interactions $X <->L$ and $Y <-> K$ (see 3.2). This 'defence in depth' mechanism can be expensive, but, if correctly balanced (e.g. right grouping of data applications), impacts positively on the overall system security.

Unfortunately, virtual machine technologies do not solve the problem of data loss. On the contrary, due to increased data splitting, we need to perform more backup operations. For this problem, our planning tool exploits the same approach of non-virtual environment. It assumes the use of separated or external file systems (e.g. NFS, SAN), storing data in dependable way (e.g. RAID), and performing replication/backup services. We classify virtual machines in working machines (e.g. application servers, web servers) that offer an execution environment and operate on remote data, and storage machines that use their disk image as permanent storage (e.g. DBMS, file servers). We spawn only the storage virtual machines in persistent mode, whereas working machines are spawned in non persistent mode. Therefore, the planner applies backup services only for storage virtual machines, spawned in persistent mode. When critical information is not easily separable from the other, it is less expensive than a full backup.

### 3.2 Enabling and protecting communications

For authorising service interactions in non-virtualised environment, we base on network topology and equipment capabilities; e.g. we look for packet filters placed between two interacting applications and select appropriate filtering rules to permit their traffic. Virtualisation adds more flexibility in redesigning network topology (at least internally to physical hosts), and in spawning capabilities where needed (e.g. displace a virtual firewall in front of a service). A basic security advantage of virtualisation is ameliorating service interaction protection in respect of traditional local networks. Integrity and confidentiality of internal traffic are often neglected and, as a result, the local network is exposed to several security risks: unauthorised network attachment or host compromise easily lead to jeopardise interactions (traffic sniffing, ARP poisoning, host impersonation, session hijacking etc. ) and services (vulnerabilities, poor controls etc. ). If we deploy two interacting services in two virtual machines on the same host connected by a dedicated virtual link, their traffic is no more exposed to external attacks. Actually, in virtualised environment, most of the security guarantees fall if the hypervisor misbehave (e.g. because of unintended design or tampering). In the following we basically assume the hypervisor as trustworthy.

**Filtering, proxing, balancing** In local area networks, we configure Virtual LANs (VLANs) to achieve host/service separation. This solution has some issues: (1) it requires VLAN aware switches, not always available in local networks (hosts connected to the same VLAN unaware switch are necessarily part of the same VLANs); (2) it specifies which hosts are part of the same virtual network but does not control which services are reachable from whom. Using local firewalls, i.e. firewalls collocated with server equipment, has several drawbacks too: (1) decreased performances due to hardware resource sharing with the application processes; (2) software dependencies problems and incompatibility with other installed software; (3) imperfect security due to application vulnerabilities and their propagation (if one of provided services is vulnerable, and privilege escalation is possible, the attacker could easily circumvent the local firewall). For virtualised environments, our tool designs alternative ad-hoc virtual networking to enforce host isolation. It creates dedicated virtual links for interacting virtual machines, it spawns separate virtual machines to host filtering and proxy services when needed, and uses VPNs to protect traffic flowing in/out a physical host (VPN configuration is discussed in the next section). The sim-

plest configuration template contains two virtual machines within a physical host: one for the application service and another for firewall services. Adding a separate virtual machine for firewall services solves problems related to dependencies and vulnerabilities, because application and firewall are now isolated. On the other side, performance problems do not increase significantly thanks to the efficiency of modern virtualisation technologies. The firewall performance depends on hardware capabilities (CPU and memory), operating system, network stack, and filtering mechanism implementation. Several improvements have been proposed to enhance virtualised networking performances [16]. The hypervisor analyses each data packet that arrive on the network interface and transfers it to the proper virtual domain. The packet transfer operation is quite complex and requires several memory operations (data allocation/deallocation). The [16] work improves this process modifying packet transmission and reception path. For high assurance, a common practice is using two or more firewall devices in cascade: if their platforms and operating systems are different, it is reasonable to assume that they have different vulnerabilities. However, this configuration increases service latency, energy consumption, and costs. For external network attachment, our tool deploys and configures two firewall virtual machines with different OS and filtering software on the same physical host, reducing energy consumption and saving costs. However, two physical hosts are still required for achieving resiliency against hardware failures. More complex virtual networking is needed when placing several application virtual machines on the same physical host. Virtual traffic filtering can be enforced either at the hypervisor or by spawning dedicated firewall virtual machines. Hypervisor has privileged access to network traffic, does not depend on virtual network configuration, and is much more efficient. However, as the hypervisor is the single point of failure it should be kept as simple as possible: e.g. packet-filtering rules can be enforced, but extended firewalling through reverse proxies does not fit. Our tool adopts the dedicated firewall virtual machine technique. First, the tool computes the feasible configurations (e.g. single or cascade virtual firewalls) in respect of security requirements. It then performs a trade-off reusing the same firewall virtual machines for several service interactions (resources optimization). Finally, firewall virtual machines are configured with a virtual network interface per each application machine that should be filtered. This approach requires configuring a specific IP subnet for each application virtual machine, but allows defining more specific packet filtering policies (per interfaces). The increased configuration complexity is masked by the planning capabilities of our tool, that computes the virtual interfaces to adopt, and generates the filtering rules for each of those interfaces. This task is achieved by extending the *communication authorisation* module and the *reachability* algorithm described in [6] to manage virtualised networks. Practically, the reachability algorithm finds the firewall interfaces interested by each communication between network nodes by traversing the network topology graph. Then the *communication authorisation* module selects and generates filtering rules for each virtual interface.

The strategies applied for firewalling can be extended to balance load among service replicas. In fact, common load-balancing techniques use a mix of routing, network address translation, reverse proxing, and name services. All but the last one works at the same level and uses mechanisms close to the firewall services. In the last case, DNS services are configured to use a pool of IP addresses, which correspond to the replicated services. Name resolutions have a validity that can be configured to implement rude load-balancing mechanisms. With a long validity, clients keep sending requests to the same server and the load-balancing is ineffective. However, if the validity period is too short, every client request may hit a different replica making session handling trickier. To achieve load-balancing via DNS (reverse-proxy) mechanisms our tool first deploys a DNS (reverse-proxy) virtual machine associated to a set of service replicas virtual machines; then it configures the virtual interfaces and virtual machines' IP addresses, and generates the appropriate DNS address pool (proxy rules). The DNS approach is also more flexible and scalable when the load-balancing task should be performed by two or more data centres. The primary DNS could balance the request addressing the data centre site and a secondary DNS service, located into selected data centre, could address the request to the available virtual machine. We are most interested in how balancing strategies interact with security mechanisms such as TLS and VPN channels, as discussed in next paragraph.

**Channel protection and Virtual Private Networks** A common solution to protect the confidentiality and integrity of interactions across trust boundaries is authenticating and ciphering data using TLS or IPSec technology. Both solutions require more computational resources, increase energy consumption, and introduce traffic overhead. In virtualised environments, we can allocate services that require channel protection in distinct virtual machines on the same physical host: as already explained in the previous paragraph, traffic isolation can be granted without traffic encryption. However, when one of the endpoints is outside the administrative domain (e.g. customers, 3rd party services), or when the endpoints could not be aggregated on a single host (e.g. for resource consumption), their traffic should be ciphered. To protect this traffic, our best practice is performing aggressive service isolation and creating a separate logical ciphered channel for each interaction. Our tool proposes a set of alternative configurations with different

trade-offs between level of isolation and resource optimization. Consider a three-tier web application interacting with two user categories, for example gold and platinum users. To address availability requirements and cost savings the presentation and application servers have to be outsourced to a 3rd hosting service. Instead, to meet stringent security requirements, the database should be kept internally to the company. The most common solution for protecting the data flow between application and database across the Internet (or Intranet) is to configure a Virtual Private Network (VPN). Often, in non-virtualised environments, IPSec is configured to tunnel the traffic between two gateways. The first problem is that the traffic between hosts and gateways is not protected and might be attacked. In addition, IPSec may be difficult to configure, especially key exchange services that may collide with firewall and address translation policies. Another solution is to configure a TLS based VPN, e.g. using OpenVPN[1], that is more flexible than IPSec and easier to configure. However, as for IPSec, the TLS VPN is often configured between two concentrators. Following the rules described in Sec. 3.1, in similar cases, our tool first provides users role isolation by splitting both the application and the database in a pair of virtual machines: one for gold users and another for platinum ones. The tool then computes two alternative solutions that rely on TLS VPN concentrator virtual machines. The first adds two TLS VPN concentrator VMs per each application-database pair. This configuration performs strong interaction separation, allows configuring a different IP subnet for every TLS VPN, and provides IP reachability only between interacting endpoints. In fact, if a VPN is compromised, only its traffic can be attacked, while other interactions are not affected. The second solution provides resource optimization by adding only two VPN concentrator VMs shared by both the application-database pairs. The VPN machines are always placed on the same physical host of the VMs originating the traffic to protect. As already discussed for local firewalls, isolating the concentrator in a separate virtual machine, limits attack propagation. To implement the configuration rules above, we extended our previous algorithms, originally described in [6], for the configuration of end-to-end TLS VPNs. In practice the tool: (1) computes the required VPN concentrator VMs; (2) computes the virtual interfaces to adopt; (3) generates VPN configuration rules for each of those interfaces. Finally, we reuse the *communication authorisation* module to permit TLS VPN communications generating the proper filtering rules. Note that, in non-virtualised environments, the proposed solution and security level would have unreasonable costs as we need to configure a host for each virtual machine and two TLS VPN concentrators for each interaction. Our tool can also configure balancing services for increasing dependability of the VPN concentrators. To achieve this, we can configure DNS to resolve the VPN service to the address pool of the available concentrators. In a more complex configuration, we also add an additional layer of reverse-proxy services acting as NAT-level load balancers and forwarding client requests to one of the available concentrators. In both cases, we assume that the load sharing mechanism is performed rarely: once an endpoint has joined the VPN, the concentrator does not change. If the current VPN concentrator fails, the client should rejoin, and the DNS (and reverse-proxy) resolves to another concentrator. At last, to better secure traffic between the company and the provider's data centre (e.g. prevent traffic analysis), our tool suggests encapsulating the TLS VPNs into an IPSec tunnel. This step uses the same configuration rules originally described in [6].

We now focus on how managing secure connections (e.g. HTTPs) in case of virtual service replicas. As presented in 3.2, our strategies for placing virtual firewalls offer reverse-proxy and balancing capabilities. In fact, the use of a TLS capable reverse-proxy is the simplest solution. Every client contacts the reverse proxy that forwards the client request to one of the application virtual machines configured in the pool. Our tool generates a set of alternative configurations using a TLS reverse-proxy virtual machine. One of the simplest configurations is to displace, on the same physical host, the reverse-proxy machine and two or more application virtual machines. In this case, only the traffic between the proxy and the external world is ciphered. The interactions between the proxy and the replicas are configured as logical links using virtual interfaces as described for firewalls. When replicas are allocated to multiple physical hosts, the traffic between proxy and remote replicas is ciphered via the VPN techniques described above. Unfortunately, the reverse-proxy solution does not scale to large installations and multiple systems. We must introduce DNS-based load sharing mechanisms that requires additional issues to be solved. Even if the DNS resolution validity period is correctly configured, when it expires, a new TLS negotiation may be required. Most notably, this happens with HTTPs sessions and the use of TLS session ID. To efficiently solve the TLS balancing problem we must address two phases: negotiating a new TLS session, and resuming pre-negotiated sessions through the session ID mechanism. The straightforward solution is using a backend engine that performs TLS negotiation and/or caches session data. For example, a relay like [20] allows a server to resume a TLS session negotiated by any other one. A similar solution that implements a centralised session cache is the "distcache project"[2], supported for example by the Apache web server. This tool can be installed directly on application server replicas (e.g. distcache is supported by the Apache web server), or on reverse-proxies / TLS relays

[1] http://openvpn.net

[2] http://distcache.sourceforge.net

to load-balance TLS interactions. We also consider more distributed solutions like the TLS session tickets specified in [21]. They provide a mechanism to store session data on the client side, in the way web cookies work. Basically, the server seals the session state into an encrypted ticket and forwards it to the client, which uses this information to resume the session against any server in the pool. Similarly to the previous solutions, session tickets are directly supported by some application servers (e.g. Apache), and can be exploited at TLS proxies to enable distributed session resumption. When selecting a server-side approach, our tool allocates replicated virtual machines for DNS services, and one or more VMs for TLS engine. Instead, for the client-side approach, the tool configures the shared keys, to protect the TLS session ticket, on each virtualised replicas.

**Monitoring and logging** Monitoring and logging are relevant tasks to trace system behaviour and to highlight problems. For security purposes, they are useful to check host and service availability, to detect network attacks, to ensure non repudiation. In non-virtualised environments, monitoring tasks often rely on dedicated hosts and networks for better performance of application services and better isolation of monitor ones. However, to report host/service status, monitoring systems also install agents on each monitored host (e.g. Nagios[3]). To contain costs and management activities, every agent or network probe collect and sends a subset of information to a centralised host, the aggregation point that analyses the reported data. This approach has its dependability and scalability limits in the aggregation point. In virtualised environments, the monitor and logging tasks can be distributed more accurately to achieve a multi-level and dependable architecture with reduced costs. Best practice for securing monitoring and logging tasks is using stealth components, to hide the presence of agents and network probes and preserve them from security attacks. In virtualised systems, placing monitoring functions at the hypervisor can hide and protect monitoring services. On the other side, as already discussed for firewalls, the trade-off is the increased complexity of the hypervisor.

Our strategy is to install an agent on every virtual machine in order to collect information like network reachability, received network packets, and service status. A set of virtual machines able to collect reported data, are located on different physical hosts and configured in load-balancing mode to achieve high availability. These virtual machines, to better address administrator objectives, are configured to perform all monitoring/logging tasks or a specific task: for example, collecting only network reachability data. We can deploy another set of virtual machines (in load-balancing configuration) able to analyse the reported data, in order to build the network model, trace network behaviour, and evaluate security risks.

[3]http://www.nagios.org

# 4 Configuration Algorithms

## 4.1 Configuration and Re-configuration

The configuration generation process is based on the requirements (functional and non-functional) of the service(s) to be hosted by the information system. It refines these requirements, exploring the alternatives exploitation of available resources. These computations originate a Directed Acyclic Graph (DAG), where every node represents a partially refined configuration. Every step of the process, performs the refinement for every partial configuration, reading all the information generated by previous phases. Some steps generate directly element to be used for the final configuration, whereas others compute artefacts that are useful only for the steps executed after. When all the refinement steps are completed, the final leaves of the configuration DAG represents the final configurations to be enforced while configuring the information system. Some configuration will presents different level of residual requirement to be managed by the actual devices. If the requirements are high, the system recommends the usage of high availability devices; ignoring these recommendations causes a configuration with lower dependability of the system.

In order to react to emergency states, some rules perform a linkage between configurations. In other words, on a configuration, if occurs an event that could compromise the achievement of the requirements. Reaction consists in performing a configuration switch to alternative refinements, where such risky events do not occur or their effects are negligible or tolerable. In fact, faults or vulnerability exploitation make affected components unusable. Such links are named “Reaction” and links the problem in a configuration to the novel configuration to be adopted. The target (emergency) configuration is chosen among alternatives as the one with less differences compared with the current one, to minimise the reaction time. This retrieval is performed, searching the configuration that is not excluded by the fault, with the maximum number of choice of refinements in common: this operation is simple since the configuration DAG maintains tracks of every choice.

The class diagram in Fig. 1 represents the data used in the process. The dashed boxes discriminates the different supplier of such data.

Administrators supply as input the list of the service components, “Service” in the diagram, and, for each of them, the different implementations of the service, “Implementation”. Every implementation is composed by one or more software packages, with their dependencies. We refer to functional requirements to indicate the implementa-

tion together with the Service Access Point and the need to permanently store data or not. Furthermore, administrators assign values to non functional (dependability) requirements, classified on a scale of LOW, MEDIUM and HIGH: *availability* (how much is important that a service is available?), *confidentiality* (does the service manage classified data? How much it is important to preserve their confidentiality?) *integrity* (how much is serious if the service or the service data are tampered?). Other requirements could be added, but the extension must supply also additional rules to the process to refine them. Since the requirements are assigned to all the elements, we created an abstract class that is extended by every object subject to requirement. The adoption of a coarser scale for such requirements is supported and possible, if the rules are changed to understand it. Our concern is the benefit of a more detailed scale would not confuse the network administrator, without an actual gain in terms of more precise configurations.

The process exploits also the *SoftwareDatabase* that contains the data on the dependencies and conflicts among software packages. It is built on the base of package managers' repository, like RPM.

Finally, the box *Refinements* collects the elements generated during the process, described in the following sections. During the description of the process the meaning and the usage of the classes will be clarified. Such elements are grouped together to create different configuration (in the diagram we connected only the superclass Component for readability purposes). Furthermore, some components are subject to event that causes a reconfiguration tasks (a change from a configuration to another).

### 4.1.1 Service provisioning

The purpose of this step consists in defining the compatible software group, using the implementation of the different service components. Each of them presents ore or more implementation whose software requirements are solved by means of the software database.

First, the algorithm generates different software packages groups (containing the application software and its dependencies) for every service. If the dependency can be satisfied in different manners, alternative groups are generated. They derive the software requirements of the services implementation. Then, additional groups are added merging such basic groups, if the software packages composing them, including the dependencies are is compatible, i.e. there are not packages in conflict. Actually, if the confidentiality or integrity is HIGH or they have a different value, or if their availability is HIGH, groups are not merged, otherwise the group takes the maximum for every dependability requirement.

Moreover, requirements are inter-related. The tool propagates the requirements of availability to the software implementation (and, of course, its dependencies) into requirements of confidentiality and integrity. We propagate the requirement of integrity as confidentiality, and vice-versa. Such requirements are propagated with a level lower that the originating one, for example an availability=HIGH will become a confidentiality=MEDIUM and integrity=MEDIUM, since they are derived requirements.

Listing 1 presents the implementation of the task that is composed by three main parts. In *group initialisation*, a different group for each implementation is created. In *dependency retrieval* the tool completes the dependency by means of software database, and in case of different alternatives to satisfy, it creates different groups, without conflicting software packages. In *merging groups*, if compatible originates additional composed groups.

**Listing 1. Allocation computation**

```
//Group initialization
foreach service in Service.getInst()
  foreach impl in service.impl()
  Group g = new Group(service, impl)
  foreach sw in impl.getSw()
    g.add(sw);

//Dependency retrieval
foreach g in Group.getInst()
  foreach sw in g.getSoftwares()
  Impl alternativeDeps = SoftwareDB.getDeps(sw);
  foreach alternativeDep in alternativeDeps
    // if the software to add do not presents any
         conflict ...

SoftwareDB.getConflicts(alternativeDep.getSoftwares()).
     intersect(g.getSoftwares)
==null
    Group gNew = g.clone();
    gNew.addAll(alternativeDep.getSoftwares);

//Merging groups
foreach g1 in Groups.getInstances()
  foreach g2 in Groups.getInstances()
  if (g1 != g2)
    if (g1.isCompatible(g2))
      Group g = Group.merge(g1,g2);
```

The generated groups of software packages are aggregated together to compose different solutions at application level, with at least one group for every service. If the service requires MEDIUM or HIGH availability, it is possible get more groups, with a lower level of availability, implementing a redundant solutions or maintaining only one group that will be than achieved or with a VM duplication or with HA devices.

In this phase, also groups with different implementation or dependencies for the same service are generated, to implement a diversity design strategy. In this case, the propagation from the availability to the other requirements is computed subtracting two levels, since to definitely compromise the availability of the service, it is needed to find a way to compromise every implementation. On the contrary,

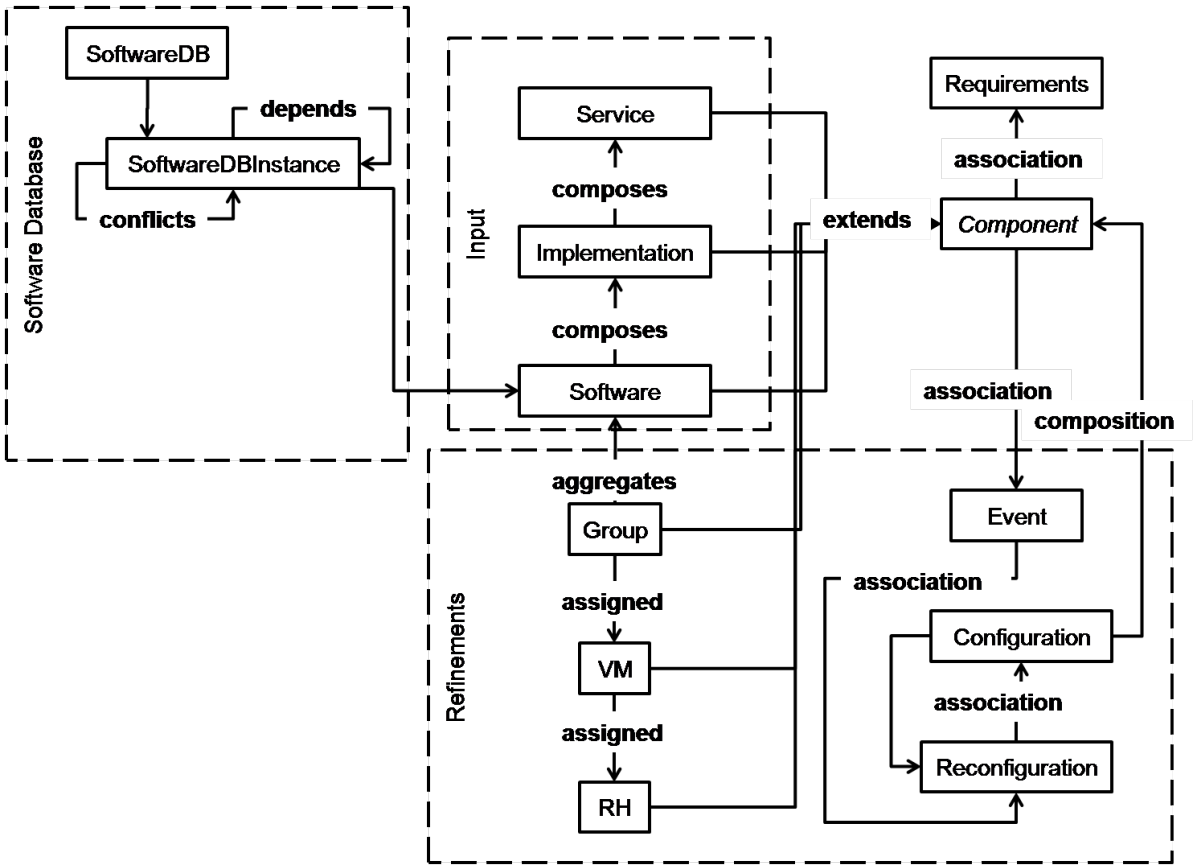



**Figure 1. Class Diagram for the refinement algorithms**

confidentiality requirement is augmented of one level, since the violation of single software is sufficient to access to critical condition.

