On Finding Optimal Registrations in Presence of Overlapping Registration Areas

Georgios Varsamopoulos and Sandeep K. S. Gupta
Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287

{Georgios.Varsamopoulos,Sandeep.Gupta}@asu.edu

ABSTRACT

Personal Communication Services (PCS) standards such as IS-41 and GSM use a location management scheme which is based on registration areas (RA). Overlapping of registration areas has been proposed to reduce the overhead of location updates in such systems [4]. Overlapping RAs form *regions* in which a mobile has a choice in selecting which RA to register to. The choice of current RA has implication on the number of registrations a mobile may require in the future. In this paper we investigate the problem of finding the best (or optimal) registration for a user in a cellular PCS environment with *overlapping Registration Areas* (RAs). We study two versions of the problem: (i) the *deterministic* version, where the entire trajectory of the mobile is known a-priori, and (ii) the *stochastic* version, where the trajectory of the mobile is not known a-priori but the system has the knowledge of mobility pattern of the mobile. The mobility pattern is modeled as a stochastic random walk across regions. In the deterministic case the method presented computes an optimal solution to the registration problem. In the stochastic case we present a method which determines the registration which minimizes the expected number of registrations by looking ahead at the probable paths which may be traversed by the mobile in future. The paper also proposes a *preemptive* and a *non-preemptive* version of the stochastic method which incur different numbers of *hard* and *soft* registration operations.

Keywords: location management, mobile communication networks, cellular networks, optimization, combinatorics.

1. INTRODUCTION

Personal Communication Services (PCS) allow mobile users with wireless terminals to receive calls (messages) irrespective of their location. PCS networks have a cellular architecture: the geographical area is divided into cells with one base station (BS) per cell. The mobile user's portable terminals communicate via wireless with fixed radio ports in the base station. Delivering calls to the mobile terminals requires that the current location (point of attachment) of the mobile terminal be known in order for the network to route the calls to the mobile terminal. The task of tracking the location of mobile terminals is know as *location management* which involves two operations: location update (or registration) and search (or paging). PCS standards such as IS-41 [3] and GSM use a location management scheme which is based on registration areas (RA). The geographical area is divided into several registration areas, where each registration area consists of several cells. The system tracks a mobile terminal's registration area instead of its cell. Whenever a mobile terminal crosses from one registration area to another it informs its new location to the system. To setup a call to a mobile terminal, the system pages all the cells in the registration area to find the current cell of the mobile terminal. A database called *location register* (LR) is associated with each registration area to keep information about the mobiles currently registered in that registration area. Both IS-41 and GSM standards use registration areas along with a two level hierarchy of location registers. The hierarchy consists of home location registers (HLR) and visitor location registers (VLR's). Location information of a mobile node is kept at both its current location registrar, i.e. its VLR, and its permanent home location registrar. This facilitates locating non-local mobiles during call delivery: the home location registrar of the mobile unit is contacted to find its current location.

Current trends in the usage of PCS networks show that the number of subscribers is expected to increase exponentially for next several years. At the same time, service providers are reducing the size of a cell in order to provide better quality of wireless connections, to keep the average number of users in a cell small, and to increase the reuse of frequencies. These two trends combined imply a dramatic increase in registration operations in a network. One way to reduce the number of registrations in a PCS system is by the use of *overlapping* RAs [4]. Overlapping Registrations Areas, in which there are sections where a mobile may choose from a set of RAs to register with, is a means of making the servicing network able to handle diverse user patterns. The choice of current RA has implication on the number of registrations a mobile may require in the future. Since the network has the ability to make a selection out of available choices, the problem of best choice, i.e. the choice that minimizes overall number of registrations, is an interesting problem to study. Minimizing the number of registrations done by each mobile in the system is beneficial for both the mobile and the location management system.

Several schemes have been proposed for location management [1, 4, 5, 6, 2]. However, very little research has been done on the potentials of registration area overlapping. Ho and Akyildiz [4] pro-

DIAL M 2001 7/01 Rome, Italy

posed a dynamic hierarchical location management scheme in the IS-41 [3] paradigm, and introduced the concept of overlapping registration areas. Wang and Akyildiz [8] proposed a *boundary Registration Area* that overlaps on the boundaries of multi-tier systems, and showed how it can be used to improve QoS (reduce call loss) in such systems. Bejerano and Cidon[1] proposed an overlayed and overlapping RA organization of the coverage area, the communication cost between two units is linearly bound by the geographical (Euclidean) distance of their respective lowest-level *Location Registers*. A similar assumption is used in the last section in this paper. The RAs of one level have twice the radius of the RAs of the level below, which makes the system have a logarithmic number of layers with respect to the size of the coverage area. The scheme has very good average search and update time, but it has very bad worst-case times. In [7] we extended the idea of overlapping RAs proposed in [4] and introduced the concept of dynamically re-sizing RAs which adapt to changes in call and mobility traffic. The scheme is based on monitoring the aggregate mobility and call pattern of the users during each *reconfiguration period* and adapting to mobility and call pattern by either expanding or shrinking registration areas at the end of each reconfiguration period. However, expanding a registration area increases the cost of call delivery (location search operation). Hence, the scheme in [7] tries to dynamically find the optimal configuration where the difference between the gains due to larger registration areas (reduction in update cost) and loss due to the increase in location search cost is maximized.