A more complex analysis is necessary for integrity issues, in fact, a different implementation may help in recognising when integrity is violated (same service returning different results) but for understand the correct results it is necessary having at least three implementation, to masquerading an integrity violation on one group, as explained in [14]. Actually, to implement this measure, it necessary having ad-hoc modules to implement a voting system. Since such modules are not common, integrity requirement propagation is not modified by diversity implementation.

Additional configurations are generated, to be used in emergency conditions. Our tool generates configurations with only the service with availability=HIGH, configuration with all the service with availability=HIGH and some or all service with availability=MEDIUM and configuration with the entire MEDIUM and HIGH availability service and some low availability. Indeed, when a configuration is chosen, the best choice is the one that offers as more service as possible, but the reactions could use also others to maintain at least the critical service available.

Listing 2 brings the results of Listing 1. It performs the configuration generation, providing every possible combination of the group, collecting in conf instances.

**Listing 2. Configuration generation**

```
// Configurations Initialisation
if (Configuration.getInst()==null)
  foreach service in Service.getInst()
    foreach impl in service.getImpl()
      foreach g in impl.getGroup()
        Configuration conf = new Configuration(g)

//Configurations generation
foreach service in Service.getInst()
  foreach conf in Configurations.getInst()
    foreach impl in service.getImpl()
      foreach g in impl.getGroup()
        Configuration conf1 = conf.clone()
        conf1.add(impl);
        //supposes that conf is inserted into the
Configuration register only if no other conf with the
     same impls is already
available
```

#### 4.1.2 React to new vulnerabilities discovered

It is common that new vulnerabilities are discovered when the system is already online. Typically, such vulnerabilities are available on vulnerability databases like National Vulnerability Database [4]. The tool matches the vulnerability with the software available in every system and, on the base of the impact they have, plans reactions. In fact, vulnerabilities entries present also a Common Vulnerability Scoring System [5] evaluation of the impact in term of confidentiality, integrity and availability.

In practice, the configuration generator retrieves all the vulnerabilities affecting every software package in the system and crosses the menaces of exploitation with the dependability requirements. Even if, it is generally a good practice having as few open vulnerability as possible in the system, the tool is concentrated to maintain satisfied dependability requirements. In other words, if a software package presents a vulnerability affecting its integrity, and the requirements for the group of software packages are for availability it does not consider vulnerability exploitation a problem. A natural objection to this method is: "What? Such requirements are linked. Integrity gives confidentiality, integrity and confidentiality gives availability...". We agree with this assertion, in fact we consider these issues in the propagation of the requirements throughout the configuration phases, and, they are already evaluated w.r.t. the interdependency among them.

Typical reaction to vulnerability consists in switching (when possible) to an alternative implementation, updating/patching menaced software packages, or turn off the dangerous application. If there is an alternative configuration that is not affected by the vulnerability in object, the reaction implies a configuration change. Unfortunately, in many cases, such solution is not feasible and we have not any alternative implementation. Consequently, the model of software and software dependencies needs to be updated with the last version of the products, which hopefully will be not affected to this. Then, the tool generates additional configurations, which are added to the configuration tree. At this point, we plan a configuration swap to one of the new configuration generated. In case of none of the previous solution is possible, it remains or tolerating the situation (if the availability of the service is crucial) or switching to a configuration that does not use to software, even if some service are not available. This last approach is to be considered transitional, and the update of the software model is performed regularly, in order to generate a configuration with all the services available again as soon as possible.

Listing 3 computes the reaction in case of vulnerability discover. For each requirements associated to a software, an event of type VULNERABILITY is set. If there is a different implementation of sw, the reaction performs the configuration swap to the corresponding configuration; otherwise, it excludes the sw from the usage (again as configuration change) or tolerates the vulnerability on the base of the availability requirement.

**Listing 3. Reaction to vulnerability discover**

```
//Reaction to vulns
foreach conf in Configuration.getInst()
  foreach impl in service.getImpl()
    foreach g in impl.getGroup()
      foreach sw in g.getSoftware()
      reqs = sw.getReqs()
        foreach req in reqs
          if (req.getValue() == HIGH
||req.getValue() == MEDIUM)
        Event vt = new Event(sw, VULNERABILITY
,req);
       //finds an alternative configuration, with the
service implemented by the sw available, but without the
      sw vulnerable
      alternativeConfs =
ConfigurationDag.filter(!contains(sw)).filter(!contains(
     service)).getCloser(conf
);
      if (alternativeConfs!=null)
        Reaction r = new Reaction (vt,
alternativeConf);
      else
      // if the requirement availability is not HIGH,
whereas the others are HIGH or MEDIUM, or the
     availability is HIGH, while the
it is we plan to switch off the service
        if (reqs.getAvailability().getValue !=
HIGH || (reqs.getAvailability().getValue != MEDIUM &&
     req.getValue()== MEDIUM))
         //finds an alternative configuration,
but without the sw vulnerable (but this will not have
     the service availabel)
          alternativeConfs =
Configurations.getInst().filter(!contains(sw).filter()))
     ;
          Reaction r = new Reaction (vt,
alternativeConf);
```

#### 4.1.3 Virtual machine architecture

VM generation considers the groups of software of every configuration generated by the previous step, together with their dependability requirements. First, for every group is created a separated VM. Second, if a group presents a high level of availability, more than one VM can be assigned, with a lower availability requirement. Depending by the service, it is indicated if the machine needs to store permanently on its own file system or not, to discriminate between working machines and storage machines.

The dependability requirements of the VM are used to eventually choose among different VM templates. If the availability requirement is high, the performance of the VM must be coherent. If the confidentiality is high, it will require VM templates with additional protection, like personal firewalls or disk-image encryption. VM integrity will imply, again, protection from network, like personal firewalls, but also digital signature of disk-images. The adoption of trusted computing techniques will be also useful to

satisfy such requirements.

The VM resulting from this computation derives the requirements from the software packages of the corresponding group, which are used to plan the deployment of the machine on virtualisation environment. Also VM configurations for emergency conditions are generated, to be linked by reactions, as we see in the following sections.

For this work, we consider having different templates of virtual machines, classified on the base of their capability to satisfy non-functional requirements. First, the machine can present a classification on the base of their performances, to answer to availability requirements. Then, they can have an operating system implementing Mandatory Access Control (MAC) that benefits confidentiality and integrity. They can offer the chance to spawn only digitally signed disk images. They can have one or more personal filtering software packages (e.g. proxy, firewall) to limit the access from external. They can offer cryptographic primitives to stores only ciphered data that increase the confidentiality, but decrease the availability. Unfortunately, the match between these capabilities and the dependability requirements is not one-to-one, and, furthermore, some mechanisms useful for requirements are counter-productive for others. For this reason, over dimensioned configurations are generated, and, only after considering all mechanisms, the configuration are chosen.

Actually, the configurations candidates to be applied are the ones with the lower requirements remaining. In fact, machines with higher specifications cost more. Another important factor is the number of machines, the lower it is, and the cheaper is the solution. More expensive, but promising, configurations are kept into account to perform reactions in case of problem.

Listing 4 presents the generation of VM starting from groups. In this part is crucial the match between the dependability requirements of the groups and the service offered by the VM. In the first part, *generation of vm* , the availability is mitigated generating different configurations with a varying number of VM assigned. Availability requirements are coherently adjusted, to consider the redundancy. In the second part, *mitigation of requirement thought VM capabilities*, creates different configuration, for every VM, adopting a different technical solution. Note that, since this process is executed more times, also combinations of different solutions are possible. MAC means mandatory access control, TC trusted computing techniques, CRYPTO cryptographic primitives, FILTERING a personal firewall system.

Typically, the best configurations are the one whose requirements on VMs are lower, since it implies

**Listing 4. VM generation**

```
//Generation of vm
foreach conf in Configuration.getInst()
  foreach impl in service.getImpl()
    foreach g in impl.getGroup()
      //Maximum number of machines to be created
     if (g.getReqs.getAvailability().getValue == HIGH)
          limit
= 4;
     else if (g.getReqs.getAvailability().getValue ==
          MEDIUM)
limit = 3;
     else limit =2;

     for (n_machines=1; n_machines++; n_machines<=limit)
       conf_new = conf.clone();
       for (i=1; i++; i<=n_machines)
         //get the correct VM template
        vm = new VM (group, conf_new);
        vm.setReqs (group.getReqs());
         // decreases the availability, one level
every additional virtual machine
        vm.decreaseAvailability(n_machines)

//Mitigation of requirement thought VM capabilities

foreach vm in VM.getInstances();
conf = vm.getConf();
if (group.getReqs().getIntegrity = HIGH)
  vmt = vm.clone();
  vmt.add(MAC);
  vmt.getReqs().decrease(INTEGRITY);
if (group.getReqs().getIntegrity = HIGH)
  vmt = vm.clone();
  vmt = vm.clone();
  vmt.add(TC);
  vmt.getReqs().decrease(INTEGRITY);
if (group.getReqs().getConfidentiality = HIGH)
  vmt = vm.clone();
  vmt.add(CRYPTO);
  // increases the integrity, since the requirement of
       confidentiality is
mitigated if the VM is integer
  vmt.getReqs().increase(INTEGRITY);
  // increases the availability requirements, since
       cryptography decreases
performance
  vmt.getReqs().increase(AVAILABILITY);
  vmt.getReqs().decrease(CONFIDENTIALITY);
if (group.getReqs().getIntegrity > MEDIUM or group.
     getReqs().getConfidentiality
> MEDIUM)
  vmt.add(FILTERING);
if (vmt.is(CRYPTO))
  avail = vmt.getAvailability ();
if (avail == MEDIUM)
  vmt.setAvailability (HIGH);
if (avail == LOW)
  vmt.setAvailability (MEDIUM);
```

#### 4.1.4 React to service performance degradation

When performance for a service is not satisfactory, due to peaks in workload or a Denial of Service Attack, the reaction consists in adding computational resources to the stressed service. Another solution consists in using VM with higher performance, instead of the original ones. Alarms derive from sensor put on the VM, and could, for example, high CPU or memory consumption. Another alarm is typically required, to advertise that the solution is oversized (for example low CPU load) to react back to normal

usage conditions.

To generate this reaction plan, the tool generated additional configuration for virtual machines that uses a higher number of VMs then in normal condition, or VM with more computational resources, also for the configuration that do not have all the service active. At this point, depending by the availability needs of the VM, we generate a reaction to the proper configuration: if availability needs is set to HIGH tools prefer a configuration with an additional VM, also if this implies losing some service (with an availability less than high); if availability is MEDIUM it generates a configuration switch to use an additional VM only if it has the same available services of the starting one, otherwise it plans the usage of a more powerful machine; if the availability is LOW, it plans the usage of a more powerful virtual machine, if available, otherwise it performs no reactions.

Listing 5 presents the computation of reaction to low performance on a virtual machine. If the availability is HIGH, we switch to another configuration with a higher number of machines, also decreasing the number of machines without considering other machines. When the availability is MEDIUM, it switches only to configuration with the same number of service available.

**Listing 5. reaction to VM performance degradation**

```
// React to performance degradation
foreach vm in VM.getInstances()
  g = vm.getGroup();
  impl = g.getImpl();
  // for each configuration in which the vm is involved
    foreach conf in vm.getConfs()
      /// in case of performance low, pass to a
           configuration
with an higher number of machines
     if (vm.getReqs().getAvailability == HIGH)
       Event e = new Event(PERFORMANCE_LOW, vm);
       alternativeConf =
ConfigurationDag.filter(.getImpl().getGroup().getVM().
     count >
.getVM().count()).getCloser(conf);
       Reaction r = new Reaction (e, alternativeConf);
     if (vm.getReqs().getAvailability == MEDIUM)
       Event e = new Event(PERFORMANCE_LOW, vm);
       alternativeConf =
ConfigurationDag.filter(conf.getImpl() ==
.getImpl()).filter(.getImpl().getGroup().getVM().count >
.getVM().count()).getCloser(conf);
       Reaction r = new Reaction (e, alternativeConf);
```

### 4.1.5 React to VM integrity violation

As already said, integrity violation implies more complex reactions, but virtualisation environment helps.

First, the tool distinguishes in storing and working machines. For working machines that are spawned in non-persistent mode, a first measure consists in rebooting the machine.

On the contrary, for storing machines, we can react swapping to a configuration in which the violated machine is not longer used. If this violation is caused by vulnerability exploitation, and the integrity is high, the system could also consider switching off the machines affected, until a reaction for the vulnerability is not put in act. Another option offered by the tool consists in changing the configuration to another one with a stronger integrity protection for the machine.

Listing 6 pseudo-code shows the implementation of this step. It creates an event of type INTEGRITY_VIOLATION for every VM with an integrity grater or equal to MEDIUM, and attaches to it a reaction to configuration where the integrity requirement is lower. In case of lack of proper alternatives, if the requirement of availability is lower than the integrity one it react passing to a configuration that does not use the tampered machine, otherwise it tolerates the violation, without any change to configuration.

**Listing 6. reaction to VM integrity violation**

```
foreach vm in VM.getInstances()
  g = vm.getGroup();
  impl = g.getImpl();
    foreach conf in vm.getConfs();
     if (vm.getReqs().getIntegrity() > MEDIUM)
     /// in case of integrity violation, search a vm,
          with
the same group but with a lower integrity requirements,
     Event e = new Event(INTEGRITY_VIOLATION, vm);
     alternativeConf =
ConfigurationDag.filter(vm.getGroup().getVM().
     getIntegrity() <
group.getVM().getIntegrity()).getCloser(conf);
     if (altenativeConf!=null)
       Reaction r = new Reaction (e, alternativeConf);
     else if (vm.getIntegrity()>vm.getAvailability)
       // if not available and integrity is greater
than availability, switch to a configuration that does
     not use the vm (switch
off the vm)
       alternativeConf =
ConfigurationDag.filter(!.contains(vm)).getCloser(conf);
```

## 4.2 Enforcing security controls

The refinement process, as defined previously, is represented as a directed acyclic graph. The general workflow to enforce security controls is composed by the following steps: (1) evaluation of the suitable and available technologies; (2) refinement of directly and indirectly security controls; (3) generation of alternative configurations. Each step refines the information provided by the previous ones and populates the graph adding a new level for each step. Each path between the root and a leaf represents an enforceable solution. Alternative configurations are represented as different nodes of the same level.

The first step analyses the security requirements to derive the suitable technologies to protect communication traffic.

Our model associates CIA properties (confidentiality, integrity and availability), expressed as LOW, MEDIUM or HIGH to a set of suitable technologies and related modes. For example, if confidentiality and integrity are set to MEDIUM the model suggests that the available technologies are TLS, IPSec and TLS VPN. Otherwise, if confidentiality and integrity are set to HIGH, the model suggests adopting IPSec or TLS VPN. On the contrary, the availability property identifies when a service has to be replicated adopting load-balancing techniques. The association between security properties and technologies is defined into technology templates, expressed using an XML based language. Such templates also contain technology-specific modes to support alternative configuration. For example, the IPSec technology-specific template contains three different configurations: tunnel, transport and remote access.

After analysing suitable technologies, the tool analyses network components capabilities to evaluate which technologies can be directly or indirectly enforced. A technology is directly enforceable if exist a set of services or devices which support the related capability (e.g. IPSec, TLS, filtering). Otherwise, if no service or device is able to support the technology we should enforce it indirectly installing a new feature or adopting virtualisation techniques. Our choice consisting providing a virtual machine with the required features in order to make this process more flexible, secure and feasible. The benefits derived from this strategy are described in details in the next sections. Our tool supports filtering and channel protection security controls which can be both directly or indirectly enforceable. This process is divided into two sub-steps: network component analysis and communication policy analysis. The first step takes in input the network description and classifies hosts, routers, firewalls and services information to discover network components features and capabilities. The second step analyses communication components to identify network components involved in the policy. More precisely the algorithm transforms the network in a graph, finds all paths between policy elements (source and destination) and identifies the security components (which have security capabilities, e.g. filtering, channel protection) involved in the communication. When filtering or channel protection capabilities are not available, our tool adopts a set of virtual machines with configurable security controls to enforce the policy. Such VMs have to be added to the output VM architecture for deployment on virtualised providers.

The adoption of virtual machines allows enforcing a security control on demand, satisfying policy requirements. First of all it is necessary understanding which security controls should be enforced using the virtualisation technology. For example, considering the objective to enforce IPSec in tunnel mode for protecting traffic between two servers. The tunnel mode requires the adoption of two IPSec gateways to enforce the tunnel (as defined in the IPSec technology-specific template). Evaluating security components involved in the policy the algorithm find only one suitable gateway. To enforce the policy two general solutions are available: (a) deploy a virtual machine as IPSec gateway; (b) deploy two virtual machines, one for each server. However the problem is quite complex. When an administrator configures IPSec in tunnel mode, the traffic that belongs on untrusted network (e.g. Internet) must be ciphered. This simple consideration entails that each gateway is displaced internally to a trusted network or at its border. Therefore, the use of a gateway displaced in untrusted network partially invalidates the benefit of ciphered channel. However, if an IPSec gateway is displaced in untrusted network, a possible solution is to deploy two virtual machines (in the trusted networks) and not configure the other gateway. The deployment of a virtual machine requires a host able to support virtualisation. To identify in the network which are the available hosts we define the virtualisation feature as a capability. Therefore, when the algorithm performs network description analysis identify the hosts which support virtualisation. At previous step the algorithm decides which virtual machines are necessary then it evaluates, using a set of strategies, which hosts are selected to guest VMs.

This approach requires to analysing allocations finding which hosts guest allocated VMs. To maximize performance a physical host supports a limited set of virtual machines depending on its hardware features. Hence if the host supports the required VMs to enforce security controls the algorithm plan to add the new VMs. Otherwise our strategy is to find the nearest suitable host.

Once the tool decides which security controls should be adopted to protect communication traffic it is necessary to identify their displacement. Our approach is to group together services and security controls as indivisible block using a predefined template. Practically a template contains a set of predefined VMs to perform security controls which can be enabled or disabled depending on requirements (e.g. TLS VPN VM and firewall VM). In addition this strategy allows configuring the related virtual network in more simple way.

#### 4.2.1 Security controls templates

Once the tool derives the required security controls, the related VMs templates (depicted in Fig. 2) are used to generate alternative configurations which should be finally deployed on virtualisation platforms. The simplest template is *ServiceTemplate* that contains three virtual machines: *VM_Service*, *VM_FW* and *VM_VPN*. The first virtual machine is adopted to host the service and it is mandatory. The *VM_VPN* instead can be activated on demand to protect ser-

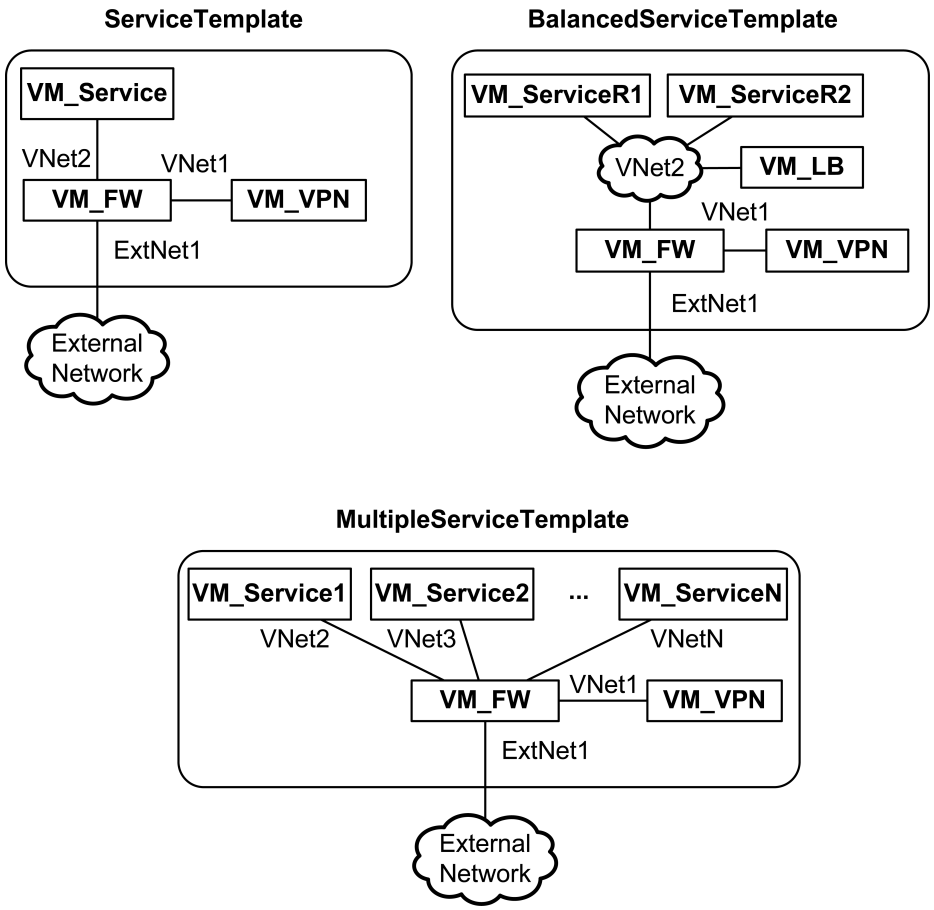



**Figure 2. Security controls VMs templates**

vice traffic. It can be implemented using IPSec or TLS VPN technologies as alternative solutions. The *VM_FW* is also mandatory and it is responsible to filter traffic among service, VPN and external network. As depicted in Fig. 2 this virtual machine is connected in bridged mode to the external network (*ExtNet1*), the local sub-network of the host, and to other VMs using two virtualised networks (*VNet1* and *VNet2*). This logical configuration allows separating and masquerading external and internal (virtualised) networks. For example, to allow traffic between the external network and the service, on the filtering virtual machine we can add a port forwarding rule, masquerading internal network. In addition, when *VM_VPN* is active, it is possible to derive a set of rules to: (a) allow only ciphered traffic between external and *VM_VPN*; (b) allow unprotected traffic only between *VM_VPN* and *VM_Service*. The security requirements, defined as input are refined using templates. For example, if the confidentiality requirement of service is set to HIGH, in the resulting template the *VM_Service* and *VM_VPN* are set to MEDIUM.

The *BalancedServiceTemplate* introduces the load-balancing functionality adopting the *VM_LB*. This, often implemented as a reverse-proxy, is in charge to dispatch traffic (protected or not protected) among replicas to enhance service availability. The application of this template is mandatory when in the configuration there are more software package groups for one implementation. In that case the availability requirement on *VM_LB* is set to HIGH. An interesting extension to ameliorate service availability is to add another load-balancer VM. This can be placed in standby and resumed as soon as the first load-balancer fails. In order to decrease the number of virtual machines and to aggregate functionalities the filtering and reverse-proxy could be grouped together as a *VM_PRX-FW*. An interesting extension of this template is to introduce load-balancing feature for VPN traffic. In a simple scenario we can substitute the service replicas with VPN virtual machine replicas enhancing VPN service availability. In more complex scenario the objective could be to improve availability of application and VPN services. In that case, we can modify the template to insert two different load-balancers: one for application and another for VPN replicas.

Finally the last template *MultipleServiceTemplate* is suitable to displace different service virtual machines on the same physical host. In this configuration each service is directly linked to the *VM_FW* using a virtual sub-network with specific network address and the VPN gateway is shared to different services. However it is possible configure the virtual machines to belong to the same sub-network. Grouping together the service VMs allows optimizing their allocation enhancing general performance introducing a security drawback. In fact, each service is protected using the same VPN channel and if the VPN is compromised, all traffic could be jeopardized. To reduce the related risk the template could be modified to support different VPN gateways. Practically we can displace a shared VPN gateway for non critical service and a specific VPN VM for each critical service.

#### 4.2.2 Security controls transformations

Once the tool selects the set of templates to adopt, it performs a set of transformations to refine the provided information. A transformation $T$ is an operation that takes as input a set of services, a template and security requirements and produces a set of alternative configurations. A subset of security controls transformations is depicted in Fig. 3. A configuration is expressed using an XML based language and contains: (a) the VM components; (b) the virtualised and physical network description; (c) the policies to enforce. The VM components and the virtual network links are derived from the set of services and template, then the virtual and physical network configuration are computed consequently. The firewall virtual machine is directly connected using a bridge to the physical network. The related external IP should be assigned accordingly to physical host subnet evaluating network description. The algorithm, after a network analysis, proposes to the administrator a set of available IPs. For each virtual links the algorithm computes the network configuration for each virtual machine. In practice it selects the addresses range from a table that contains the allocations for private networks (populated considering the [1]) and generates other network information (gateway, DNS). Finally, the tool generates the related policies for the provided solution. This process is composed by two steps: (1) generation of channel protection policies; (2) generation of filtering policies. Considering the selected security technology and the service description, the algorithm generates the related properties for each virtual machine related to a protected communication. For example, if the selected technology is HTTP over TLS, the TLS properties (e.g. cipher suite, server authentication or mutual authentication, etc. ) are attached to the service virtual machine component configuration. In similar manner, when the tool selects TLS VPN or IPSec, the algorithm attaches the related properties to the VPN component. Finally the algorithm evaluates the components security properties and virtual network configuration generates the filtering policies. In practice, the firewall VM is configured to forward traffic between external world and internal services according to security properties. For example, if the internal service is protected using HTTP over TLS, the firewall policy forwards only HTTPS traffic between external and internal networks. To clarify the security controls refinement we discuss the transformations of Fig. 3. The most simple transformation is identified by (a) and generates a configuration that contains a service and a firewall virtual machines. The tool (1) defines the properties of the virtual network that links service to the firewall; (2) generates a port forwarding policy to allow external traffic reaching internal service using specific protocol and port. The algorithm analyses the table of private addresses and provides the network configurations for the virtual machines shown in listing 7. For example, let us consider that the service is a web server that uses HTTP protocol on port 80: the firewall (*VM_FW*) is configured to forward traffic from external network to the internal service as described in listing 7.

**Listing 7. Network configuration and policies for (a)**

```
<VM id='VM_Service'>
  <network-interface value='eth0'>
    <addr type='IPv4' value='10.10.10.2' />
    <netmask value='255.255.255.0' />
    <gateway value='10.10.10.1' />
    <dns value='ask to admin' />
  </network-interface>
</VM>

<VM id='VM_FW'>
  //bridged interface
  <network-interface value='eth0'>
    <addr type='IPv4' value='ask to admin' />
    <netmask value='ask to admin' />
    <gateway value='ask to admin' />
    <dns value='ask to admin' />
  </network-interface>
  //VNet1
  <network-interface value='eth1'>
    <addr type='IPv4' value='10.10.10.1' />
    <netmask value='255.255.255.0' />
    <gateway value='ask to admin' />
    <dns value='ask to admin' />
  </network-interface>
  <policies>
    <filtering-policy id='1'>
      <type='forwarding' />
      <src ip='*' proto='tcp' port='80' />
      <dst ip='10.10.10.2' proto='tcp' port='80' />
      <action value='allow' />
    </filtering-policy>
  </policies>
</VM>
```

The (b) example demonstrates service load-balancing. In this case the tool generates network configurations for: (1) the sub-network *VNet2* that contains the load-balancer

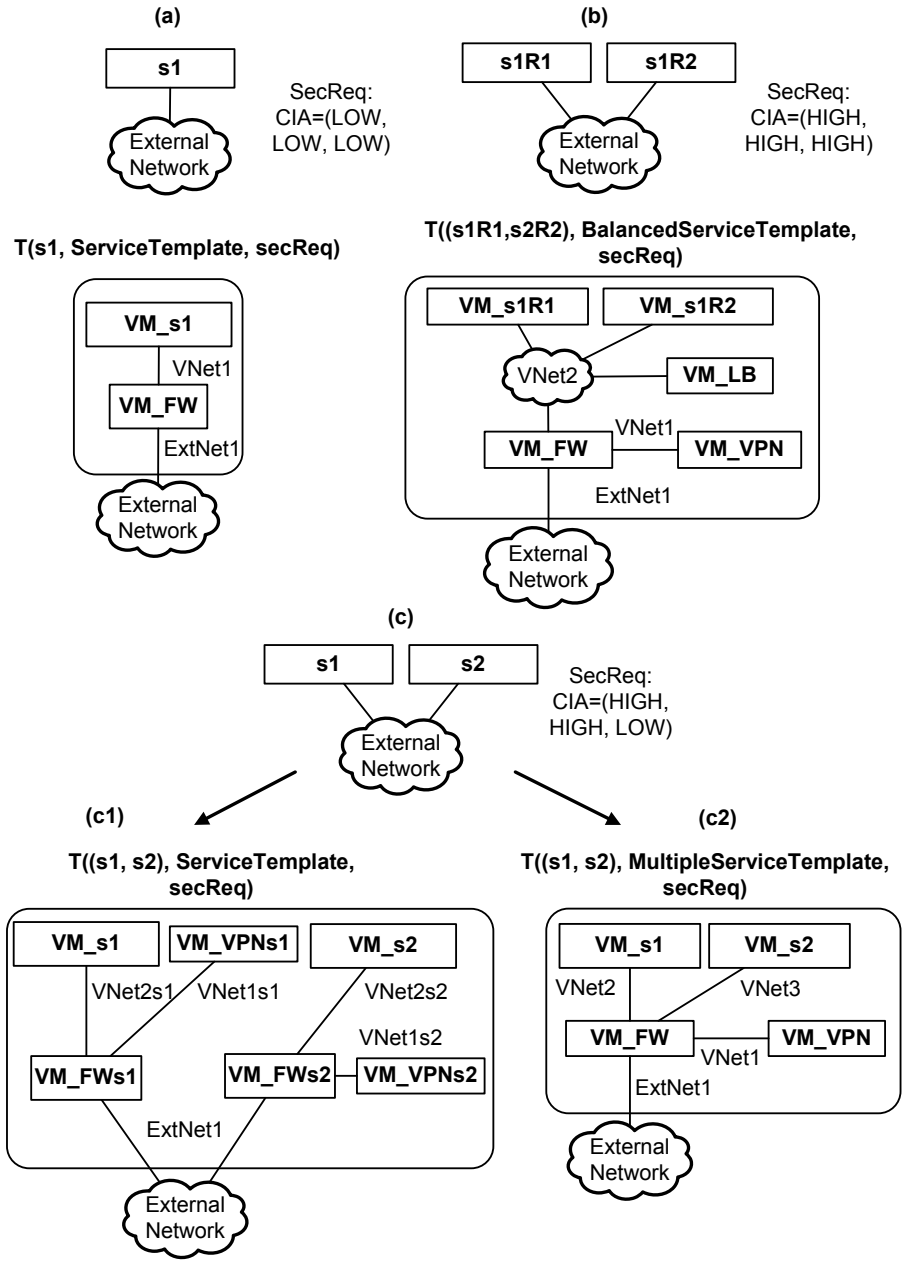



**Figure 3. Security controls transformations**

(*VM_LB*) and the replicas (*VM_s1R1* and *VM_s1R2*); (2) the sub-network *VNet1* that contains the VPN gateway (*VM_VPN*). The virtual firewall (*VM_FW*) policies are: (1) port forwarding for load-balanced services; (2) port forwarding for VPN service (if the gateway is based on TLS VPN technology). In addition, the tool configures the *VM_LB* to balance traffic among replicas and the *VM_VPN* to protect communications. The load-balancer configuration depends on mechanism selected to implement its functionality. The VPN gateway, often implemented as TLS VPN (for flexibility purposes), can be configured as client or server. For example, it is often configured as server on the service side and in client mode on external clients (any-to-one interaction model). In other cases, VPN gateways could be configured as one-to-one, i.e. to tunnel traffic between two different services or any-to-any, i.e. to tunnel

**Listing 8. VPN and balancer network configuration and policies for (b)**

```
<VM id='VM_VPN'>
  <network-interface value='eth0'>
    <addr type='IPv4' value='10.10.10.2' />
    <netmask value='255.255.255.0' />
    <gateway value='10.10.10.1' /> <!-- to VM_FW -->
    <dns value='ask to admin' />
  </network-interface>
  <vpn-network-conf>
    <vpn-pool type='IPv4' lowValue='10.10.10.5'
highValue='10.10.10.10'/>
      <vpn-service-addr type='IPv4' value='10.10.10.2' />
      <vpn-service-interface type='tap' value='tap0'/>
      <vpn-service-protocol type='tcp' port='1194'/>
  </vpn-network-conf>
</VM>
<VM id='VM_LB'>
  <network-interface value='eth0'>
    <addr type='IPv4' value='10.10.20.2' />
    <netmask value='255.255.255.0' />
    <gateway value='10.10.20.1' /> <!-- to VM_FW -->
    <dns value='ask to admin' />
  </network-interface>
  <balancer-conf>
    <balancer id='balancer-VM_LB'>
      <addr type='IPv4' value='10.10.20.2' />
      <proto type='tcp' port='80' />
    </balancer>
    <replicas>
      <replica id='replica-VM_s1R1'>
        <addr type='IPv4' value='10.10.20.3' />
        <proto type='tcp' port='80' />
      </replica>
      <replica id='replica-VM_s1R2'>
        <addr type='IPv4' value='10.10.20.4' />
        <proto type='tcp' port='80' />
      </replica>
    </replicas>
  </balancer-conf>
  <balancer-policies>
    <balancer-policy id='1'>
      <balancer-fronted value='balancer-VM_LB'/>
      <balancer-replica value='replica-VM_s1R1'/>
      <balancer-replica value='replica-VM_s1R2'/>
    </balancer-policy>
  </balancer-policies>
</VM>
<VM id='VM_s1R1'>
  <network-interface value='eth0'>
    <addr type='IPv4' value='10.10.20.3' />
    <netmask value='255.255.255.0' />
    <gateway value='10.10.20.1' />
    <dns value='ask to admin' />
  </network-interface>
</VM>
<VM id='VM_s1R2'>
  <network-interface value='eth0'>
    <addr type='IPv4' value='10.10.20.4' />
    <netmask value='255.255.255.0' />
    <gateway value='10.10.20.1' />
    <dns value='ask to admin' />
  </network-interface>
</VM>
```

traffic among different services. In listings 8, 9 we propose a configuration to balance *VM_s1R1* and *VM_s1R2* replicas. In this case, we adopt the TLS VPN approach and the listing 8 contains useful information for *VM_VPN* configuration. First of all, the definition of the network addresses which belong to the VPN and assigned to clients, in that case, the tool allocates 5 IPs from 10.10.10.5 to 10.10.10.10. In addition, the tool generates, using the specific TLS VPN template, the virtual interface (tap0) and the protocol and port of the VPN service (tcp/1194). To allow external clients accessing the internal services (*VM_s1R1* and *VM_s1R2*) they must join to the VPN. For that purpose, the virtualised firewall is configured to forward ciphered traffic (using TCP protocol) to the virtual host 10.10.10.2 on port 1194. The load balancer configuration is quite simple and contains the description of adopted replicas and a set of policies to balance traffic. In practice for each replica the tool defines an IP address that belongs to the *VNet2* and adds *VM_s1R1* and *VM_s1R2* as members of balancing pool.

**Listing 9. Firewall network configuration and policies for (c)**

```
<VM id='VM_FW'>
  //bridged interface
  <network-interface value='eth0'>
    <addr type='IPv4' value='ask to admin' />
    <netmask value='ask to admin' />
    <gateway value='ask to admin' />
    <dns value='ask to admin' />
  </network-interface>
  //VNet1
  <network-interface value='eth1'>
    <addr type='IPv4' value='10.10.10.1' />
    <netmask value='255.255.255.0' />
    <gateway value='ask to admin' />
    <dns value='ask to admin' />
  </network-interface>
  //VNet2
  <network-interface value='eth2'>
    <addr type='IPv4' value='10.10.20.1' />
    <netmask value='255.255.255.0' />
    <gateway value='ask to admin' />
    <dns value='ask to admin' />
  </network-interface>
  <policies>
    //allow access to VPN
    <filtering-policy id='1'>
      <type='forwarding' />
      <src ip='*' proto='tcp' port='1194' />
      <dst ip='10.10.10.2' proto='tcp' port='1194' />
      <action value='allow' />
    </filtering-policy>
    //allow access to balancer
    <filtering-policy id='2'>
      <type='forwarding' />
      <src ip='10.10.10.*' proto='tcp' port='80' />
      <dst ip='10.10.20.2' proto='tcp' port='80' />
      <action value='allow' />
    </filtering-policy>
  </policies>
</VM>
```

The last example (Listing 9) describes the alternative solutions (c1 and c2) to configure a set that aggregates two or more services. The major difference between the provided solutions is that in (c1) each service is protected by a dedicated firewall and VPN gateway, on the contrary, the (c2) adopts shared firewall and VPN virtual machines. The different approaches have pros and cons. The dedicated fire-

wall and VPN allow protecting the services in more fine grained way. For example, if a VPN channel is compromised, the other services are not involved. On the contrary, if we adopt a shared VPN gateway, in case of attacks, each service could be compromised. However the use of shared resources enhances the entire system performance. In addition, the configuration of dedicated firewall and VPN, often requires assigning an external IP address for each firewall (*VM_FWs1* and *VM_FWs2*) to correctly perform the port forwarding. Consider for example that *VM_s1* and *VM_s2* are web servers hosting *s1* and *s2* applications and each one communicates with external world using tcp protocol on port 80. In this situation we need to assign an external IP address for each firewall to distinguish *s1* and *s2* requests.

# 5 Virtual machines generation and management

Once the VM configurations are generated, it is necessary create and configure the related VMs to implement a solution. This goal requires the creation of the software environment that hosts the services, the configuration of networks, services, policies and the deployment of the VM to a physical host. In addition, to react on fault events, it is also important monitor VM services, manage startup, shutdown and migration of the virtual machines.

## 5.1 Building software environment and specific configurations

Several approaches could be followed to build the software environment, for example it is possible create manually a base system template, building a VM from a common GNU/Linux distribution. Then, the base system template could be modified in automatic way, generating a set of specific templates, adding the required software packages to implement services. For example, the base system could be transformed into a VPN VM template adding IPSec or OpenVPN software packages. Similarly adding the Apache Tomcat package we can implement a web server VM template. The next step requires translating the services, network configuration and policies from a technology specific and device/service independent language (previous described models) to the device/service specific language (e.g. network configuration for a GNU/Linux system, rules for a netfilter firewall). This task can be performed using a set of adapters, one for each device/service category. For example, an adapter for VPN should be able to translate an IPSec configuration into the racoon specific language and a TLS VPN into the OpenVPN language. Similarly, for filtering, the adapter should translate the device/service independent configuration into a set of netfilter or PF (OpenBSD Packet Filter) rules. The adapter output is a device/service specific configuration that must be deployed into the VM.

## 5.2 Deploying and Managing VMs

Finally, the configured VMs should be deployed into the physical machine that will host the services. This task requires to: (1) identify which physical machine are able to host the VMs; (2) define a set of mechanisms to transfer the VMs to physical hosts. Parsing the system description model allow to identify which physical hosts support virtualisation and which are their performance. This information is useful to know how many virtual machines can be deployed on a particular host. The deployment process, or in other words the task that transfers the VM to a physical host, is quite simple but it depends on the virtualisation technology adopted (e.g. Xen, KVM, VMware, etc. ). On the contrary the VM managing tasks (startup, shutdown, migration, network and disk management) are not simple to address. To handle these tasks and reducing the problem complexity a possible solution is to adopt a toolkit, like *libvirt* [15], able to deal with different virtualisation technologies, hiding details. The *libvirt* toolkit offers a set of API to interact with the virtualisation capabilities of physical hosts to deploy and manage VMs.

## 5.3 Our approach

In this section we describe our approach to generate and manage the virtual machines accordingly to the previous defined models.

### 5.3.1 Architecture layers

To perform the tasks described before we introduce in our architecture three different layers: *VM software*, *VM configuration*, *VM management*. The VM software layer contains the activities to build a specific VM template like *VM_FW*, *VM_VPN*, *VM_LB* and VM for specific services, for example to host Apache Tomcat web application. More practically we start from a common GNU/Linux distribution, built manually, to derive automatically a set of predefined templates, adding the required packages, e.g. OpenVPN for *VM_VPN*. To perform this process automatically we adopt a tool (*ubuntu-vm-builder* [27]) able to add new software packages to a virtual machine. The predefined templates could be generated only once, when the tool is run for the first time. On the contrary, a specific template that hosts a particular service must be generated when needed. The build process is performed on a particular physical host, the builder machine that is also used to deploy and manage VMs. The VM configuration layer performs the following tasks: (1) VM internal network configuration; (2) service

configuration; (3) policy configuration; (4) VM configuration. The VM internal network configuration is generated accordingly to the security controls transformation, starting from network configuration and policies model, defined before. The information is translated using a specific adapter that transforms the model to a specific network configuration language. For example, considering the listing 7 and suppose that the VM is implemented using an Ubuntu Linux operating system, each VM network configuration is translated into '/etc/network/interfaces' language. For bridging interfaces, the tool asks to admin the IP address that should be defined accordingly to physical host and external network. The service configuration, for example an Apache Tomcat web application, is translated using a specific template and a service dependant adapter. The policies, similarly to network configuration are translated from network configuration and policies model to a specific security control language, e.g. netfilter, using a specific adapter. Finally it is necessary to build the configuration of the virtual machine as XML model. To perform this step the tool takes as inputs (a) the VM template generated before (e.g. *VM_FW*); (b) network configuration and policies model. Considering the listing 10, the template is used to define every properties of the XML model except for the network interface tags, that are generated using network configuration and policy model. In addition, this model is used to create the virtual network properties, as shown for *VNet1* in listing 11. The specific configurations, except for virtual network properties, derived from previous tasks, are deployed into the VM. More practically, the VM disk is mounted on local file system of the builder machine, and specific configuration are copied accordingly. On the contrary the virtual network properties are used in the next step to configure the virtualisation environment.

The VM management layer is able to setup the virtualisation environment, deploy, migrate, start and stop a virtual machine. In order to create and deploy a VM on the physical host, the first activity is setup the virtualisation environment. In practice, our tool, using the *libvirt-java* API, creates the network environment on the remote host, accordingly to the network properties described in listing 11. In the next step, the tool takes as input the VM configuration model of 10 and using the API creates the new domain on remote physical host. Finally, the last step copies the VM disk on the remote host, and the VM is ready to start.

**Listing 10.** *libvirt* **VM configuration model**

```
<domain type='kvm'>
  <name>vm-fw1</name>
  <uuid>0cc4736f-2568-241d-c610-2e7ba90002f5</uuid>
  <memory>262144</memory>
  <currentMemory>262144</currentMemory>
  <vcpu>1</vcpu>
  <os>
    <type arch='i686' machine='pc-0.11'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <clock offset='utc'/>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/bin/kvm</emulator>
    <disk type='file' device='disk'>
      <source file='/home/vm-deployment/vm-fw1-disk.img'
          />
      <target dev='hda' bus='ide'/>
    </disk>
    <disk type='file' device='cdrom'>
      <target dev='hdc' bus='ide'/>
      <readonly/>
    </disk>
    <interface type='network'>
      <mac address='54:52:00:60:ef:e5'/>
      <source network='VNet1'/>
    </interface>
    <interface type='bridge'>
      <mac address='54:52:00:04:3b:c0'/>
      <source bridge='br0'/>
    </interface>
    <serial type='pty'>
      <target port='0'/>
    </serial>
    <console type='pty'>
      <target port='0'/>
    </console>
    <input type='mouse' bus='ps2'/>
    <graphics type='vnc' port='-1' autoport='yes' keymap=
        'it'/>
    <video>
      <model type='cirrus' vram='9216' heads='1'/>
    </video>
  </devices>
</domain>
```

**Listing 11.** *libvirt* **virtual network properties for** *VNet1*

```
<network>
  <name>VNet1</name>
  <uuid>3e3fce45-4f53-4fa7-bb32-11f34168b82b</uuid>
  <bridge name='virbr1' stp='on' forwardDelay='0' />
  <ip address="10.10.10.254" netmask="255.255.255.0">
    <dhcp>
      <range start="10.10.10.100" end="10.10.10.200" />
      <host mac="54:52:00:60:ef:e5" name="VM_FW" ip="
          10.10.10.1" />
    </dhcp>
  </ip>
</network>
```

## 6 Conclusion

We have analysed how the best practices for service and network dependability change when exploiting modern virtualisation technologies. Based on the results of this study, we have updated the process and tools we had developed for semi-automatic dependability planning of information systems.

In particular, this paper has detailed the algorithms that our tools exploit to automatically compute allocation and reaction plans for virtualised information systems. Each algorithm is defined in terms of the relevant modelling ontology and the pool of transformation rules working on this ontology. The actual implementation exploits XML representations and transformation languages. We have also developed prototype integration of our tools with popular virtual machine management software, which is an essential step towards automatic deployment of the generated plans.

During the selection of the relevant configuration strategies, we have focused on the point of view of the virtual data center customer: our approach can, and should be, extended taking in further consideration the point of view of the hosting provider, which may partially conflict with the customer's one.

## Acknowledgment

This work was developed in the framework of IST-026600 DESEREC, "Dependability and Security by Enhanced Reconfigurability", an Integrated Project partially funded by the E.C. under the Framework Program 6, IST priority.

## References

[1] Network Working Group, Address Allocation for Private Internets. IETF RFC 1918, February 1996.

[2] W3C Consortium, Web Services Choreography Description Language. http://www.w3.org/TR/ws-cdl-10/, 2005.

[3] POSITIF Consortium, The POSITIF System Description Language (P-SDL). http://www.positif.org/, 2007.

[4] National Vulnerability Database. http://nvd.nist.gov/, 2008.

[5] Common Vulnerability Scoring System. http://www.first.org/cvss/, 2009.

[6] M. D. Aime, P. C. Pomi, and M. Vallini. Policy-driven system configuration for dependability. *Emerging Security Information, Systems, and Technologies, The International Conference on*, 0:420–425, 2008.

[7] M. D. Aime, P. C. Pomi, and M. Vallini. Planning dependability of virtualised networks. *Dependability, International Conference on*, 0:46–51, 2009.

[8] Amazon. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/.

[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177, 2003.

[10] N. Damianou, N. Dulay, E. C. Lupu, and M. Sloman. Ponder: a language for specifying security and management policies for distributed systems. *Imperial College Research Report DoC 2000/1*, 2000.