In this paper we investigate the problem of finding the best (or optimal) registration for a user in a cellular PCS environment with *overlapping Registration Areas* (RAs). We study two versions of the problem: (i) the *deterministic* version, where the entire trajectory of the mobile is known a-priori, and (ii) the *stochastic* version, where the trajectory of the mobile is not known a-priori but the system has the knowledge of mobility pattern of the mobile. The mobility pattern is modeled as a stochastic random walk across regions. In the deterministic case the method presented computes an optimal solution to the registration problem. In the deterministic case, the problem of finding the optimal registration for a given network and a given trajectory of the mobile can be formulated as a shortest path problem. We present this formulation and some properties of the solution. The solution for the deterministic case is used to motivate the solution for the stochastic case which is a much harder problem. The algorithm presented in the stochastic case tries to make a good guess by taking into account the probabilities of each possible finite path that can be followed. In the stochastic case we present a method which determines the registration which minimizes the metric *expected number of minimum registrations (Expct)* by looking ahead at the probable paths which may be traversed by the mobile in future. The paper also proposes a *preemptive* and a *non-preemptive* version of the stochastic method which incur different numbers of *hard* and *soft* registration operations.

2. SYSTEM MODEL

The system model assumed in this paper consists of an abstract cellular environment that follows the VLR/HLR paradigm: we assume a geographical area which is covered by a cellular network, similar to the cellular technology of mobile telephony. Although in real cellular networks there are parts of the geographical area which are not covered, we assume that the entire area is completely covered by cells. Holes in the coverage area are usually called *blind spots*; our network is assumed not to have any blind spots.

Cells are clustered into Registration Areas, each of which has a Location Register. RAs are assumed to overlap, which means that any cell may belong to multiple RAs. In a system of overlapping RAs, the coverage area of an RA is increased, compared to a non-overlapping RA, which means that, depending on the actual move pattern, the user may remain associated with an RA for a longer duration. At the same time the average number of users in an RA remains the same, as the total number of users doesn't change compared to a system with non-overlapping RAs. Also, overlapping gives the ability to perform a registration while a user is not performing a hand-off (called *soft* registrations), thus not requiring a time-critical registration (or *hard* registration) during an inter-RA hand-off.

The registration areas are considered to cover a contiguous geographical area, which means that any mobile can move from any cell of the registration area to any other cell of the registration by traversing only through cells that are clustered to the RA in question. Although the shape of the RAs is unrestricted, in our illustrations we depict them as circles. The number of RAs available in a region is maintained by the system; every BS knows what registration areas are available in its cell. We will see that when a mobile enters a region where new RAs are available and/or other RAs are no longer available, the system may initiate a registration operation for the mobile. Figure 1a shows an example of the assumed model in it's simplified version: RAs (dotted circles) overlap among themselves thus creating regions (intersections). In the next two sections, communication cost within a registration area is considered to be the same among all cells; in the last section, communication cost within a registration area varies among cells.

3. DETERMINISTIC OPTIMAL REGISTRATION PROBLEM

3.1 Problem description

Consider a mobile unit that traverses a coverage area like in Figure 1a. The mobile has to perform some registrations, since it doesn't stay under one registration area. Every time the mobile leaves a registration area, it has to register with another registration area that is available in the region it is currently in. The mobile starts at region "a" and ends in region "i". One possible registration sequence would be (a,b,c) in R_1 (i.e. mobile remains registered with RA R_1 during its trajectory through regions a, b, and c), then (d,e,f) in R_6, (g,h) in R_2 and lastly (i) in R_3. Another possible registration sequence is (a,b,c) in R_1, (d,e,f,g) in R_2, and (h,i) in R_3. Notice that the number of registrations (including the initial one) is 4 in the first example and 3 in the second example. Is there a better, i.e. shorter, registration sequence than that? For the given example the shortest registration sequence that can exist is of length 3. Notice that we are implicitly assuming that the mobile always stays online. Otherwise, the mobile could turn off before exiting R_1, travel to R_3 and switch back on and register to R_3, thus yielding a registration sequence of length 2. We do not, however, consider this case in this paper.