[11] T. Eilam, M. Kalantar, A. Konstantinou, G. Pacifici, J. Pershing, and A. Agrawal. Managing the configuration complexity of distributed applications in internet data centers. *Communications Magazine, IEEE*, 44(3):166–177, March 2006.

[12] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone. Requirements engineering meets trust management - model, methodology, and reasoning. In *In Proc. of iTrust '04, LNCS 2995*, pages 176–190. Springer-Verlag, 2004.

[13] M. Israel, J. Borgel, and A. Cotton. Heuristics to perform molecular decomposition of large mission-critical information systems. In *SECURWARE '08: Proceedings of the 2008 Second International Conference on Emerging Security Information, Systems and Technologies*, pages 338–343, 2008.

[14] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4:382–401, 1982.

[15] libvirt. http://libvirt.org/.

[16] A. Menon, A. L. Cox, and W. Zwaenepoel. Optimizing Network Virtualization in Xen. Proceedings of the USENIX Annual Technical Conference, 2006.

[17] Microsoft Corporation. Azure Service Platform. http://www.microsoft.com/azure/.

[18] D. M. Nicol, W. H. Sanders, and K. S. Trivedi. Model-based evaluation: From dependability to security. *IEEE Transactions on Dependable and Secure Computing*, 1(1):48–65, 2004.

[19] J. Oberheide, E. Cooke, and F. Jahanian. Empirical exploitation of live virtual machine migration. In *In Proceedings of the BlackHat DC convention*, 2008.

[20] E. Rescorla, A. Cain, and B. Korver. SSLACC: A Clustered SSL Accelerator. In *Proceedings of the 11th USENIX Security Symposium*, pages 229–246, Berkeley, CA, USA, 2002. USENIX Association.

[21] J. Salowey, H. Zhou, P. Eronen, and H. Tschofenig. Transport Layer Security (TLS) Session Resumption without Server-Side State. IETF RFC 5077, January 2008.

[22] F. Satoh, Y. Nakamura, N. K. Mukhi, M. Tatsubori, and K. Ono. Methodology and tools for end-to-end soa security configurations. In *SERVICES '08: Proceedings of the 2008 IEEE Congress on Services - Part I*, pages 307–314, Washington, DC, USA, 2008. IEEE Computer Society.

[23] J. Strassner. *Policy-Based Network Management: Solutions for the Next Generation (The Morgan Kaufmann Series in Networking)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[24] The DMTF Technical Committee. The Common Information Model (CIM). http://www.dmtf.org/standards/cim, 2008.

[25] Trusted Computing Group. https://www.trustedcomputinggroup.org, 2009.

[26] Ubuntu. Ubuntu Enterprise Cloud. http://www.ubuntu.com/cloud/private.

[27] Ubuntu. ubuntu-vm-builder. https://help.ubuntu.com/8.04/serverguide/C/ubuntu-vm-builder.html.

[28] VMware. http://www.vmware.com.

# Model Transformations Given Policy Modifications in Autonomic Management

Raphael M. Bahati and Michael A. Bauer
Department of Computer Science
The University of Western Ontario, London, ON N6A 5B7, CANADA
Email: {rbahati;bauer}@csd.uwo.ca

***Abstract*—This paper presents an approach for adapting a model learned from the use of an active set of policies to run-time policy modifications. The work leverages our most resent efforts utilizing Reinforcement Learning methodologies to facilitate dynamic use of policies within autonomic computing. The use of policies in this context offers significant benefits to autonomic systems in that it allows systems to determine how to effectively meet the desired, and at times seemingly conflicting, objectives under dynamically changing conditions. Contrary to other approaches that make use of some form of learning to model performance management, our approach enables the model learnt from the use of an active set of policies to be reused once those policies change. Since the learning mechanisms are modelled from the structure of the policies, policy modifications can be mapped onto the learnt model. Our analysis of the policy modifications suggest that most of the learned model could be reused, potentially accelerating the learning process. In this paper, we provide formal definitions on the different kinds of policy modifications that might occur as well as elaborate, with detailed examples, on how such modifications could impact the currently learned model.**

***Index Terms*—Model Transformation, Model Adaptation, Reinforcement Learning, Autonomic Management, Policy-based Management.**

## I. Introduction

Policy-based management has been proposed as a way in which the behavior of systems and applications can be adjusted at run-time rather than through re-engineering [1]. The concept of policy-based management has been adopted by standard bodies such as the (Internet Engineering Task Force (IETF) [2], the Distributed Management Task Force (DMTF) [3], and the Object Management Group (OMG) [4]. Policy-based management has also attracted significant interest from a wide variety of efforts, ranging from academia [5] to industry [6], which have resulted in a great deal of diversity in terms of what policies are and how they should be used.

It is often common that policies are used to express required or desired behavior of systems and applications. In the context of autonomic computing, for example, policies can be input to or embedded within the autonomic management elements of the system to provide the kinds of directives which an autonomic manager could make use of in order to meet operational requirements. Thus, through the modifications of the policies driving autonomic management, the behavior of systems could be dynamically adjusted. Such directives could come from the users of such systems or other autonomic management elements within the computing environment.

While the use of policies has been the subject of considerable research focus, very little work has been done on evaluating the effectiveness of the policies when in use to determine whether the performance objectives are, in fact, being met. More often than not, policy use within autonomic computing has typically focused on the specification and use "as is" within systems. It is often assumed, for example, that once the policies are specified, the system would behave as expected. Consequently, many of these approaches fail to recognize the stochastic, dynamic, and heterogeneous nature of today's systems which necessitates the need for autonomic systems to adapt not only to how they use policies, but also to run-time policy modifications. This is essential in order for systems to cope with not only changes to the configuration of the managed environment, but also changes to the perceived quality of services, something that is often time-, user-dependent, and application-specific.

To this end, we have proposed an adaptive policy-driven autonomic management framework highlighting two key contributions in the use of policies within autonomic computing: (1) The ability to evaluate, through model-learning, the effectiveness of the enabled policies based on past experience from their use which, in turn, provide guidance to the autonomic system on how to effectively use the policies [7]. (2) The ability to adapt the model learned from the use of policies to run-time policy modifications, essentially allowing most of the learned model to be reused in the new environment [8]. The focus of this work is on the second of these approaches. It expands on the strategies we initially proposed in [8] by formally describing the types of changes to a set of policies that could occur and how we map such changes to model transformations. This is done within the context of the adaptive policy-driven autonomic management framework in [9].

The rest of the paper is organized as follows. We begin with an overview of how we model learning in Section II, summarizing some of the key definitions we make use of in this paper. We then present our approach to model adaptation in Section III. The section begins with formal definitions of the types of changes that might occur to a set of policies, and then describe how such modifications are mapped to specific cases of model transformations. We summarize the results of the experiments in Section IV, describe some related work in Section V, and conclude in Section VI.

## II. Learning by Reinforcement

This section summarizes key definitions outlining our approach to modelling Reinforcement Learning [10] in the context of policy-driven autonomic management. A more detailed account can be found in [7].

*Definition 1:* An expectation policy is defined by the tuple $p_i = \langle C, A \rangle$ where:

- $C$ is conjunctive conditions associated with policy $p_i$ with each condition, $c_j \in C$, defined by the tuple $c_j = \langle$ `ID, metricName, operator,` $\Gamma\rangle$, where `ID` is a unique identification for the condition; `metricName` is the name of the metric associated with the condition; `operator` is the relational operator associated with the condition; and $\Gamma$ is the threshold of the condition.
- $A$ is a set of actions associated with policy $p_i$ with each action, $a_j \in A$, defined by the tuple $a_j = \langle$ `ID, function, parameters,` $\tau\rangle$, where `ID` is a unique identification for the action; `function` is the name of the function that should be executed; `parameters` is a set of function parameters; and $\tau$ is a set of tests associated with the action.

A policy-driven autonomic management system is likely to consist of multiple expectation policies, a subset of which may be active (or enabled) at any given time, which brings us to our next definition.

*Definition 2:* Suppose that $P^A$ denotes a set of all expectation policies such that $p_i \in P^A$ where $p_i = \langle C, A \rangle$. Let $P$ be a subset of expectation policies an autonomic manager uses to make management decisions; i.e., $P \subseteq P^A$. A policy system corresponding to $P$ is defined by the tuple $PS = \langle P, W_C \rangle$ where:

- $W_C = \langle c_i, \omega_i \rangle$ associates each policy condition, $c_i$, with a metric weight, $\omega_i$, such that, for all $c_i \in p_m$ and $c_j \in p_n$, $\omega_i = \omega_j$ if $c_i$.`metricName` $= c_j$.`metricName`.

Essentially, a *policy system* is the set of active policies where each condition occurring in some policy has an associated weight. The metrics weights, which are specified manually in our current implementation, provide a way of distinguishing policy conditions based on the significance of violating a particular metric. In essence, $W_C$ provides a way of biasing how the autonomic system responds to quality of service violations. These definitions provide the fundamental structure that is used to build our reinforcement learning model.

### A. System States

To model system's dynamics from the use of an active set of policies, we make use of a mapping between the enabled expectation policies and the managed system's states whose structure is derived from the metrics associated with the enabled policy conditions.

*Definition 3:* A policy system $PS = \langle P, W_C \rangle$ derives a set of system metrics, $m_i \in M$, such that, for each $p_j = \langle C_j, A_j \rangle$ where $p_j \in P$, $M = \bigcup_{c_i \in C_j} \{c_i.\texttt{metricName}\}$.

The set $M$ is the set of all metrics occurring in any of the active policies. For each metric in this set, there are a finite number of threshold values to which the metric is compared; there can be ordered to form regions:

*Definition 4:* A policy system $PS = \langle P, W_C \rangle$ with metrics set $M$ derives a set of metric regions, $r_{m_i} \in M_R^P$, for each metric $m_i \in M$, whose structure is defined by the tuple $r_{m_i} = \langle \alpha_{m_i}, \sigma_{m_i} \rangle$, where:

- $\alpha_{m_i} = \langle$ `ID, metricName,` $\omega\rangle$ corresponds to a unique metric from among the metrics of the conditions of the policies in $P$; such that, `metricName` is the name of the metric and $\omega$ is the weight of the condition (see Definition 2) associated with metric $m_i$.
- $\sigma_{m_i} = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_k\}$ is a set of thresholds from the conditions associated with metric $m_i$ such that, $\Gamma_i < \Gamma_j$ if $i < j$. As such, $\sigma_{m_i}$ derives a set of metric regions which map the observed metric measurement onto appropriate localities (i.e., intervals) as defined by the thresholds of the policy conditions associated with metric $m_i$, such that $R_{m_i} = \{R_{m_i}^1, R_{m_i}^2, \ldots, R_{m_i}^{k+1}\}$, where $R_{m_i}^1 = (-\infty, \Gamma_1)$, $R_{m_i}^2 = (\Gamma_1, \Gamma_2)$, etc., and $R_{m_i}^{k+1} = (\Gamma_k, \infty)$.

To be precise, the actual boundaries of region $R_{m_i}^j$ are determined by the operators of the policy conditions associated with a given metric. For example, a system with two conditions $c_1 = \langle$ `ID,` $m_i$`, >,` $\Gamma_1\rangle$ and $c_2 = \langle$ `ID,` $m_i$`,` $\geq$`,` $\Gamma_2\rangle$ such that $\sigma_{m_i} = \langle \Gamma_1, \Gamma_2 \rangle$ would yield three regions in our approach, namely: $R_{m_i}^1 = [-\infty, \Gamma_1)$, $R_{m_i}^2 = [\Gamma_1, \Gamma_2)$, and $R_{m_i}^3 = [\Gamma_2, \infty]$. Thus, a metric measurement of, say, $\Gamma_1$ would fall into region $R_{m_i}^2$. This brings us to our next definition.

*Definition 5:* Given a set of metric-regions $r_{m_i} \in M_R^P$ for each metric $m_i \in M$ such that $r_{m_i} = \langle \alpha_{m_i}, \sigma_{m_i} \rangle$, where $\sigma_{m_i}$ derives a set of metric regions $R_{m_i}^j \in R_{m_i}$; we define a weighting function, $f(R_{m_i}^j) \rightarrow \mathbb{R}$, which assigns a numeric value to the j-th region in $R_{m_i}$ such that, $f(R_{m_i}^k) > f(R_{m_i}^l)$ if $k < l$.

An example of such a mapping, which we make use of in our current implementation, is defined by Equation 1:

$$f(R_{m_i}^j) \;=\; 100 - (\frac{100}{n-1})(j-1) \qquad (1)$$

where $n$ is the total number of regions in $R_{m_i}$. This function assigns a numeric value between 100 and 0 for each metric's region in $R_{m_i}$, starting from 100 for the most desirable region and decrementing at equal intervals towards the opposite end of the spectrum, whose region is assigned a value of 0. This approach guarantees that the highest value is assigned to the most desirable region (i.e., the region corresponding to the highest quality of service), assuming, of course, that the assumptions about the conditions of the expectation policies hold (see Definition 1). That is, smaller metric values are "more desirable" though, in general, this is not a necessary requirement of the weighting function. The idea is that, regions of greater "desirability", i.e., preferred quality of service, are assigned higher values. The key role of these regions is that they partition the space of values that a metric can take on with respect to the thresholds in conditions involving that metric. We use these to define a state within our model.

*Definition 6:* A policy system $PS = \langle P, W_C \rangle$ with metrics $M$ and metrics-regions $M_R^P$ derives a set of system states

$S$ such that each state $s_i \in S$ is defined by the tuple $s_i = \langle \mu, M(s_i), A(s_i) \rangle$, and where:

- $\mu$ is a `type` which classifies a state as either "violation" or "non-violation" depending, respectively, on whether or not there are any policy violations as a result of visiting state $s_i$.
- $P(s_i)$ is a set of expectation policies that are violated when the system is in state $s_i$ such that, $P(s_i) \in P$. As noted previously, a policy is said to be violated if all its conditions evaluate to true when matched against violation notifications received within a management cycle.
- $A(s_i)$ is a set of actions advocated by the expectation policies in $P(s_i)$ plus the $\gamma$-action; i.e., $a_0$ which corresponds to "do-nothing".
- $M(s_i)$ is a set of state metrics for each metric $m_j \in M$, $r_{m_j} \in M_R^P$, $r_{m_j} = \langle \alpha_{m_j}, \sigma_{m_j} \rangle$, such that each state metric $s_i.m_j \in M(s_i)$ is defined as follows:

*Definition 7:* A state metric $s_i.m_j \in M(s_i)$ given $\alpha_{m_j} = \langle \texttt{ID}, \texttt{metricName}, \omega \rangle$ and $\sigma_{m_j} = \langle \Gamma_1, \Gamma_2, \ldots, \Gamma_k \rangle$ is defined by the tuple $s_i.m_j = \langle \texttt{ID}, \omega, \texttt{value}, R^l_{m_j} \rangle$ where:

- `ID` is an integer value that uniquely identifies each metric $m_i \in M$.
- $\omega$ is the weight associated with metric $m_i$.
- `value` is the observed metric measurement, or average value when state $s$ is visited multiple times.
- $R^l_{m_j}$ is the region corresponding to a region in $\sigma_{m_j}$ in which the average metric measurement (i.e., `value`) falls; i.e., if $R^l_{m_j} = (\Gamma_1, \Gamma_2)$, then $\Gamma_1 < s_i.m_j.\texttt{value} < \Gamma_2$. Thus, for each such region, $f(R^l_{m_j})$ associates a value as described by Equation 1.

Using this approach, each state can be uniquely identified by the region occupied by each state metric based on the conditions of the expectation policies and the `value` associated with each metric. That is, for a set of policies involving $n$ metrics, each state would have $n$ metrics $\{m_1, m_2, ..., m_n\}$ each assigned a region whose intervals are derived from the thresholds of the conditions associated with the metric. We elaborate further on this in Section II-C.

### B. System Transitions

Transitions are essentially determined by the actions taken by the management system and labelled by a value determined by the learning algorithm.

*Definition 8:* A state transition $t_i(s_p, a_p, s_c)$ is a directed edge corresponding to a transition originating from state $s_p$ and ending on state $s_c$ as a result of taking action $a_p$ while in state $s_p$, and is labelled by $\langle \lambda, Q_{t_i}(s_p, a_p) \rangle$, where:

- $\lambda$ is the frequency (i.e., the number of times) through which the transition occurs.
- $Q_{t_i}(s_p, a_p)$ is the action-value estimate associated with taking action $a_p$ in state $s_p$. In our current implementation, $Q_{t_i}(s_p, a_p)$ is computed using a one-step Q-Learning [10], [11] algorithm.

It is worth pointing out that, non-deterministic transitions are also possible. That is, taking the same action in the same state during different time-steps may result in transitions to different states. A change in the system's state may also be due to external factors other than the impact of the actions taken by the autonomic manager. In a dynamic Web server environment, for example, a transition may be a result of a request to a page with a database-intensive query, which could potentially cause a state transition. In this work, such transitions are referred to as $\gamma$-transitions; the action responsible for $\gamma$-transitions is denoted by $a_0$ (i.e., $\gamma$-`action`) as illustrated in Table IV.

### C. State-Transition Model

A state-transition model is then defined for a set of active expectation policies:

*Definition 9:* A state-transition model derived from the policy system $PS = \langle P, W_C \rangle$ is defined by the graph $G^P = \langle S, T \rangle$ where:

- $S$ is a set of system states (see Section II-A) derived from the metrics of the conditions of the enabled expectation policies and where each state, $s_i \in S$, corresponds to a vertex on the graph.
- $T$ is a set of transitions (see Section II-B) where each transition, $t_i \in T$, corresponds to a directed edge on the graph. A transition is determined when the autonomic manager takes an action as a result of being in one state, which may, or may not, result in a transition to another state.

For illustration purposes, suppose that the policies of Figure 1 are the only enabled expectation policies. From the conditions of Table I, six metrics are then formed as illustrated in Table II. Metric $m_1$, for instance, would correspond to the "`CPU:utilization`" policy conditions. This metric is mapped onto three regions based on the thresholds of the conditions the metric is associated with; i.e., $\sigma_{m_1} = \{15.0, 85.0\}$. This means that, "`CPU:utilization`" could fall into three unique localities; its value could be less than 15.0% (i.e., region $R^1_{m_1}$) between 85.0% and 15.0% inclusive (i.e., region $R^2_{m_1}$), or greater than 85.0% (i.e., region $R^3_{m_1}$). Each of the regions is also assigned a numeric value between 0 and 100 as described in Definition 5. It is worth pointing out that, for the system with only the policies in Figure 1 enables, 144 (i.e., $3^2 \times 2^4$) states are possible. This is because there are two state metrics (i.e., $m_1$ and $m_5$) with three possible regions each (i.e., $3^2$ possible permutations) and four state metrics (i.e., $m_2$, $m_3$, $m_4$, and $m_6$) with two possible regions each (i.e., $2^4$ possible permutations).

| $c_i$ | $W_C(c_i)$ | Policy Condition |
|---|---|---|
| 1 | 1/8 | `CPU:utilization > 85.0` |
| 2 | 1/8 | `CPU:utilizationTREND > 0.0` |
| 3 | 1/8 | `CPU:utilization < 15.0` |
| 4 | 1/8 | `MEMORY:utilization > 40.0` |
| 5 | 1/8 | `MEMORY:utilizationTREND > 0.0` |
| 6 | 1/8 | `APACHE:responseTime > 2000.0` |
| 7 | 1/8 | `APACHE:responseTimeTREND > 0.0` |
| 8 | 1/8 | `APACHE:responseTime < 250.0` |

TABLE I
POLICY CONDITIONS SET FROM THE EXPECTATION POLICIES IN FIGURE 1.

Once the metrics structure has been constructed, the next step in the creation of actual states involves mapping the Mon-

```
[1] expectation policy{CPUViolation(PDP, PEP)}
 if(CPU:utilization > 85.0) &
   (CPU:utilizationTREND > 0.0)
 then{AdjustMaxClients(-25)
        test{newMaxClients > 49} |
      AdjustMaxKeepAliveRequests(-30)
        test{newMaxKeepAliveRequests > 1} |
      AdjustMaxBandwidth(-128)
        test{newMaxBandwidth > 255}}
[2] expectation policy{RESPViolation(PDP, PEP)}
 if(APACHE:responseTime > 2000.0) &
   (APACHE:responseTimeTREND > 0.0)
 then{AdjustMaxClients(+25)
        test{newMaxClients < 151} &
        test{IdleWorkers  < 5} |
      AdjustMaxKeepAliveRequests(-30)
        test{newMaxKeepAliveRequests > 1} |
      AdjustMaxBandwidth(-128)
        test{newMaxBandwidth > 255}}
[3] expectation policy{CPUandRESPViol.(PDP, PEP)}
 if(CPU:utilization > 85.0) &
   (CPU:utilizationTREND > 0.0) &
   (APACHE:responseTime > 2000.0)
 then{AdjustMaxKeepAliveRequests(-30)
        test{newMaxKeepAliveRequests > 1} |
      AdjustMaxBandwidth(-128)
        test{newMaxBandwidth > 255}}
[4] expectation policy{MEMORYViolation(PDP, PEP)}
 if(MEMORY:utilization > 40.0) &
   (MEMORY:utilizationTREND > 0.0)
 then{AdjustMaxClients(-25)
        test{newMaxClients > 49}}
[5] expectation policy{SERVERnormal(PDP, PEP)}
 if(CPU:utilization < 15.0) &
   (APACHE:responseTime < 250.0)
 then{AdjustMaxClients(-25)
        test{newMaxClients > 49} &
        test{IdleWorkers > 25} |
      AdjustMaxKeepAliveRequests(+30)
        test{newMaxKeepAliveRequests < 95}}
```

Fig. 1. Sample policy system where $P = \{p_1, p_2, p_3, p_4, p_5\}$.

itor events collected during a single management interval onto appropriate state metrics and regions. Suppose, for example, that during a single management interval, the management system receives the events in Figure 2. By mapping the events using the metrics structure in Table II, a new state is then created as illustrated in Table III; lets call it state $s_*$. For a management system with only the policies of Figure 1 enabled, being in state $s_*$ would mean the violation of policy $p_1$; i.e.. $P(s_*) = \{p_1\}$. Thus, in addition to the action "$a_0$ : $\gamma$-`action`" which corresponds to "doing nothing", $A(s_*)$ would consist of a set of unique actions from the actions of the policies in $P(s_*)$ as illustrated in Table IV.

```
CPU:utilization = 90.0
CPU:utilizationTREND = 1.0
MEMORY:utilization = 32.0
MEMORY:utilizationTREND = -2.0
APACHE:responseTime = 1500.0
APACHE:responseTimeTREND = -1.0
```

Fig. 2. A set of Monitor events collected during a single management interval.

In the sections that follow, we describe our approach for adapting the state-transition model to run-time policy modifications (denoted by $\triangle[P]$) which we formally define in Section III-A.