In order to formalize the problem, we use the following definitions. A *Service Area* of a cellular network is given as a finite set C of cells $C = \{c_1, c_2, \ldots\}$. A *Registration Area* is a finite set R of cells: $R = \{c_i, c_j, \ldots\}$, $R \subset C$. Actually C is partitioned into Registration Areas, such that $C = \cup_i R_i$. RAs are geographically contiguous. We define the *availability vector* $A(c)$ for a cell c as an

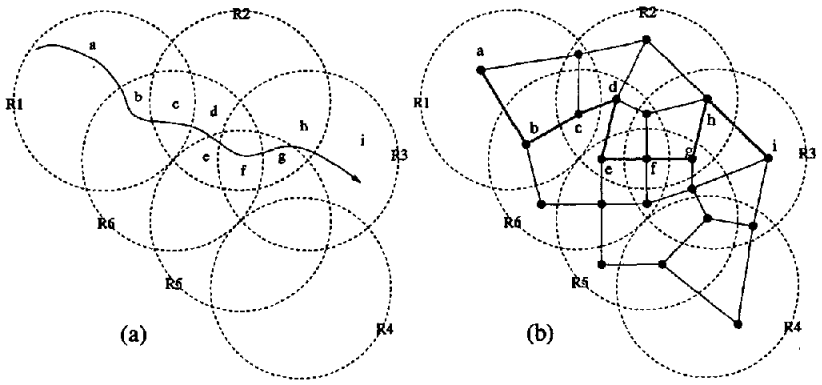

Figure 1: (a) Example of the system model and a user traversing the regions formed by the overlapping RAs. (b) The graph G_C that is built based on the system model. The $R1 - R5$ are Registration Areas; $a - i$ are the regions that are traversed by the user.

ordered set:

$$A(c) \stackrel{\text{def}}{=} \{R |\ R \in A(c) \Leftrightarrow c \in R\}.$$

For systems with non-overlapping RAs, $|A(c)| = 1$ for all cells c in the system. When RAs overlap, $\forall c |A(c)| \geq 1$. We define a *region* $g \subset C$ such that:

1. $\forall (c_i, c_j) \in g \times g,\ A(c_i) = A(c_j)$ (basic property of a region), and
2. $\forall (c_i, c_j) \in g \times (C - g),\ A(c_i) \neq A(c_j)$ (maximality property of a region).

Based on the basic property of a region above, we can extend the availability notion to a region as:

$$A(g) \stackrel{\text{def}}{=} A(c), \text{ for any } c \in g.$$

A user that moves in the coverage area of the network, the move can be denoted as a sequence of regions $p = g_1 g_2 \ldots g_k$, assuming that the user's move starts in region g_1 and ends in region g_k. While the user moves from region to region, availability of RAs changes. The user has to be registered to a Registration Area at all times, which means that the user has to have an RA selected out of the available RAs in the region. A user registers at most once per region, that is, once the user selects an RA and registers, the user doesn't re-register to another RA as long as it is in the same region. The user may continue to be registered to the previously registered RA, if the RA is available in the current region, thus no registration action is performed, or may switch to a different RA among the available ones, in this case a registration action is performed. If the user moves to a region where the last RA is not available, then the user has to choose among the available ones and perform a registration action to the chosen RA. We formalize, the problem of finding optimal registration under the above registration model as follows. Consider that the region path p is partitioned into disjoint subpaths $p_1, p_2, \ldots, p_m$, $(p_i = g_{i_1} g_{i_2} \ldots g_{i_j})$, such that $p_1 p_2 \ldots p_m = p$. We say that a registration sequence $S = R_1 R_2 ... R_m$ is *consistent* with path p *iff* there exists a partitioning $p_1 p_2 \ldots p_m$ of path p such that $\forall g_i \in p_j\ \ R_j \in A(g_i)$. Then, the **deterministic optimal registration problem** is: *given a sequence of regions $g_1 g_2 \ldots g_m$ and their availability vectors $A(g_1), A(g_2), \ldots, A(g_m)$, find a consistent registration sequence S of minimum length.* Note that the optimal registration is the first registration in S.