## III. Adapting to Policy Changes

Figure 3 describes our approach to model-adaptation. The left-part of the diagram (i.e., prior to $\triangle[P]$) depicts the Reinforcement Learning mechanisms described in Section II. whereby the agent learns the model of the environment based on past experience in the use of an active set of policies, $P$. In this approach, the conditions of the policies define system states while the actions of the policies define possible causes of transitions between states. That is, at each time step, the agent takes action $a_i \in A(s)$ where $A(s)$ is a set of actions advocated by the policies that are violated when the system is in state $s$. This, in turn, causes a transition as depicted in the diagram. Thus, the model is continuously updated based on the experience garnered from the agent's interaction with its environment; i.e., based on the states and transitions encountered. The right-part of the diagram (i.e., after $\triangle[P]$ occurs) depicts the mapping (using the transformation function $\Psi$) from the current system's model (i.e., $G_n^P = \langle S, T \rangle$) to a new model (i.e., $G_{n+1}^{P'} = \langle S', T' \rangle$), as a result of run-time modification to the policies in $P$.

### A. Types of Modifications

By run-time policy modifications, we mean any type of modification resulting in changes to the characteristics of the policies driving autonomic management. Since the structure specific to the model of the environment is derived from the characteristics of the enabled policies (see Section II-A), any run-time changes to these characteristics could have possible ramifications on the learning process. This would likely depend on what kind of modification is done on the policies.

*1) Adding/Removing a Policy:* A policy modification could involve adding a policy onto the policies in $P$; i.e.,

**Given:**

I) $PS = \langle P, W_C \rangle$

II) $p_* = \langle C, A \rangle$ where $p_* \notin P$ :

- $C = \{c_1, c_2, \ldots, c_x\}$ : $c_x = \langle$`ID, metricName, operator,` $\Gamma \rangle$
- $A = \{a_1, a_2, \ldots, a_y\}$ : $a_y = \langle$`ID, function, parameters,` $\tau \rangle$

**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P \cup p_*$

A policy modification could also involve removing an existing policy from $P$; i.e.,

**Given:**

I) $PS = \langle P, W_C \rangle$

II) $p_* = \langle C, A \rangle$ where $p_* \in P$ :

- $C = \{c_1, c_2, \ldots, c_x\}$ : $c_x = \langle$`ID, metricName, operator,` $\Gamma \rangle$
- $A = \{a_1, a_2, \ldots, a_y\}$ : $a_y = \langle$`ID, function, parameters,` $\tau \rangle$

| $m_i$ | $c_k$ | Policy Condition | $R_{m_i}$ | $R^j_{m_i}$ | $f(R^j_{m_i})$ |
|---|---|---|---|---|---|
| $m_1$ | 3 | `CPU:utilization < 15.0` | $m_1$.value $< 15.0$ | $R^1_{m_1}$ | 100 |
| | | | $15.0 \leq m_1$.value $\leq 85.0$ | $R^2_{m_1}$ | 50 |
| | 1 | `CPU:utilization > 85.0` | $m_1$.value $> 85.0$ | $R^3_{m_1}$ | 0 |
| $m_2$ | | | $m_2$.value $\leq 0.0$ | $R^1_{m_2}$ | 100 |
| | 2 | `CPU:utilizationTREND > 0.0` | $m_2$.value $> 0.0$ | $R^2_{m_2}$ | 0 |
| $m_3$ | | | $m_3$.value $\leq 40.0$ | $R^1_{m_3}$ | 100 |
| | 4 | `MEMORY:utilization > 40.0` | $m_3$.value $> 40.0$ | $R^2_{m_3}$ | 0 |
| $m_4$ | | | $m_4$.value $\leq 0.0$ | $R^1_{m_4}$ | 100 |
| | 5 | `MEMORY:utilizationTREND > 0.0` | $m_4$.value $> 0.0$ | $R^2_{m_4}$ | 0 |
| $m_5$ | 8 | `APACHE:responseTime < 250.0` | $m_5$.value $< 250.0$ | $R^1_{m_5}$ | 100 |
| | | | $250.0 \leq m_5$.value $\leq 2000.0$ | $R^2_{m_5}$ | 50 |
| | 6 | `APACHE:responseTime > 2000.0` | $m_5$.value $> 2000.0$ | $R^3_{m_5}$ | 0 |
| $m_6$ | | | $m_6$.value $\leq 0.0$ | $R^1_{m_6}$ | 100 |
| | 7 | `APACHE:responseTimeTREND > 0.0` | $m_6$.value $> 0.0$ | $R^2_{m_6}$ | 0 |

TABLE II
METRICS STRUCTURE DERIVED FROM THE EXPECTATION POLICIES OF FIGURE 1.

| $m_i$ | Metric Value | $R_{m_i}$ | $R^j_{m_i}$ | $f(R^j_{m_i})$ |
|---|---|---|---|---|
| $m_1$ | `CPU:utilization = 90.0` | $m_1$.value $> 85.0$ | $R^3_{m_1}$ | 0 |
| $m_2$ | `CPU:utilizationTREND = 1.0` | $m_2$.value $> 0.0$ | $R^2_{m_2}$ | 0 |
| $m_3$ | `MEMORY:utilization = 32.0` | $m_3$.value $\leq 40.0$ | $R^1_{m_3}$ | 100 |
| $m_4$ | `MEMORY:utilizationTREND = -2.0` | $m_4$.value $\leq 0.0$ | $R^1_{m_4}$ | 100 |
| $m_5$ | `APACHE:responseTime = 1500.0` | $250.0 \leq m_5$.value $\leq 2000.0$ | $R^2_{m_5}$ | 50 |
| $m_6$ | `APACHE:responseTimeTREND = -1.0` | $m_6$.value $\leq 0.0$ | $R^1_{m_6}$ | 100 |

TABLE III
$s_*$ - A STATE DERIVED FROM THE MONITOR EVENTS IN FIGURE 2 AND THE METRICS STRUCTURE OF TABLE II.

| $a_i$ | $Q(s_*, a_i)$ | Policy Actions |
|---|---|---|
| $a_0$ | | $\gamma$-action |
| $a_2$ | | `AdjustMaxClients(-25)` **test**{`newMaxClients > 49`} |
| $a_4$ | | `AdjustMaxKeepAliveRequests(-30)` **test**{`newMaxKeepAliveRequests > 1`} |
| $a_6$ | | `AdjustMaxBandwidth(-128)` **test**{`newMaxBandwidth > 255`} |

TABLE IV
$A(s_*)$ - A SET OF UNIQUE ACTIONS FROM THE ACTIONS OF THE POLICIES THAT ARE VIOLATED WHEN THE SYSTEM IS IN STATE $s_*$.

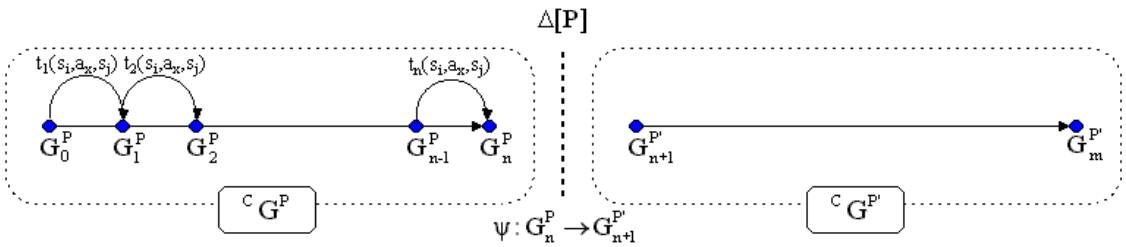



Fig. 3. Adapting to run-time policy modifications.

**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P | p_*$

Since the impact of such modifications on the state-transition model would depend on what conditions and/or actions are affected as a result, adding/removing an entire policy can be modelled in the context of adding/removing individual conditions and actions within policies.

On the one hand, adding policy $p_* = \langle C, A \rangle$ into the policies in $P$ can be modelled by the following sequence of policy modifications: (i) Adding the policy without conditions or actions; i.e., $p_* = \langle \emptyset, \emptyset \rangle$ (see below). (ii) Adding one condition at a time from the conditions in $C$ (see Section III-A2). (iii) Adding one action at a time from the actions in $A$ (see Section III-A5).

Policy Modification - $\triangle_x$[P]: *Adding a policy without conditions or actions.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M^P_R = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle \emptyset, \emptyset \rangle$ such that $p_* \notin P$

**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P \cup p_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

On the other hand, removing policy $p_* = \langle C, A \rangle$ from the policies in $P$ can be modelled by the following sequence of policy modifications: (i) Removing one action at a time from the actions in $A$ (see Section III-A6). (ii) Removing one condition at a time from the conditions in $C$ (see Section III-A3). (iii) Removing the policy without conditions or actions; i.e., $p_* = \langle \emptyset, \emptyset \rangle$ (see below).

Policy Modification - $\triangle_y$[P]: *Removing a policy without conditions or actions.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle \emptyset, \emptyset \rangle$ such that $p_* \in P$

**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P | p_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

*2) Adding a Policy Condition:* Another type of policy modification could involve adding a condition into one of the policies in $P$.

**Given:**

- $PS = \langle P, W_C \rangle$
- $p_* = \langle C, A \rangle$ such that $p_* \in P$
- $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C \cup c_*$

There are several ways in which adding a policy condition could impact the system's metrics structure (see Definition 4):

In the first case, adding a policy condition may result in an increase in the number of system metrics. This may be a result of adding a policy condition whose `metricName` is not in any of the conditions of the policies in $P$.

Policy Modification - $\triangle_1$[P]: *Adding a policy condition resulting in a new metric.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ where $\forall\ p_i \in P$, $c_* \notin p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \notin M$
V) $r_m \notin M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
   - $\alpha_m = \langle$`ID, metricName,` $\omega \rangle$
   - $\sigma_m = \{\Gamma\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C \cup c_*$
2) $M' \leftarrow M \cup m$
3) $M_R^{P'} \leftarrow M_R^P \cup r_m$

In the second case, adding a policy condition may result in changes to the regions of an existing metric; i.e., $r_m \in M_R^P$. This may be a result of adding a policy condition whose `metricName` is in at least one of the conditions of the policies in $P$ and where there is no occurrence of the condition within the policies in $P$.

Policy Modification - $\triangle_2$[P]: *Adding a policy condition resulting in changes to the regions of existing metrics.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ where $\forall\ p_i \in P$, $c_* \notin p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \in M$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
   - $\alpha_m = \langle$`ID, metricName,` $\omega \rangle$
   - $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_j, \Gamma_{j+1}, \ldots, \Gamma_k\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C \cup c_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P : \exists\ r_m \in M_R^P$: $r'_m = \langle \alpha'_m, \sigma'_m \rangle$ :
   a) $\alpha'_m = \langle$`ID, metricName,` $\omega \rangle$
   b) $\sigma'_m = \sigma_m \cup c_*.\Gamma : \sigma'_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_j, \mathbf{\Gamma}, \Gamma_{j+1}, \ldots, \Gamma_k\}$ where $\Gamma_j < \mathbf{\Gamma} < \Gamma_{j+1}$ for some $j$.

In the third case, adding a policy condition may result in no changes to the number of metrics and their regions. This may be a result of adding a policy condition that is identical (i.e., share the same `metricName`, `operator`, and $\Gamma$) to at least one of the conditions of the policies in $P$.

Policy Modification - $\triangle_3$[P]: *Adding a policy condition resulting in no changes to the metrics' regions.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ and where $\exists\ p_i \in P$ such that $c_* \in p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \in M$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
   - $\alpha_m = \langle$`ID, metricName,` $\omega \rangle$
   - $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{j-1}, \Gamma_j, \Gamma_{j+1} \ldots \Gamma_k\}$ : $c_*.\Gamma = \Gamma_j$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C \cup c_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

*3) Removing a Policy Condition:* Another type of policy modification could involve removing a condition from one of the policies in $P$.

**Given:**

- $PS = \langle P, W_C \rangle$
- $p_* = \langle C, A \rangle$ such that $p_* \in P$ and $c_* \in C$
- $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C|c_*$

There are several ways in which removing a policy condition could impact the system's metrics structure:

In the first case, removing a policy condition may result in a decrease in the number of system metrics. This may be a result of removing a policy condition whose `metricName` is not in any of the remaining conditions of the policies in $P$; i.e., there is only a single occurrence of the condition with the given `metricName` within the policies in $P$.

Policy Modification - $\triangle_4$[P]: *Removing a policy condition resulting in a decrease in the number of metrics.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ and $c_* \in C$ where $\forall\ p_i \in P|p_*$, $c_* \notin p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \in M$, such that, $\forall\ m_i \in M|m$, $m_i \neq m$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$
- $\sigma_m = \{\Gamma\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C|c_*$
2) $M' \leftarrow M|m$
3) $M_R^{P'} \leftarrow M_R^P|r_m$

In the second case, removing a policy condition may result in changes to the regions of existing metrics. This may be a result of removing a policy condition whose `metricName` is in at least one of the conditions of the policies in $P$ and where there is only a single occurrence of the condition within the policies in $P$.

Policy Modification - $\triangle_5$[P]: *Removing a policy condition resulting in changes to the regions of existing metrics.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ and $c_* \in C$ where $\forall\ p_i \in P|p_*$, $c_* \notin p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` where $m \in M$, and where $\exists\ m_i \in M|m$ such that $m = m_i$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{j-1}, \mathbf{\Gamma_j}, \Gamma_{j+1}, \ldots, \Gamma_k\}$ : $c_*.\Gamma = \Gamma_j$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C|c_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P : \exists\ r_m \in M_R^P : r'_m = \langle \alpha'_m, \sigma'_m \rangle$ :
   a) $\alpha'_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$
   b) $\sigma'_m = \sigma_m|c_*.\Gamma : \sigma'_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{j-1}, \Gamma_{j+1}, \ldots, \Gamma_k\}$

In the third case, removing a policy condition may result in no changes to the number of metrics and their regions. This may be a result of removing a policy condition that is identical (i.e., share the same `metricName`, `operator`, and $\Gamma$) to at least one of the conditions of the policies in $P$; i.e., there are multiple occurrences of the condition within the policies in $P$.

Policy Modification - $\triangle_6$[P]: *Removing a policy condition resulting in no changes to the number of metrics and their regions.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ and $c_* \in C$ where $\exists\ p_i \in P|p_*$ such that $c_* \in p_i$
IV) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \in M$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{j-1}, \mathbf{\Gamma_j}, \Gamma_{j+1}, \ldots, \Gamma_k\}$ : $c_*.\Gamma = \Gamma_j$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C', A \rangle$ where $C' \leftarrow C|c_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

*4) Modifying a Policy Condition:* Another type of policy modification could involve modifying the threshold of a policy condition within the conditions of the policies in $P$.

Policy Modification - $\triangle_7$[P]: *Modifying the threshold of a policy condition.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $c_* = \langle$`ID, metricName, operator,` $\Gamma \rangle$ such that $m = c_*$.`metricName` and where $m \in M$
IV) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{j-1}, \Gamma_j, \Gamma_{j+1}, \ldots, \Gamma_k\}$ : $c_*.\Gamma = \Gamma_j$

**Then:**

1) $PS' = \langle P', W_C \rangle : c'_* = \langle$`ID, metricName, operator,` $\Gamma' \rangle\ \forall\ p_* \in P : c_* \in p_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P : \exists\ r_m \in M_R^P : r'_m = \langle \alpha'_m, \sigma'_m \rangle$ :
   a) $\alpha'_m = \langle \mathtt{ID}, \mathtt{metricName}, \omega \rangle$

b) $\sigma'_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_l, \Gamma', \Gamma_{l+1}, \ldots, \Gamma_k\}$ where $\Gamma_l < \Gamma' < \Gamma_{l+1}$ for some $l$

*5) Adding a Policy Action:* Another type of policy modification could involve adding a policy action onto one of the policies in $P$.

Policy Modification - $\triangle_8$[P]: *Adding a policy action onto a policy in $P$.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$
IV) $a_* = \langle$`ID, function, parameters,` $\tau \rangle$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle$`ID,metricName,` $\omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_k\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C, A' \rangle$ where $A' \leftarrow A \cup a_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

*6) Removing a Policy Action:* Another type of policy modification could involve removing a policy action from one of the policies in $P$.

Policy Modification - $\triangle_9$[P]: *Removing an action from a policy in $P$.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_* = \langle C, A \rangle$ such that $p_* \in P$ and $a_* \in A$
IV) $a_* = \langle$`ID, function, parameters,` $\tau \rangle$
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle$`ID,metricName,` $\omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_k\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : p'_* = \langle C, A' \rangle$ where $A' \leftarrow A | a_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

*7) Modifying a Policy Action:* Another type of policy modification could involve changing the attributes (i.e., `parameters`, or $\tau$) of a policy action within the actions of the policies in $P$.

Policy Modification - $\triangle_{10}$[P]: *Modifying the attributes of a policy action.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $a_* = \langle$`ID, function, parameters,` $\tau \rangle$
IV) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle$`ID,metricName,` $\omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_k\}$

**Then:**

1) $PS' = \langle P', W_C \rangle : a'_* = \langle$`ID, function, parameters',` $\tau' \rangle \; \forall \; p_* \in P : a_* \in p_*$
2) $M' \leftarrow M$
3) $M_R^{P'} \leftarrow M_R^P$

### B. Adaptation Transformations

This section presents our approach for adapting the model learned from the use of an active set of policies to the policy modifications described in Section III-A. We have identified several cases relating to the types of changes to an active set of policies, $P$, in the context of how they might impact the model (i.e., the state-transition graph) leaned from using $P$. For illustration purposes, this section assumes that initially, set $P$ consists of the expectation policies depicted in Figure 1. Suppose also that the current model of the system (i.e., $G_n^P = \langle S, T \rangle$), as depicted in Figure 4, includes the seven states shown in Table V. Note that, states $s_2$ and $s_6$ are considered "non-violation" whereas the other states are considered "violation" states (see $\mu$ in Definition 6). The focus of our discussion centers on how the various changes to the policies in $P$ might affect the original model.

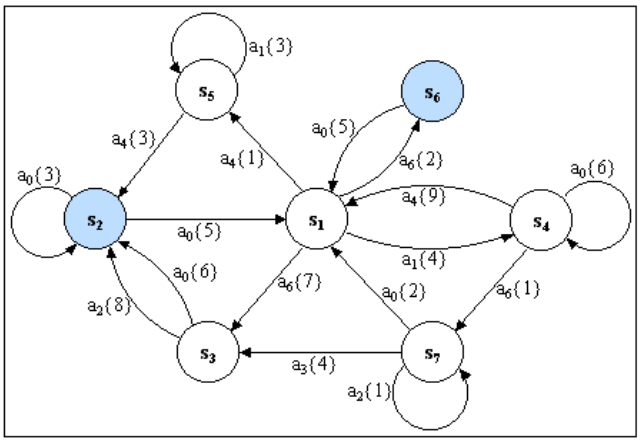



Fig. 4. A sample state transition model.

*1) Adapting to $\triangle[P]$ Affecting States Actions:* This describes our approach for dealing with policy modifications that only impact the actions within states and not states transitions. From the definitions of Section III-A, this may be in response to policy modification $\triangle_8[P]$ or $\triangle_{10}[P]$.

Model Transformation - $\Psi_1[G_n^P]$: *Adapting to policy modification $\triangle_8[P]$ or $\triangle_{10}[P]$.*

**Given:**

I) $G_n^P = \langle S, T \rangle$ :
- $\forall \; s_i \in S, \; s_i = \langle \mu, M(s_i), P(s_i), A(s_i) \rangle : s_i.m_k = \langle$`ID,` $\omega$`,value,` $R^l_{m_k} \rangle$ where $s_i.m_k \in M(s_i)$
- $\forall \; t(s_i, a, s_j) \in T, \; t(s_i, a, s_j) = \langle \lambda, Q_t(s_i, a) \rangle$

II) Policy modification $\triangle_8[P]$ or $\triangle_{10}[P]$

**Then:** $G_{n+1}^{P'} = \langle S', T' \rangle$ where

1) $S' \leftarrow S$ :

| | $f(R^k_{m_j})$ | | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $f(R^k_{m_1})$ | $f(R^k_{m_2})$ | $f(R^k_{m_3})$ | $f(R^k_{m_4})$ | $f(R^k_{m_5})$ | $f(R^k_{m_6})$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 4 | $s_4$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1 | $s_5$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 7;2 | $s_3$;$s_6$ |
| $s_2$ | 0 | 100 | 0 | 100 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_1$;$s_2$ |
| $s_3$ | 50 | 100 | 0 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 6 | $s_2$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 8 | $s_2$ |
| $s_4$ | 0 | 100 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_1$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 1 | $s_7$ |
| $s_5$ | 0 | 0 | 100 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 3 | $s_5$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 3 | $s_2$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | | |
| $s_6$ | 50 | 100 | 0 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 5 | $s_1$ |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_3$ |

TABLE V
SAMPLE STATES BASED ON THE METRICS STRUCTURE OF TABLE II AND THE STATE-TRANSITION MODEL OF FIGURE 4.

a) $\forall\ s_i \in S : p_* \in P'(s_i);\ A'(s_i) \leftarrow A(s_i) \cup a_*\ (\triangle_8[P])$
b) $\forall\ s_i \in S;\ a_* \leftarrow a'_* \iff \exists\ a \in A(s_i) : a_* = a\ (\triangle_{10}[P])$

2) $T' \leftarrow T$

Suppose, for example, that policy modification $\triangle_{10}[P]$ involves modifying policy action "AdjustMaxBandwidth(-128)" (which corresponds to action $a_6$ in Table V) to "AdjustMaxBandwidth(-64)" in the policies of Figure 1. The following describes the steps involved in transforming the state-transition model of Figure 5(a), whose states are shown in Table V and correspond to the steps of model-transformation $\Psi_1[G^P_n]$:

1) For each state that has been encountered to date (see Table V), the agent must update the actions within the states to ensure that the appropriate action is modified (see Step 1(b)). Since action $a_6$ was initially in actions sets $A(s_1)$, $A(s_4)$, and $A(s_5)$, each occurrence of the action would be modified to "AdjustMaxBandwidth(-64)" as illustrated in Figure 5(c).
2) This change, however, would not affect the transitions associated with the modified action. Consider, for example, transitions $t(s_1, a_6\{7\}, s_3)$ and $t(s_1, a_6\{2\}, s_6)$, both involving action $a_6$ (see Figure 5(a)). The fact that the action is still part of the actions within the states means that the agent may still take such transitions. Hence, no changes would be made to the transitions unless they are encountered in the future, in which case action-value estimates and frequencies will be updates accordingly.

*2) Adapting to $\triangle[P]$ Affecting States Transitions:* This describes our approach for dealing with policy modifications that only affect the state transitions as a result of modifications to the composition of policy actions within states. From the definitions of Section III-A, this may be in response to policy modification $\triangle_3[P]$, $\triangle_6[P]$, or $\triangle_9[P]$. Our approach essentially involves ensuring that the state transitions set $T(s_i)$ is consistent with the state actions set $A(s_i)$.

Model Transformation - $\Psi_2[G^P_n]$: *Adapting to policy modification $\triangle_3[P]$, $\triangle_6[P]$, or $\triangle_9[P]$.*

**Given:**

I) $G^P_n = \langle S, T\rangle$ :
- $\forall\ s_i \in S,\ s_i = \langle \mu, M(s_i), P(s_i), A(s_i)\rangle$ : $s_i.m_k = \langle \texttt{ID}, \omega, \texttt{value}, R^l_{m_k}\rangle$ where $s_i.m_k \in M(s_i)$
- $\forall\ t(s_i, a, s_j) \in T,\ t(s_i, a, s_j) = \langle \lambda, Q_t(s_i, a)\rangle$

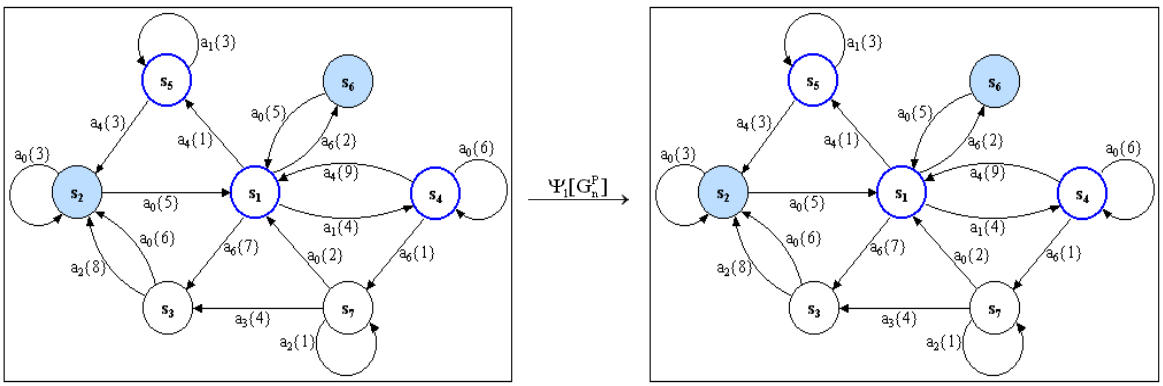
II) Policy modification $\triangle_3[P]$, $\triangle_6[P]$, or $\triangle_9[P]$

**Then:** $G^{P'}_{n+1} = \langle S', T'\rangle$ where

1) $S' \leftarrow S : \forall\ s_i \in S\ ;\ a \in A'(s_i) \iff a \in \{P'(s_i) \cup a_0\}$
2) $\forall\ t(s_i, a_*, s_j) \in T(s_i) : a_* \notin A'(s_i);\ T' \leftarrow T|t(s_i, a_*, s_j)$
3) $\forall\ a_i \in A(s_i : \exists\ t(s_i, a, s_*) : s_i, s_* \in S'$, compute $Q(s_i, a)$ (see Equation 2).

$$Q(s,a) = \sum_{t_i(s,a,s'_j)\in T(s)} Pr[t_i(s,a,s'_j)] \times Q_{t_i}(s,a) \quad (2)$$

Suppose, for example, that policy modification $\triangle_9[P]$ involves deleting policy action "AdjustMaxBandwidth(-128)" (which corresponds to action $a_6$ in Table V) from policy $p_2$ in Figure 1. The following describes the steps involved in transforming the state-transition model of Figure 6(a), whose states are shown in Table V and correspond to the steps of model-transformation $\Psi_2[G^P_n]$:




(a) Original model: $G_n^P = \langle S, T \rangle$

(b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R_{m_j}^k)$ | | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 4 | $s_4$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1 | $s_5$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-64) | 7;2 | $s_3$;$s_6$ |
| $s_2$ | 0 | 100 | 0 | 100 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_1$;$s_2$ |
| $s_3$ | 50 | 100 | 0 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 6 | $s_2$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 8 | $s_2$ |
| $s_4$ | 0 | 100 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_1$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-64) | 1 | $s_7$ |
| $s_5$ | 0 | 0 | 100 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 3 | $s_5$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 3 | $s_2$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-64) | | |
| $s_6$ | 50 | 100 | 0 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 5 | $s_1$ |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_3$ |

(c) System states of the new model.

Fig. 5. State-transition model after $\triangle_{10}[P]$; i.e., modifying action $a_6$ ="AdjustMaxBandwidth(-128)" to "AdjustMaxBandwidth(-64)" in the policies of Figure 1.

1) Since policy $p_2$ is the only violated policy in state $s_4$, $\triangle_9[P]$ would result in the removal of action $a_6$ from the actions set $A(s_4)$ as illustrated in Figure 6(c). While policy $p_2$ is also violated in states $s_1$ and $s_5$, $\triangle_9[P]$ would not affect such states. This is because action $a_6$ is also part of other violated policies in those states, such as policy $p_1$ (i.e., $a_6 \in p_1$) where $p_1 \in P'(s_1)$ and $p_1 \in P'(s_5)$.
2) The removal of action $a_6$ from state $s_4$ in the above step would mean that transition $t(s_4, a_6\{1\}, s_7)$ (see Figure 6(a)) is no longer valid. As such, the transition will be removed as illustrated in Figure 6(b).
3) The final step of the adaptation is for the agent to perform backup updates so that any impact on the model as a result of removing transitions is propagated to the action-value estimates of the states actions.

*3) Adapting to $\triangle[P]$ Affecting Metrics Regions:* This describes our approach for dealing with policy modifications that affects the regions within state metrics. From the definitions of Section III-A, this may be in response to policy modification $\triangle_2[P]$, $\triangle_5[P]$, or $\triangle_7[P]$.

Model Transformation - $\Psi_3[G_n^P]$: *Adapting to policy modification $\triangle_2[P]$, $\triangle_5[P]$, or $\triangle_7[P]$.*

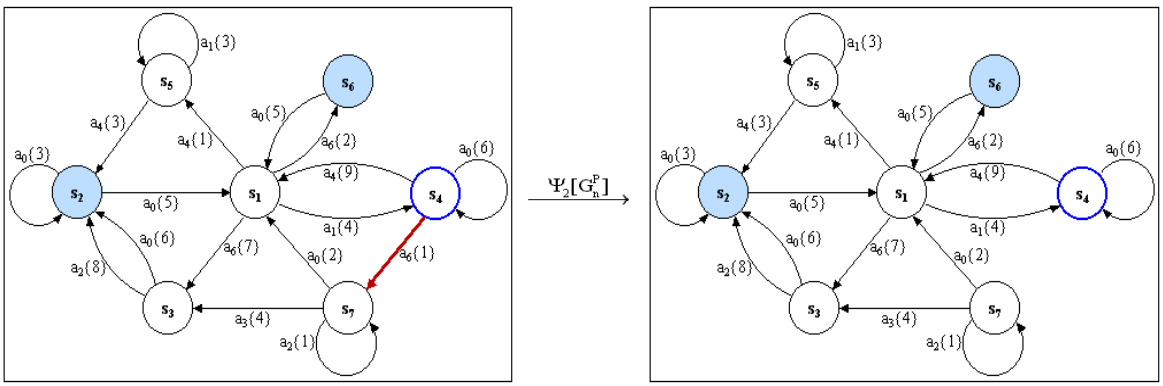
**Given:**

I) $G_n^P = \langle S, T \rangle$ :
- $\forall\ s_i \in S,\ s_i = \langle \mu, M(s_i), P(s_i), A(s_i) \rangle$ : $s_i.m_k = \langle \texttt{ID}, \omega, \texttt{value}, R_{m_k}^l \rangle$ where $s_i.m_k \in M(s_i)$
- $\forall\ t(s_i, a, s_j) \in T,\ t(s_i, a, s_j) = \langle \lambda, Q_t(s_i, a) \rangle$

II) Policy modification $\triangle_4[P]$ or $\triangle_5[P]$ or $\triangle_7[P]$

**Then:** $G_{n+1}^{P'} = \langle S', T' \rangle$ where

1) $S' \leftarrow S : \forall\ s_i \in S$;
   a) $s_i.m.R_m^l \leftarrow (\Gamma_l, \Gamma_{l+1}) : \sigma'_m.\Gamma_l < s_i.m.\texttt{value} < \sigma'_m.\Gamma_{l+1}$
   b) $a \in A'(s_i) \iff a \in \{P'(s_i) \cup a_0\}$




(a) Original model: $G_n^P = \langle S, T \rangle$

(b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R^k_{m_j})$ | | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 4 | $s_4$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1 | $s_5$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 7;2 | $s_3$;$s_6$ |
| $s_2$ | 0 | 100 | 0 | 100 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_1$;$s_2$ |
| $s_3$ | 50 | 100 | 0 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 6 | $s_2$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 8 | $s_2$ |
| $s_4$ | 0 | 100 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_1$ |
| $s_5$ | 0 | 0 | 100 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 3 | $s_5$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 3 | $s_2$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | | |
| $s_6$ | 50 | 100 | 0 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 5 | $s_1$ |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_3$ |

(c) System states of the new model.

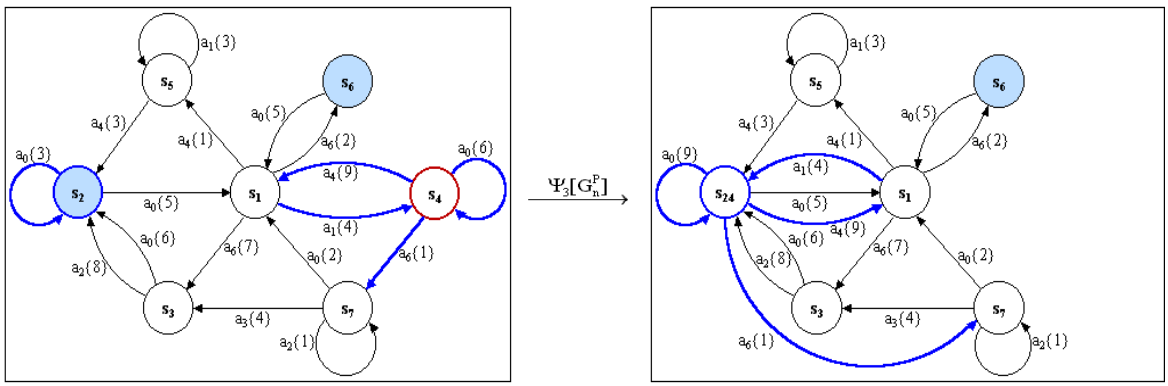
Fig. 6. State-transition model after $\triangle_9[P]$; i.e., removing action $a_6$ ="AdjustMaxBandwidth(-128)" from policy $p_2$ in Figure 1.

| $m_i$ | $c_k$ | Policy Condition | $R_{m_i}$ | $R^j_{m_i}$ | $f(R^j_{m_i})$ |
|---|---|---|---|---|---|
| $m_6$ | 7 | APACHE:responseTimeTREND > -1.0 | $m_6$.value $\leq -1.0$ | $R^1_{m_6}$ | 100 |
| | | | $m_6$.value $> -1.0$ | $R^2_{m_6}$ | 0 |

TABLE VI
METRIC $m_6$ STRUCTURE AFTER $\triangle_7[P]$; I.E., MODIFYING THE THRESHOLD OF CONDITION $c_7$ ="APACHE:responseTimeTREND > 0.0" TO "APACHE:responseTimeTREND > -1.0" IN THE POLICIES OF FIGURE 1.

2) $\forall\ t(s_i, a_*, s_j) \in T(s_i) : a_* \notin A'(s_i)$; $T' \leftarrow T|t(s_i, a_*, s_j)$
3) $\forall\ s_i, s_j \in S' : s_i = s_j$ where $i \neq j$;
   a) $T'(s_i) \leftarrow T'(s_i) \cup T'(s_j)$
      $\forall\ t(s_i, a, s) \in T'(s_i)$ and $\forall\ t'(s_j, a', s') \in T'(s_j) : a = a'$ and $s = s'$;
      i) $Q_t(s_i, a) \leftarrow \frac{t.\lambda \times Q_t(s_i, a) + t'.\lambda \times Q_{t'}(s_j, a')}{t.\lambda + t'.\lambda}$
      ii) $t.\lambda \leftarrow t.\lambda + t'.\lambda$
      iii) $T' \leftarrow T|t'(s_j, a', s')$
   b) $S' \leftarrow S'|s_j$
4) $\forall\ a_i \in A(s_i : \exists\ t(s_i, a, s_*) : s_i, s_* \in S'$, compute $Q(s_i, a)$ (see Equation 2).

Suppose, for example, that policy modification $\triangle_7[P]$ involves modifying the threshold of policy condition $c_7$ ="APACHE: responseTimeTREND > 0.0" (which corresponds to metric $m_6$ in Table II) to "APACHE: responseTimeTREND > -1.0" in the policies of Figure 1. Thus, instead of $\sigma_{m_6} = \{0.0\}$, as was the case prior to $\triangle_7[P]$ (see Table II), the new regions will be defined by $\sigma_{m_6} = \{-1.0\}$, as is illustrated in Table VI. Suppose also that the following measurements correspond to the average value of metric $m_6$; "$s_2.m_6$.value $= -0.5$", "$s_3.m_6$.value $= -2.0$", "$s_6.m_6$.value $= -1.5$", and




(a) Original model: $G_n^P = \langle S, T \rangle$

(b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R^k_{m_j})$ | | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 4 | $s_4$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1 | $s_5$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 7;2 | $s_3$;$s_6$ |
| $s_{24}$ | 0 | 100 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | 5;9 | $s_1$;$s_{24}$ |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_1$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 1 | $s_7$ |
| $s_3$ | 50 | 100 | 0 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 6 | $s_2$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 8 | $s_2$ |
| $s_5$ | 0 | 0 | 100 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | | $a_1$ | AdjustMaxClients(+25) | 3 | $s_5$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 3 | $s_2$ |
| | | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | | |
| $s_6$ | 50 | 100 | 0 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 5 | $s_1$ |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_3$ |

(c) System states of the new model.

Fig. 7. State-transition model after $\triangle_7[P]$; i.e., modifying the threshold of condition $c_7$ ="APACHE:responseTimeTREND > 0.0" to "APACHE:responseTimeTREND > -1.0" in the policies of Figure 1.

"$s_7.m_6.\texttt{value} = -2.5$". The following describes the steps involved in transforming the state-transition model of Figure 7(a), whose states are shown in Table V and correspond to the steps of model-transformation $\Psi_3[G_n^P]$:

1) Since "$m_6.\texttt{value} < -1.0$" in states $s_3$, $s_6$, and $s_7$, $f(R^l_{m_6})$ would not be affected by the policy modification in those states since the measurements would remain in region $R^1_{m_6}$. However, since "$s_2.m_6.\texttt{value} = -0.5$", $f(R^k_{m_6})$ would recompute the region value by changing it from 100 (see Table V) to 0 since the measurement would now fall in region $R^2_{m_6}$ (see Table VI) instead of the previous region $R^1_{m_6}$ (see Table II). The agent must also update the states actions sets of each state to ensure that actions set $A'(s_i)$ contains only the actions of the policies that are violated when the system is in state $s_i$ (see Step 1(b)). Note that $\triangle_7[P]$ would result in the violation of policy $p_2$ when the agent is in state $s_2$. This was not the case prior to the policy modification since only one of the policy conditions was violated. As such, actions $a_1$, $a_4$, and $a_6$ would be added onto the actions set $A(s_2)$; i.e., $A'(s_2) = \{a_0, a_1, a_4, a_6\}$.
2) Since the previous stage did not result in the deletion of states actions, no transition will be affected.
3) Changing the region from $R^1_{m_6}$ to $R^2_{m_6}$ in state $s_2$ would mean states $s_2$ and $s_4$ are identical; i.e., they have the same region value assignment (i.e., $f(R^l_{m_i})$) for each identical metric. Thus, the two states would be merged together onto state $s_{24}$, as illustrated in Figure 7. This would also involve merging the states transitions such that transition $t(s_4, a_6\{1\}, s_7)$ becomes transition $t(s_{24}, a_6\{1\}, s_7)$, while $t(s_4, a_4\{9\}, s_1)$ becomes transition $t(s_{24}, a_4\{9\}, s_1)$ (see Step 3(a)). In the case of two identical transitions such as $t(s_4, a_0\{6\}, s_4)$ and $t(s_2, a_0\{3\}, s_2)$, a new transition $t(s_{24}, a_0\{9\}, s_{24})$ would be formed as illustrated in Figure 7(b) such that the frequency of the new transition is the sum of the frequencies of the two merged transitions. Once the transitions have been updated, the duplicate state is removed (see Step 3(b)).

4) The final step of the adaptation is for the agent to perform backup updates so that any impact on the model as a result of merging states and transitions is propagated to the action-value estimates of the states actions.

*4) Adapting to $\triangle[P]$ Resulting in a Decrease in State Metrics:* This describes our approach for dealing with policy modifications that reduces the size of the metrics set $M$, possibly resulting in two or more identical states. As mentioned previously, two states are said to be identical if they share the same metrics region values as determined by the mapping $f(R^j_{m_i})$ (see Equation 1) based on each metric's measurement (i.e., $m_i$.value). From the definitions of Section III-A, this may be in response to policy modification $\triangle_4[P]$.

Model Transformation - $\Psi_4[G^P_n]$: *Adapting to policy modification $\triangle_4[P]$.*

**Given:**

I) $G^P_n = \langle S, T\rangle$ :
- $\forall\ s_i \in S,\ s_i = \langle \mu, M(s_i), P(s_i), A(s_i)\rangle$ : $s_i.m_k = \langle \texttt{ID}, \omega, \texttt{value}, R^l_{m_k}\rangle$ where $s_i.m_k \in M(s_i)$
- $\forall\ t(s_i, a, s_j) \in T,\ t(s_i, a, s_j) = \langle \lambda, Q_t(s_i, a)\rangle$

II) Policy modification $\triangle_4[P]$

**Then:** $G^{P'}_{n+1} = \langle S', T'\rangle$ where

1) $S' \leftarrow S : \forall\ s_i \in S$;
   a) $M'(s_j) \leftarrow M(s_j)|m : m$ is the deleted metric
   b) $a \in A'(s_i) \iff a \in \{P'(s_i) \cup a_0\}$
2) $\forall\ t(s_i, a_*, s_j) \in T(s_i) : a_* \notin A'(s_i)$; $T' \leftarrow T|t(s_i, a_*, s_j)$
3) $\forall\ s_i, s_j \in S' : s_i = s_j$ where $i \neq j$;
   a) $T'(s_i) \leftarrow T'(s_i) \cup T'(s_j)$ $\forall\ t(s_i, a, s) \in T'(s_i)$ and $\forall\ t'(s_j, a', s') \in T'(s_j) : a = a'$ and $s = s'$;
      i) $Q_t(s_i, a) \leftarrow \dfrac{t.\lambda \times Q_t(s_i, a) + t'.\lambda \times Q_{t'}(s_j, a')}{t.\lambda + t'.\lambda}$
      ii) $t.\lambda \leftarrow t.\lambda + t'.\lambda$
      iii) $T' \leftarrow T|t'(s_j, a', s')$
   b) $S' \leftarrow S'|s_j$
4) $\forall\ a_i \in A(s_i : \exists\ t(s_i, a, s_*) : s_i, s_* \in S'$, compute $Q(s_i, a)$ (see Equation 2).

Suppose, for example, that policy modification $\triangle_4[P]$ involves removing policy condition $c_5$ ="MEMORY:utilizationTREND > 0.0" (which corresponds to metric $m_4$ in Table II) from policy $p_4$ in Figure 1. The following describes the steps involved in transforming the state-transition model of Figure 8(a), whose states are shown in Table V and correspond to the steps of model-transformation $\Psi_4[G^P_n]$:

1) For each state that has been encountered to date (see Table V), the agent must first remove all the information associated with metric $m_4$ (see step 1(a)). This would result in a system's model with seven states, each with five metrics; i.e., $M(s_i) = \{m_1, m_2, m_3, m_5, m_6\}$. The agent must also update the states actions sets of each state to ensure that actions set $A'(s_i)$ contains only the actions of the policies that are violated when the system is in state $s_i$ (see Step 1(b)). Note that $\triangle_4[P]$ would result in the violation of policy $p_4$ when the agent is in states $s_2$, $s_4$, or $s_6$. This was not the case prior to the policy modification since only one of the policy conditions was violated; i.e., $f(R^l_{m_1}) = 0$ while $f(R^l_{m_4}) = 100$. Thus, removing $m_4$ from state the above states would mean that $p_4$ is now violated. As such, action $a_2$ would be added onto actions sets $A(s_2)$, $A(s_4)$, and $A(s_6)$, making states $s_2$ and $s_6$ violation states.
2) The state-transition model must also be updated to ensure that the actions within states transitions are consistent with the updated states actions sets. Since the previous stage did not result in the deletion of states actions, no transition will be affected.
3) The agent must then determine whether any states should be merged together if, in fact, the modifications above may have resulted in two or more identical states. Note that the deletion of metric $m_4$ would mean states $s_3$ and $s_6$ are identical; i.e., they have the same region value assignment (i.e., $f(R^l_{m_i})$) for each identical metric. Thus, the two states would be merged together onto state $s_{36}$, as illustrated in Figure 8. This would also involve merging the states transitions such that transition $t(s_6, a_0\{5\}, s_1)$ becomes transition $t(s_{36}, a_0\{5\}, s_1)$ (see Step 3(a)). In the case of two identical transitions such as $t(s_1, a_6\{2\}, s_6)$ and $t(s_1, a_6\{7\}, s_3)$, a new transition $t(s_1, a_6\{9\}, s_{36})$ would be formed, as illustrated in Figure 8(b), such that the frequency of the new transition is the sum of the frequencies of the two merged transitions. Once the transitions have been updated, the duplicate state is removed (see Step 3(b)).
4) The final step of the adaptation is for the agent to perform backup updates so that any impact on the model as a result of removing transitions and/or merging states is propagated to the action-value estimates of the states actions.

*5) Adapting to $\triangle[P]$ Resulting in an Increase in State Metrics:* This describes our approach for dealing with policy modifications that increase the size of the metrics set $M$. From the definitions of Section III-A, this may be in response to policy modification $\triangle_1[P]$.

Model Transformation - $\Psi_5[G^P_n]$: *Adapting to policy modification $\triangle_1[P]$.*

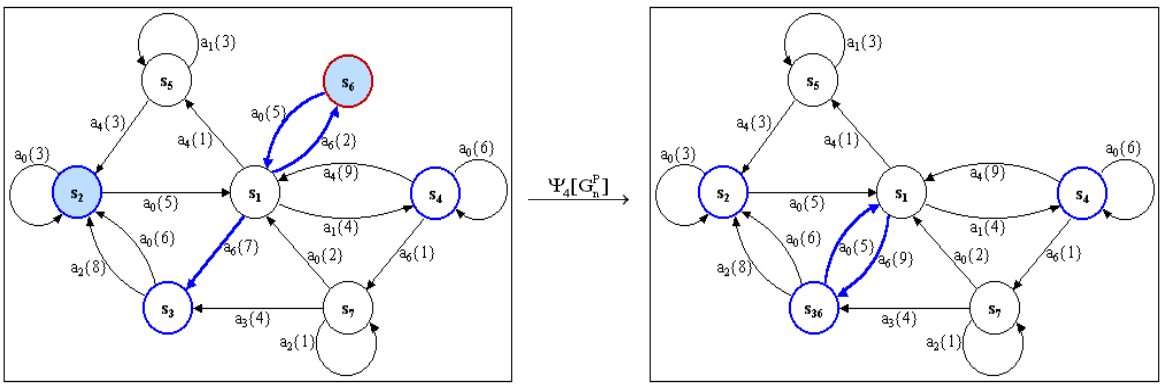
**Given:**

I) $G^P_n = \langle S, T\rangle$ :
- $\forall\ s_i \in S,\ s_i = \langle \mu, M(s_i), P(s_i), A(s_i)\rangle$ : $s_i.m_k = \langle \texttt{ID}, \omega, \texttt{value}, R^l_{m_k}\rangle$ where $s_i.m_k \in M(s_i)$
- $\forall\ t(s_i, a, s_j) \in T,\ t(s_i, a, s_j) = \langle \lambda, Q_t(s_i, a)\rangle$

II) Policy modification $\triangle_1[P]$

**Then:** $G^{P'}_{n+1} = \langle S', T'\rangle$ where

1) $S' \leftarrow \{\emptyset\}$




(a) Original model: $G_n^P = \langle S, T \rangle$ (b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R^k_{m_j})$ | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_3$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | $a_1$ | AdjustMaxClients(+25) | 4 | $s_4$ |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1 | $s_5$ |
| | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 7;2 | $s_3$;$s_6$ |
| $s_2$ | 0 | 100 | 0 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_1$;$s_2$ |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| $s_{36}$ | 50 | 100 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 5;6 | $s_1$;$s_2$ |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | 8 | $s_2$ |
| $s_4$ | 0 | 100 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_1$ |
| | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 1 | $s_7$ |
| $s_5$ | 0 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | $a_1$ | AdjustMaxClients(+25) | 3 | $s_5$ |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 3 | $s_2$ |
| | | | | | | $a_6$ | AdjustMaxBandwidth(-128) | | |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_3$ |

(c) System states of the new model.

Fig. 8. State-transition model after $\triangle_4[P]$; i.e., deleting condition $c_5$ ="MEMORY:utilizationTREND > 0.0" from policy $p_4$ in Figure 1.

| $m_i$ | $c_k$ | Policy Condition | $R_{m_i}$ | $R^j_{m_i}$ | $f(R^j_{m_i})$ |
|---|---|---|---|---|---|
| $m_7$ | 10 | APACHE:refusedRequests < 10.0 | $m_7$.value $\leq$ 10.0 | $R^1_{m_7}$ | 100 |
| | | | $m_7$.value > 10.0 | $R^2_{m_7}$ | 0 |

TABLE VII
METRICS $m_7$ STRUCTURE AFTER $\triangle_1[P]$; I.E., ADDING CONDITION $c_{10}$ ="APACHE:refusedRequests < 10.0" ONTO POLICY $p_5$ IN FIGURE 1.

2) $T' \leftarrow \{\emptyset\}$

Suppose, for example, that policy modification $\triangle_1[P]$ involves adding the condition $c_{10}$ ="APACHE:refusedRequests < 10.0" onto policy $p_5$ in Figure 1. This would result in a new state metric (i.e., $m_7$ ="APACHE:refusedRequests") with two regions associated with it, whose structure is shown in Table VII. For a system whose model is derived from visiting the states in Table V, such a change to the policies set $P$ would mean a binary split of each of the seven states to account for the two regions of the new metric. However, since no measurements associated with metrics $m_7$ would have been collected as was pointed out in Section III-B5, there would be no way of knowing how to map the old transitions onto the new states set. As such, $\triangle_1[P]$ is the only policy modification requiring learning a new model from scratch.

### C. A Sequence of Transformations

Consider a more complex example where $\triangle[P]$ involves removing policy $p_4$ in Figure 1; i.e.,

**Given:**

I) $PS = \langle P, W_C \rangle$

II) $p_4 = \langle C, A \rangle$ such that $p_4 \in P$ :

- $C = \{c_4, c_5\}$ :
  a) $c_4$ = "`MEMORY:utilization > 40.0`" : $m_3$ = "`MEMORY:utilization`" and where $m_3 \in M : \forall\ m_i \in M|m_3,\ m_i \neq m_3$ and where $\forall\ p_i \in P|p_4,\ c_4 \notin p_i$
  b) $c_5$ = "`MEMORY:utilizationTREND > 0.0`" : $m_4$ = "`MEMORY:utilizationTREND`" and where $m_4 \in M : \forall\ m_i \in M|m_4$, $m_i \neq m_4$ and where $\forall\ p_i \in P|p_4$, $c_5 \notin p_i$
- $A = \{a_2\}$ : $a_2$ = `AdjustMaxClients(-25)`

**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P|p_4$

Note that since the policy consists of two conditions and one action, such a policy modification can be modelled as a series of transformations involving the following:

1) Modifying $p_4$ to create $p_4^1$ by removing condition $c_5$ from policy $p_4 = \langle \{c_4, c_5\}, \{a_2\} \rangle$ such that $\triangle_4[P] = P_1$.
2) Modifying $p_4^1$ to create $p_4^2$ by removing action $a_2$ from policy $p_4^1 = \langle \{c_4\}, \{a_2\} \rangle$ such that $\triangle_9[P_1] = P_2$.
3) Modifying $p_4^2$ to create $p_4^3$ by removing condition $c_4$ from policy $p_4^2 = \langle \{c_5\}, \{\emptyset\} \rangle$ such that $\triangle_4[P_2] = P_3$.
4) Removing policy $p_4^3 = \langle \{\emptyset\}, \{\emptyset\} \rangle$ from $P_3$ such that $\triangle_y[P_3] = P'$.

In the following, we expand on each of these transformations, elaborating on the steps involved in transforming the state-transition model whose states are shown in Table V.

The first transformation involves adapting the state-transition model as a result of removing policy condition $c_5$ ="`MEMORY:utilizationTREND > 0.0`" from policy $p_4$, which relates to policy modification $\triangle_4[P]$; i.e.,

Policy Modification - $\triangle_4[P]$: *Removing a policy condition resulting in a decrease in the number of metrics.*

**Given:**

I) $PS = \langle P, W_C \rangle$
II) $M_R^P = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_4 = \langle \{c_4, c_5\}, \{a_2\} \rangle$ such that $p_4 \in P$ where $\forall\ p_i \in P|p_4,\ c_5 \notin p_i$
IV) $c_5$ = "`MEMORY:utilizationTREND > 0.0`" : $m_4$ = "`MEMORY:utilizationTREND`" and where $m_4 \in M : \forall\ m_i \in M|m_4,\ m_i \neq m_4$
V) $r_{m_4} \in M_R^P : r_{m_4} = \langle \alpha_{m_4}, \sigma_{m_4} \rangle$ :
- $\alpha_{m_4} = \langle 4, m_4, \omega \rangle$
- $\sigma_{m_4} = \{0.0\}$

**Then:**

1) $PS_1 = \langle P_1, W_C \rangle : p_4^1 = \langle \{c_4\}, \{a_2\} \rangle$
2) $M_1 \leftarrow M|m_4$
3) $M_R^{P_1} \leftarrow M_R^P|r_{m_4}$

Such a modification can be modelled using model-transformation $\Psi_4[G_n^P]$, as was illustrated in Section III-B4 (see Figure 8).

Suppose that we apply the second transformation (i.e., in response to removing policy action $a_2$ = "`AdjustMaxClients(-25)`" from policy $p_4^1$) onto the state-transition model of Figure 8(b) whose states are shown in Figure 8(c). Note that such a change relates to policy modification $\triangle_9[P_1]$; i.e.,

Policy Modification - $\triangle_9[P_1]$: *Removing an action from a policy in $P_1$.*

**Given:**

I) $PS_1 = \langle P_1, W_C \rangle$
II) $M_R^{P_1} = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$
III) $p_4^1 = \langle \{c_4\}, \{a_2\} \rangle$ such that $p_4^1 \in P_1$
IV) $a_2$ = `AdjustMaxClients(-25)`
V) $r_m \in M_R^P : r_m = \langle \alpha_m, \sigma_m \rangle$ :
- $\alpha_m = \langle$ `ID`, `metricName`, $\omega \rangle$
- $\sigma_m = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_k\}$

**Then:**

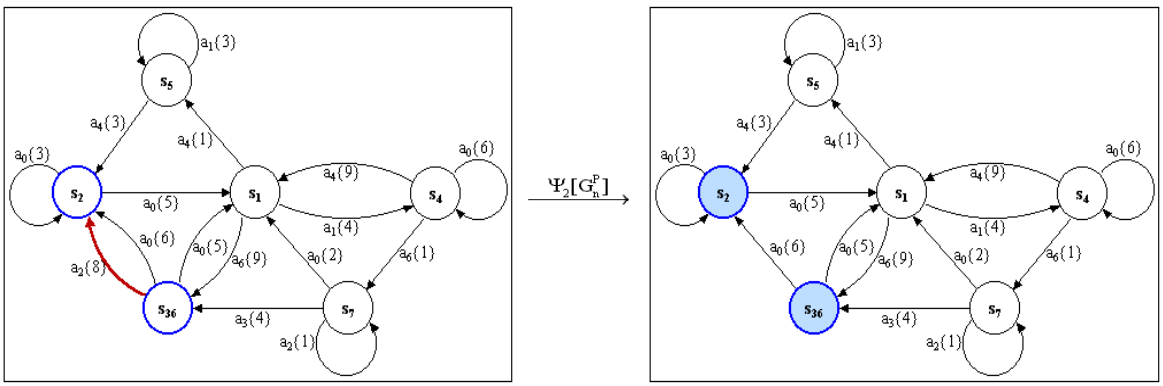
1) $PS_2 = \langle P_2, W_C \rangle : p_4^2 = \langle \{c_4\}, \{\emptyset\} \rangle$
2) $M_2 \leftarrow M_1$
3) $M_R^{P_2} \leftarrow M_R^{P_1}$

Such a modification can be modelled using model-transformation $\Psi_2[G_n^P]$. The following describes the steps involved in transforming the model and corresponds to the steps of model-transformation $\Psi_2[G_1^{P_1}]$:

1) For each state that has been encountered to date (see figure 8(c)), the agent must update the actions within the states to ensure that actions set $A'(s_i)$ contains only the actions of the policies that are violated when the system is in state $s_i$. Note that since policy $p_4^1$ is the only violated policy in states $s_2$ and $s_{36}$, $\triangle_9[P^1]$ would result in the removal of action $a_2$ from the actions sets $A(s_2)$ and $A(s_{36})$. While policy $p_4^1$ is also violated in states $s_1$ and $s_4$, $\triangle_9[P^1]$ would not affect such states. This is because action $a_2$ is also part of other violated policies in those states, such as policy $p_1$ (i.e., $a_2 \in p_1$) and policy $p_2$ (i.e., $a_2 \in p_2$) where $p_1 \in P'(s_1)$ and $p_2 \in P'(s_4)$. Consequently, both states $s_2$ and $s_{36}$ will change from "violation" to "non-violation" states.
2) The state-transition model must also be updated to ensure that transitions within states are consistent with the updated states actions sets. Note that the removal of action $a_2$ from state $s_{36}$ in the above stage would mean that transition $t(s_{36}, a_2\{8\}, s_2)$ (see Figure 9(a)) is no longer valid. As such, the transition will be removed as illustrated in Figure 9(b).
3) The final step of the adaptation is for the agent to perform backup updates so that any impact on the model as a result of removing transitions is propagated to the action-value estimates of the states actions.

Suppose that, we apply the third transformation (i.e., in response to removing policy condition $c_4$ ="`MEMORY:utilization > 40.0`" from policy $p_4^2$) onto the state-transition model of Figure 9(b), whose states are shown in Figure 9(c). Note that such a change relates to policy modification $\triangle_4[P_2]$; i.e.,

Policy Modification - $\triangle_4[P]$: *Removing a policy condition resulting in a decrease in the number of metrics.*




(a) Original model: $G_n^P = \langle S, T \rangle$

(b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R_{m_j}^k)$ | | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_3$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_1$ | 0 | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | $a_1$ | `AdjustMaxClients(+25)` | 4 | $s_4$ |
| | | | | | | $a_2$ | `AdjustMaxClients(-25)` | | |
| | | | | | | $a_4$ | `AdjustMaxKeepAliveRequests(-30)` | 1 | $s_5$ |
| | | | | | | $a_6$ | `AdjustMaxBandwidth(-128)` | 7;2 | $s_3$;$s_6$ |
| $s_2$ | 0 | 100 | 0 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_1$;$s_2$ |
| $s_{36}$ | 50 | 100 | 0 | 100 | 100 | $a_0$ | $\gamma$-action | 5;6 | $s_1$;$s_2$ |
| $s_4$ | 0 | 100 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | | $a_1$ | `AdjustMaxClients(+25)` | | |
| | | | | | | $a_2$ | `AdjustMaxClients(-25)` | | |
| | | | | | | $a_4$ | `AdjustMaxKeepAliveRequests(-30)` | 9 | $s_1$ |
| | | | | | | $a_6$ | `AdjustMaxBandwidth(-128)` | 1 | $s_7$ |
| $s_5$ | 0 | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | | $a_1$ | `AdjustMaxClients(+25)` | 3 | $s_5$ |
| | | | | | | $a_2$ | `AdjustMaxClients(-25)` | | |
| | | | | | | $a_4$ | `AdjustMaxKeepAliveRequests(-30)` | 3 | $s_2$ |
| | | | | | | $a_6$ | `AdjustMaxBandwidth(-128)` | | |
| $s_7$ | 100 | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_1$ |
| | | | | | | $a_2$ | `AdjustMaxClients(-25)` | 1 | $s_7$ |
| | | | | | | $a_3$ | `AdjustMaxKeepAliveRequests(+30)` | 4 | $s_3$ |

(c) System states of the new model.

Fig. 9. Adapting state-transition model of Figure 8 to $\triangle_9[P_1]$; i.e., removing action $a_2$ ="`AdjustMaxClients(-25)`" from policy $p_4^1$.

**Given:**

I) $PS_2 = \langle P_2, W_C \rangle$

II) $M_R^{P_2} = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$

III) $p_4^2 = \langle \{c_4\}, \{\emptyset\} \rangle$ such that $p_4^2 \in P^2$ where $\forall\ p_i \in P_2 | p_4^2,\ c_4 \notin p_i$

IV) $c_4 =$ "`MEMORY:utilization > 40.0`" : $m_3 =$ "`MEMORY:utilization`" and where $m_3 \in M : \forall\ m_i \in M_2 | m_3,\ m_i \neq m_3$

V) $r_{m_3} \in M_R^{P_2} : r_{m_3} = \langle \alpha_{m_3}, \sigma_{m_3} \rangle$ :

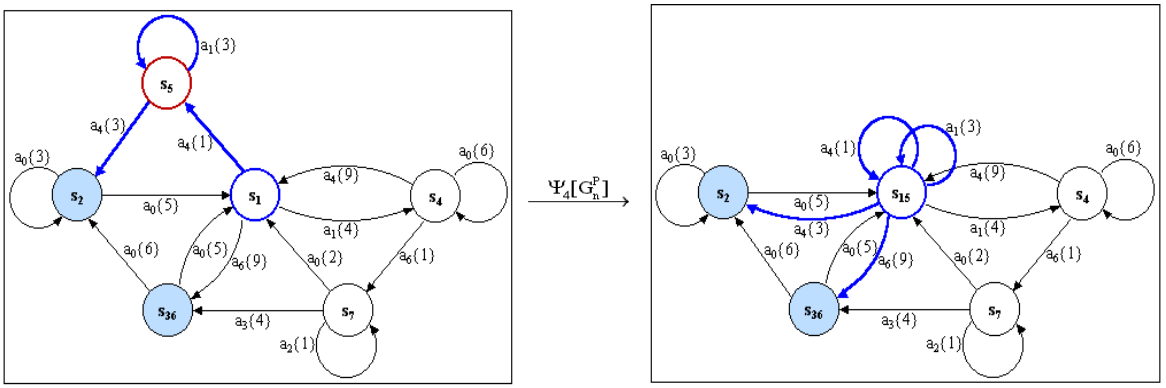
- $\alpha_{m_3} = \langle 3, m_3, \omega \rangle$
- $\sigma_{m_3} = \{40.0\}$

**Then:**

1) $PS_3 = \langle P_3, W_C \rangle : p_4^3 = \langle \{\emptyset\}, \{\emptyset\} \rangle$
2) $M_3 \leftarrow M_2 | m_3$
3) $M_R^{P_3} \leftarrow M_R^{P_2} | r_{m_3}$

Such a modification can be modelled using model-transformation $\Psi_4[G_n^P]$. The following describes the steps involved in transforming the model and corresponds to the steps of model-transformation $\Psi_4[G_2^P]$:

1) For each state that has been encountered to date (see Table 9(c)), the agent must first remove all the information associated with metric $m_3$ (see step 1(a)). This would result in a system's model with six states each with four metrics; i.e., $M(s_i) = \{m_1, m_2, m_5, m_6\}$. The agent must also update the states actions sets of each state to ensure that actions set $A'(s_i)$ contains only the actions of the policies that are violated when the system is in state $s_i$ (see Step 1(b)). Since $\triangle_4[P_2]$ is the third step in the sequence of changes to the policy $p_4$ (which, at this point, would have no actions associated with it), such a policy modification would result in no changes to the composition of the states actions.
2) The state-transition model must also be updated to ensure that the actions within states transitions are consistent with the updated states actions sets. Since the previous stage did not result in the deletion of states actions, no transition will be affected.
3) The agent must then determine whether any states should be merged together if, in fact, the modifications above may have resulted in two or more identical




(a) Original model: $G_n^P = \langle S, T \rangle$

(b) New model: $G_{n+1}^{P'} = \langle S', T' \rangle$

| | $f(R_{m_j}^k)$ | | | | $A(s_i)$ | | $t(s_i, a_l, s_*)$ | |
|---|---|---|---|---|---|---|---|---|
| $s_i$ | $m_1$ | $m_2$ | $m_5$ | $m_6$ | $a_l$ | State action | $\lambda$ | $s_*$ |
| $s_{15}$ | 0 | 0 | 0 | 0 | $a_0$ | $\gamma$-action | | |
| | | | | | $a_1$ | AdjustMaxClients(+25) | 3;4 | $s_{15}$;$s_4$ |
| | | | | | $a_2$ | AdjustMaxClients(-25) | | |
| | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 1;3 | $s_{15}$; $s_2$ |
| | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 9 | $s_{36}$ |
| $s_2$ | 0 | 100 | 0 | 100 | $a_0$ | $\gamma$-action | 5;3 | $s_{15}$;$s_2$ |
| $s_{36}$ | 0 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 5;6 | $s_{15}$;$s_2$ |
| $s_4$ | 0 | 100 | 0 | 0 | $a_0$ | $\gamma$-action | 6 | $s_4$ |
| | | | | | $a_1$ | AdjustMaxClients(+25) | | |
| | | | | | $a_4$ | AdjustMaxKeepAliveRequests(-30) | 9 | $s_{15}$ |
| | | | | | $a_6$ | AdjustMaxBandwidth(-128) | 1 | $s_7$ |
| $s_7$ | 100 | 100 | 100 | 100 | $a_0$ | $\gamma$-action | 2 | $s_{15}$ |
| | | | | | $a_2$ | AdjustMaxClients(-25) | 1 | $s_7$ |
| | | | | | $a_3$ | AdjustMaxKeepAliveRequests(+30) | 4 | $s_{36}$ |

(c) System states of the new model.

Fig. 10. Adapting state-transition model of Figure 9 to $\triangle_4[P_2]$; i.e., removing condition $c_4$ ="MEMORY:utilization > 40.0" from policy $p_4^2$.

states. Note that the deletion of metric $m_3$ would mean states $s_1$ and $s_5$ are identical; i.e., they have the same region value assignment (i.e., $f(R_{m_i}^l)$) for each identical metric. Thus, the two states would be merged together into state $s_{15}$ as illustrated in Figure 10. This would also involve merging the states transitions (see Step 3(a)) such that, transitions $t(s_1, a_4\{1\}, s_5)$, $t(s_5, a_1\{3\}, s_5)$, and $t(s_5, a_4\{3\}, s_2)$ (see Figure 10(a)) become transitions $t(s_{15}, a_4\{1\}, s_{15})$, $t(s_{15}, a_1\{3\}, s_{15})$, and $t(s_{15}, a_4\{3\}, s_2)$, respectively (see Figure 10(b)).

4) The final step of the adaptation is for the agent to perform backup updates so that any impact on the model as a result of removing transitions and/or merging states is propagated to the action-value estimates of the states actions.

Finally, we apply the fourth transformation (i.e., in response to removing policy $p_4^3$) onto the state-transition model of Figure 10(b), whose states are shown in Figure 10(c). Note that such a change relates to policy modification $\triangle_y[P_3]$; i.e.,

Policy Modification - $\triangle_y[P_3]$: *Removing a policy without conditions or actions.*

**Given:**

I) $PS_3 = \langle P_3, W_C \rangle$

II) $M_R^{P_3} = \{r_1, r_2, \ldots, r_l\} : r_i = \langle \alpha_i, \sigma_i \rangle$

III) $p_4^3 = \langle \emptyset, \emptyset \rangle$ such that $p_4^3 \in P_3$

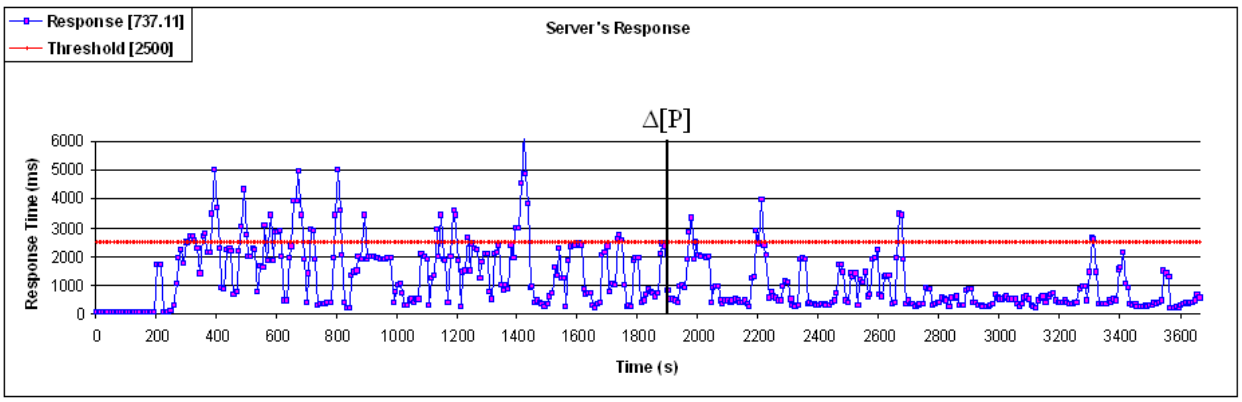
**Then:**

1) $PS' = \langle P', W_C \rangle : P' \leftarrow P_3 | p_4^3$
2) $M' \leftarrow M_3$
3) $M_R^{P'} \leftarrow M_R^{P_3}$

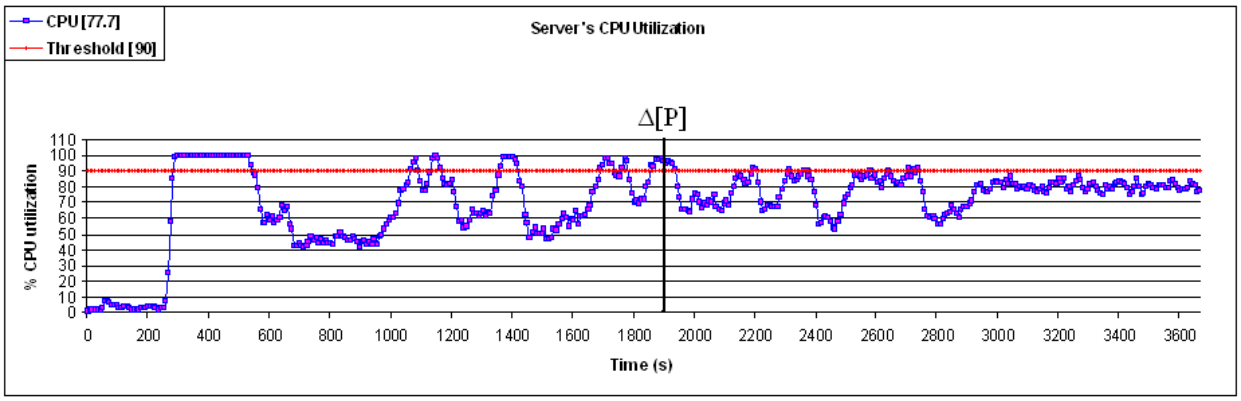
Since $\triangle_y[P_3]$ has no impact on the state-transition model, removing policy $p_4^3$ from Figure 1 would result in no changes to the state-transition model of Figure 10. As such, the removal of policy $p_4$ can be modelled as a sequence of transformations on the original state-transition model involving: $\triangle_4[P] = P_1$, $\triangle_9[P_1] = P_2$, $\triangle_4[P_2] = P_3$, and $\triangle_y[P_3] = P'$.

## IV. Experience

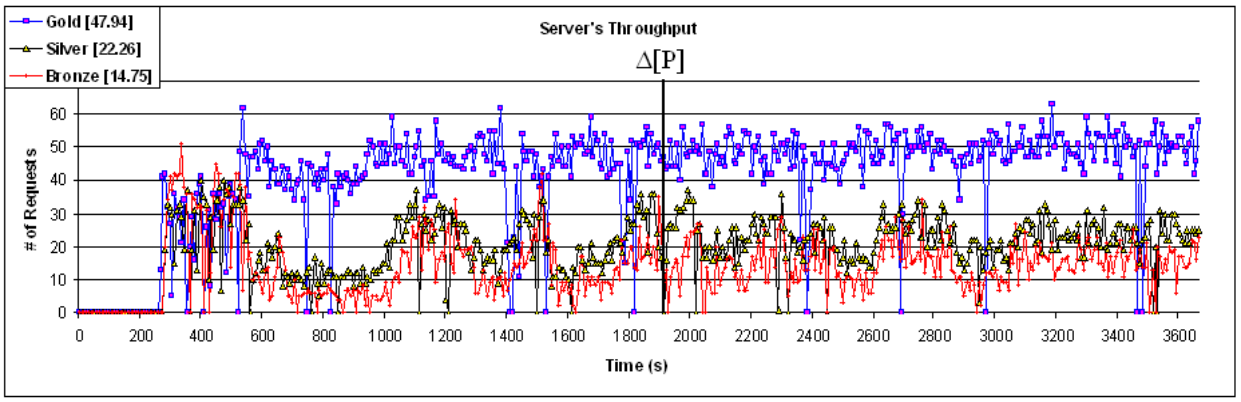
In this section, we present experiments illustrating the effectiveness of the adaptation mechanisms on the behavior of the server in response to run-time policy modifications. In particular, we compare results from two experimental settings based on how the agent responded to run-time policy modifications. The first experiment (*EXP-1*) investigated the behavior of the server where the learning agent discards everything it may have learned prior to $\triangle[P]$ and begins learning from scratch; i.e., $G_{n+1}^{P'} = \{\emptyset, \emptyset\}$. The second experiment (*EXP-2*) looked at the behavior of the server where the learning agent adapts the model of the environment dynamics to take into account the changes to an active set of policies, essentially




(a) Response Time




(b) CPU Utilization




(c) Throughput

Fig. 11. Adapting to run-time policy modification: $\triangle[P]$ involves replacing "mixed" expectation policies with "simple" expectation policies

reusing some of the learned information; i.e., $\Psi : G_n^P \rightarrow G_{n+1}^{P'}$ (see Figure 3).

### A. Testbed Environment

Our testbed consisted of a collection of networked workstations: an administrative workstation to run the experiments; a Linux workstation with a 2.0 GHz processor and 2.0 Gb of memory, which hosted the Apache Web Server along with the PHP module and the MySQL database server; and three workstations used for generating the server's workload, which consisted of clients requests associated with gold, silver, and bronze service classes.

To emulate the stochastic behavior of users, an Apache benchmark tool called JMeter [12] was used. The tool provides support for concurrent sampling of requests through a multi-threaded framework. It provides the ability to specify the client's think time as well as the number of concurrent - independent - *keep-alive* connections. In the experiments reported here, these values were set to ensure that the server was under overload conditions (i.e., saturated) for the entire duration of the experiments. In our current implementation, the tool has been configured to traverse a Web graph of an actual Web site: in our case, phpBB [13]. In the experiments reported in this section, only *read-only* database requests were

considered.

In order to support service differentiation, a Linux Traffic Controller (TC) was used to configure the bandwidth associated with the gold, silver, and bronze service classes. The service classes bandwidth was assigned proportionately according to the ratio 85:10:5. Bandwidth sharing was also permitted. The tuning parameter `MaxBandwidth` is what determines how much bandwidth is assigned to each service class. It corresponds to the physical capacity (in kbps) of the network connection to the workstation hosting the servers. Thus, given that the first policy of Figure 1 has been violated and that it is no longer possible, for example, to adjust the parameters `MaxClients` and `MaxKeepAliveRequests`, then the last policy action (i.e., `AdjustMaxBandwidth(-128)`) would be executed. This action essentially reduces the total bandwidth by 128 kbps. The percentage of the new bandwidth is what is eventually assigned to the different service classes.

### B. Results

The results presented in this section aim at demonstrating the effectiveness of the adaptation mechanisms on the policy modifications involving changing policy sets. Two sets of policies were considered and included "simple" and "mixed" expectation policies. Simple expectation policies included policies describing possible sets of actions to be taken whenever a single objective is violated. Mixed expectation policies include policies consisting of both "simple" and "complex" policies, with complex expectation policies describing the actions of the system whenever several objectives are violated. $\triangle[P]$, in this case, involved replacing "mixed" expectation policies with "simple" expectation policies; i.e., disabling all the complex policies within the mixed expectation policies set. The performance comparisons in terms of the averages and magnitude of violations reported in this section focused specifically on the behavior of the server after $\triangle[P]$ occurred.

Figure 11 shows the behavior of the server when dealing with the above policy modification, which occurred at about 1,900 seconds after the start of the experiment. In terms of response time measurements (see Figure 11(a)), the quality of service specific to the violations as well as the stability of the measurements greatly improved after $\triangle[P]$ occurred. This is clearly visible in the graph where there are very few instances where the measurements exceeded the threshold. The same is also true when CPU utilization and throughput measurements are considered, as Figures 11(b) and 11(c), respectively, illustrate. Thus, the overall quality of service greatly improved as a result of the adaptation mechanisms.

The top graph of Figure 12 compares the average throughput measurements of identical service classes. *EXP-1* corresponded to the case where the learning agent discarded everything it had learned prior to $\triangle[P]$ and began learning from scratch, while *EXP-2* corresponded to the experiment where the agent adapted the model in response to the policy modification. Thus, results reported for *EXP-2* are essentially those of Figure 11, but whose averages only include the measurements after $\triangle[P]$ occurred. From these results, the server performed slightly better in *EXP-2* when compared to the results in *EXP-1*, particularly for the gold service class where more requests were served. For the other classes (i.e., silver and bronze), however, the performance gains were statistically insignificant since the measurements fell within the standard error, as is illustrated by the error bars associated with the mean throughput.

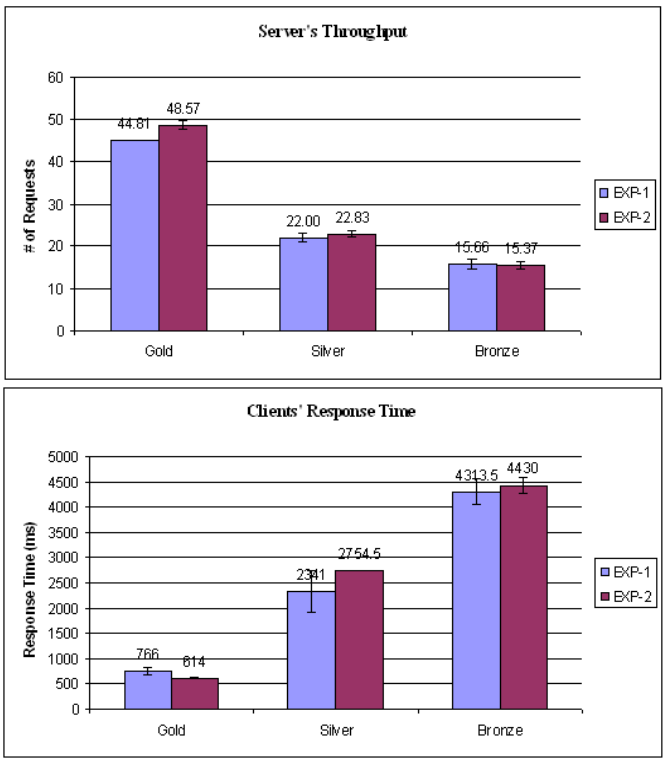
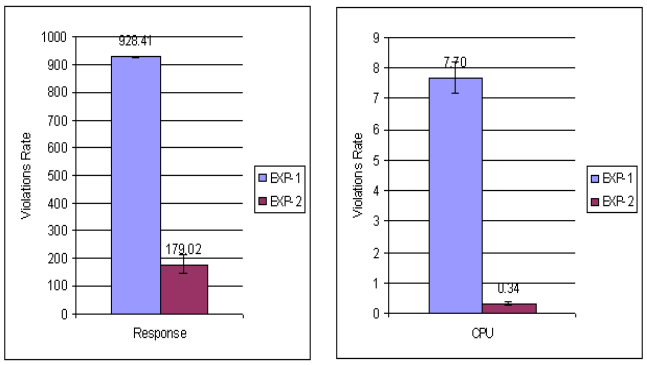



Fig. 12. Mean performance comparisons.

The same was also true when clients response time measurements across the service classes were compared as shown in the bottom graph of Figure 12. Thus, we can conclude that, model adaptation had a positive impact on the overall performance of the server when averages associated with the server's throughput and clients response times were considered. That is, the server performed slightly better in *EXP-2* when compared to the results in *EXP-1*, particularly for the gold service class where more requests were served. For the other classes (i.e., silver and bronze), however, the performance differences were statistically insignificant since the measurements fell within the standard error.

While the results above show slight improvement in the overall quality as a result of the adaptation mechanisms, mean performance is essentially an approximation of the quality of service performance of the system. The more accurate measure is the severity and the frequency with which performance objectives are violated, as summarized in Figure 13. Note that, *EXP-2* recorded at least an 80% reduction in the magnitude of response time violations (see Figure 13(a)) and a 96% reduction in the magnitude of CPU utilization violations (see Figure 13(b)). Thus, reducing the magnitude of violations such as instances in which clients requests wait on a server's queue for longer than 2500 ms, for example, is likely to




(a) Response Time (b) CPU Utilization

Fig. 13. Magnitude of violations performance comparisons.

improve the quality of service from a user's prospective - a measure mean performance tend to obscure. Furthermore, performance objective under these kinds of environments often need to account for not only customers' needs, but also system's constraints. For example, in order to control power consumption, an autonomic system may need to limit CPU utilization. Thus, the ability to achieve system-wide performance objectives, which may seem conflicting at times, under dynamically changing configuration conditions is what the experiments presented in this paper have demonstrated.

Furthermore, the results summarized in this section highlight one of the key benefits of utilizing reinforcement learning methodologies in policy-driven autonomic management, particularly when it comes to policy specification. That is, very "simple" expectation policies are sufficient to deliver significant performance gains. With learning enabled, more complex policies, particularly those spanning multiple objectives, could be learned instead of requiring systems administrators to manually encode into a single policy how systems should achieve multiple objectives. Thus, a general knowledge of the performance impact due to a change in the value of a parameter is sufficient to define simple but effective policies. This can significantly reduce the burden faced by human administrators, by having them specify simple policies, leaving it to autonomic systems to figure out how to achieve more complex objectives.

## V. Related Work

Policy-based management approaches have attracted significant interest within autonomic computing. The work in [14], for example, proposes a flexible policy-expression language called AGILE [15], which facilitates run-time policy configuration and adaptation of autonomic systems. AGILE in itself is both a policy expression language and a framework that facilitates dynamic composition of autonomic techniques including signal processing, trend analysis, and utility functions. IBM [6] has also been at the forefront in policy-driven autonomic management research. Their work in [16] proposes an Agent Building and Learning Environment (ABLE) capable of configuring new behaviors and capabilities to intelligent autonomic systems. In this approach, policies are specified using rule beans which make use of simple scripting text or Extensible Markup Language (XML) [17] to define simple assignments including if-then, if-then-else rules, when-do pattern match rules, etc.

Policy-based management has recently evolved to recognize the need to evaluate the correctness and effectiveness of policies specified by system administrators. For example, there has been some interest in policy conflicts detection and resolution (see, for example, [18]). Others have proposed frameworks that make use of *event calculus* (EC) [19] for policy verifications which adds the notion of time for first-order predicate logic. It provides a formalism for representation and reasoning about actions and their effects. This approach, however, requires that rules are specified to identify all possible inconsistencies from a set of policies, which is intractable for complex systems. This is particularly the case because, in most situations, some of the characteristics for identifying conflicts can only be detected at run-time.

The need to dynamically adapt the use of policies in autonomic management has also gained some traction. The work in [20], for example, proposes an adaptive policy-based framework that support dynamic policy configuration in response to changes within the managed environment. In their approach, policy adaptation describes the ability to modify network behavior by dynamically changing the policy parameters as well as selecting, enabling, or disabling policies at run-time. Reconfiguration events are used to trigger high-level control policies, which then determines which lower-level policies must be adapted to reconfigure the managed system. They use the Ponder deployment framework [21] to distribute policies to the different management components. They describe several ways in which the system's behavior could be adapted by changing the way policies are used [20]:

- *Dynamic Modification of Policy Parameters:* In this approach, the system can adapt policies by making changes to the policy parameters. This may involve computing new attribute values based, for example, on Service Level Agreements (SLAs), resource availability, etc. Related work on this type of adaptation in the use of policies can be found in [20], [22].
- *Enabling/Disabling Policies From a Set of Active Policies:* In this approach, the system can use run-time context to dynamically use the information provided by the policies, such as enabling/disabling a policy under certain circumstances or selecting actions based on context. Related work on adaptation through policy selection can be found in [20], [23].
- *Adaptation By Learning:* In this approach, the system can determine policy use through some learning mechanisms based on past experience in the use of policies. This type of adaptation is still in its very early stages, particularly in the filed of autonomic management, with the majority of the research work focusing on learning high-level policies. The most recent work can be found in [24], [25], [26], [27].

Our interest is on how learning approaches could facilitate dynamic use of policies within autonomic computing.

In our most recent work [7], we have explored the use of Reinforcement Learning methodologies in providing guidance to the autonomic system in terms of how to effectively use existing policies. One drawback to this and other approaches was that when policies change, the existing model is discarded and the system must learn a new model of the environment from scratch. Our experience, however, suggest that policy modifications are frequently "incremental" - e.g., a change to a threshold, the disabling of a single policy, etc. The ability to reuse a learned model, or part there of, has the potential benefit of significantly accelerating the learning process, as this paper has demonstrated.

## VI. Conclusion

The approach taken in this paper is only the first step in the broader goal of developing adaptive policy driven autonomic solutions involving multiple agents working together towards a set of common, and at times seemingly competing, objectives. This is often the case since performance objectives may need to account for both customers needs and systems constraints. The complexity of today's IT infrastructure, however, means that we can no longer depend on centralized approaches to performance management. It is becoming increasingly common to find multiple agents coexisting within a single computing environment. The work in [28], for example, proposes an approach for coordinating two independent autonomic managers: one designed to address power consumption objectives while another deals with performance objectives such as response time based on some learning mechanisms. As such, policy modifications directives could come not only from the users of the systems, but also from other autonomic managers. The fact that our approach to modelling the learning process is dependent only on the structure of the policies means that such changes to policies could be traced back to the model derived from the use of those policies. In this paper, we have elaborated on how a model learned from the use of an active set of policies could be adapted (i.e., reused) to cope with run-time policy modifications.

While the initial results are encouraging, we intend to validate these approaches with a comprehensive prototype evaluating the impact of the different kinds of policy modifications on the effectiveness of the adaptation transformations. We are also working on formulating formal proofs that show that, indeed if $P$ is changed to $P'$ and $G_n^P$ is transformed to $G_{n+1}^{P'}$, then $G_{n+1}^{P'}$ is a model of $P'$.

## References

[1] N. Damianou, N. Dulay, E. C. Lupu, and M. S. Sloman, "Ponder: A Language for Specifying Security and Management Policies for Distributed Systems: The Language Specification," Imperial College, London, England, Version 2.2, 2000.

[2] (2009, December) The Internet Engineering Task Force (IETF). [Online]. Available: http://www.ietf.org/

[3] (2009, December) The Distributed Management Task Force (DMTF). [Online]. Available: http://www.dmtf.org/

[4] (2009, December) The Object Management Group (OMG). [Online]. Available: http://www.omg.org/

[5] (2009, December) Policy Research Group. Imperial College. [Online]. Available: http://www-dse.doc.ic.ac.uk/Research/policies/ponder.shtml

[6] (2009, December) Autonomic Computing Research. IBM. [Online]. Available: http://www.research.ibm.com/autonomic/

[7] R. M. Bahati and M. A. Bauer, "Modelling Reinforcement Learning in Policy-driven Autonomic Management," in *IARIA International Journal On Advances in Intelligent Systems*, vol. 1, no. 1, December 2008.

[8] R. M. Bahati and M. A. Bauer, "Adapting to Run-time Changes in Policies Driving Autonomic Management," in *International Conference on Autonomic and Autonomous Systems (ICAS'08)*, Guadeloupe, March 2008, pp. 88–93.

[9] R. M. Bahati, M. A. Bauer, C. Ahn, O. K. Baek, and E. M. Vieira, "Using Policies to Drive Autonomic Management," in *WoWMoM-2006 Workshop on Autonomic Communications and Computing (ACC'06)*, Buffalo, NY, USA, June 2006, pp. 475–479.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: an Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[11] C. Watkins, "Learning from Delayed Rewards," Ph.D. dissertation, Cambridge University, Cambridge, UK, 1989.

[12] (2009, December) Apache JMeter. The Apache Software Foundation. [Online]. Available: http://jakarta.apache.org/jmeter/

[13] (2009, December) PHP Bulletin Board. [Online]. Available: http://www.phpbb.com/

[14] R. J. Anthony, "Policy-centric Integration and Dynamic Composition of Autonomic Computing Techniques," in *International Conference on Autonomic Computing (ICAC'07)*, Jacksonville, FL, USA, June 2007.

[15] (2009, December) AGILE. [Online]. Available: http://www.policyautonomics.net/

[16] J. P. Bigus, D. A. Schlosnagle, J. R. Pilgrim, W. N. M. III, and Y. Diao, "ABLE: A toolkit for Building Multiagent Autonomic Systems," in *IBM Systems Journal*, vol. 41, no. 3, September 2002.

[17] (2009, December) Extensible Markup Language (XML). [Online]. Available: http://www.w3.org/XML/

[18] N. Dunlop, J. Indulska, and K. Raymond, "Dynamic Conflict Detection in Policy-Based Management Systems," in *International Enterprise Distributed Object Computing Conference (EDOC'02)*, Washington, DC, USA, 2002, pp. 15–26.

[19] R. Kowalski and M. Sergot, "A Logic-based Calculus of Events," in *New Generation Computing*, vol. 4, no. 1, Tokyo, Japan, 1986, pp. 67–95.

[20] L. Lymberopoulos, E. C. Lupu, and M. S. Sloman, "An Adaptive Policy-Based Framework for Network Services Management," in *Journal of Networks and Systems Management*, vol. 11, no. 3, 2003, pp. 277–303.

[21] N. Dulay, E. C. Lupu, M. S. Sloman, and N. Damianou, "A Policy Deployment Model for the Ponder Language," in *IEEE/IFIP International Symposium on Integrated Network*, Seattle, CA, USA, May 2001, pp. 529–544.

[22] K. Yoshihara, M. Isomura, and H. Horiuchi, "Distributed Policy-based Management Enabling Policy Adaptation on Monitoring using Active Network Technology," in *IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM'01)*, Nancy, France, October 2001.

[23] R. M. Bahati, M. A. Bauer, and E. M. Vieira, "Adaptation Strategies in Policy-Driven Autonomic Management," in *International Conference on Autonomic and Autonomous Systems (ICAS'07)*, July 2007.

[24] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, "A Hybrid Reinforcement Learning Approach to Autonomic Resource Allocation," in *International Conference on Autonomic Computing (ICAC'06)*, Dublin, Ireland, June 2006, pp. 65–73.

[25] G. Tesauro, "Online Resource Allocation Using Decompositional Reinforcement Learning," in *National Conference on Artificial Intelligence (AAAI'05)*, Pittsburgh, PA, USA, 2005, pp. 886–891.

[26] D. Vengerov and N. Iakovlev, "A Reinforcement Learning Framework for Dynamic Resource Allocation: First Results," in *International Conference on Autonomic Computing (ICAC'05)*, Seattle, WA, USA, January 2005, pp. 339–340.

[27] P. Vienne and J.-L. Sourrouille, "A Middleware for Autonomic QoS Management based on Learning," in *International Conference on Sofware Engineering and Middleware*, Lisbon, Portugal, September 2005, pp. 1–8.

[28] J. O. Kephart, H. Chan, R. Das, D. W. Levine, G. Tesauro, F. Rawson, and C. Lefurgy, "Coordinating Multiple Autonomic Managers to Achieve Specified Power-performance Tradeoffs," in *International Conference on Autonomic Computing (ICAC'07)*, Jacksonville, FL, USA, June 2007.


www.iariajournals.org

**International Journal On Advances in Intelligent Systems**
ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2679

**International Journal On Advances in Internet Technology**
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING
issn: 1942-2652

**International Journal On Advances in Life Sciences**
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO
issn: 1942-2660

**International Journal On Advances in Networks and Services**
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION
issn: 1942-2644

**International Journal On Advances in Security**
ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS
issn: 1942-2636

**International Journal On Advances in Software**
ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2628

**International Journal On Advances in Systems and Measurements**
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL
issn: 1942-261x

**International Journal On Advances in Telecommunications**
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA
issn: 1942-2601