3.2 Algorithm M

The problem, as described above, can be translated into a shortest path problem. First, we define the *expanded form* of a registration sequence $S = R_1 R_2 ... R_m$ consistent to a path $p = g_1 g_2 \ldots g_k$ as a registration sequence: $Expand(S = \underbrace{R_1 R_2 ... R_m}_{m}) \stackrel{\text{def}}{=} S' = \underbrace{R_1 \ldots R_1 R_2 \ldots R_2 \ldots R_m}_{k} \ |\ S'(i) \in A(g_i),\ i = 1, 2, \ldots, k.$

Similarly, we define the *condensed form* of an expanded form, which is the registration sequence as first defined:

$Condense(S = \underbrace{R_1 \ldots R_1 R_2 \ldots R_2 \ldots R_m}_{k}) \stackrel{\text{def}}{=} S^* = \underbrace{R_1 R_2 ... R_m}_{m}$,

$S(i) \in A(g_i),\ i = 1, 2, \ldots, k$. We will denote $Condense(S)$ as S^*. Note that for a condensed form there may be multiple expanded forms, but the converse is not true. The utility of the expanded form is that it shows exactly what RA the user is registered to in each region of the trajectory; the utility of the condensed form is that it quickly shows the "length" of the registration sequence. In this subsection we will show a way of finding a minimal registration sequence in the expanded form.

The RA coverage layout is given as a graph $G_C(V_C, E_C)$ in which vertices represent regions and edges denote adjacency between two regions. A graph for the example layout in Figure 1a is given in Figure 1b. We also define $\mathcal{A}$ as an ordered set of availability vectors for a given trajectory T:

$$T = (g_1, g_2, ..., g_m) \Leftrightarrow \mathcal{A}_T \stackrel{\text{def}}{=} \{A(g_1), A(g_2), \ldots, A(g_m)\}$$

For the example trajectory, $\mathcal{A}_{\text{abcdefghi}} = ((R_1), (R_1, R_6), (R_1, R_2, R_6), (R_2, R_6), (R_2, R_5, R_6), (R_2, R_3, R_5, R_6), (R_2, R_3, R_5), (R_2, R_3), (R_3))$.

A user trajectory T is given as a sequence $g_1 g_2 \ldots g_m$ of regions which is a path $(g_1, g_2, \ldots, g_m)$ in graph G_C. Using $\mathcal{A}_T$ we build a directed acyclic graph (dag) $G_M(V_M, E_M)$, in which a vertex $g_i \in V_M$ denotes a pair (g_i, r) such that $r \in A(g_i)$, and a directed edge $e = ((g_i, r_i), (g_j, r_j)) \in E_M$ iff $\exists\ k$ such that $T(k) = g_i$ and $T(k+1) = g_j$. The graph G_M has

- $|V_M| = \sum_{i=1}^{m} |A(g_i)|$ vertices, and
- $|E_M| \leq m \times \max^2\{A(g_1), A(g_2), \ldots, A(g_m)\})$ edges.

Edges exist between vertices that represent adjacent regions in the path T. We name the vertex (*initial region, initial RA*) as *source*

and the vertex (*final region, final RA*) as the *sink*. Conceptually, the dag G_M denotes the "states" of the mobile while traversing a given route, with the edges denoting "registration" actions. Any path from a source to the sink denotes a valid registration sequence for the given trajectory. Edges are weighted; zero weight is assigned to an edge if the RAs in the two connected vertices are the same, thus denoting no need for registration; a unit weight is assigned to an edge if the RAs in the two connected vertices are not the same, thus denoting the need for a registration. The dag G_M for the example of Figure 1 is given in Figure 2. If the final region has more than one RAs available, then we have multiple sinks; in such case, we obtain a new dag G'_M by augmenting G_M with a "new" sink vertex to which every sink vertex in G_M is connected with a 0-edge (see Figure 2). An optimal solution is obtained by finding a shortest path from the source to the sink in G'_M and then taking the sequence of RAs that appear on the vertices of the shortest path, ignoring the RAs of the source and the sink. For the example path *abcdefghi*, a shortest path in the graph G'_M (see Figure 2) is $((a,R_1),(a,R_1),(b,R_1),(c,R_1),(d,R_2),(e,R_2),(f,R_2),(g,R_2),(h,R_2),(i,R_3))$. The optimal registration sequence is $M(R_1,T) = R_1R_1R_1R_2R_2R_2R_2R_2R_3$ with $M^*(R_1,T) = R_1R_2R_3$.

Algorithm M, as shown in Appendix A, takes an initial registration r_o and a trajectory $T = g_1g_2\ldots g_m$. Based on the knowledge of the availability set $\mathcal{A}_T$, it builds the graph G_M and finds an optimal path, which is an expanded form of a minimal registration sequence. If there is no initial registration (i.e. if we want to decide on the initial registration too), then the source vertex is $(\mathbf{null}, g_1)$ that is connected to the vertices (r_i, g_1), $r_i \in A(g_1)$, of the graph RA. In such case, the optimal path is denoted as $M(T)$. We can view $M(T)$ as a *amnesic* version of $M(r_o,T)$ (i.e. the mobile forgets the RA with which it last registered).

PROPERTY 1. *If* $T = g_1g_2\ldots g_m$, *and if* $\alpha = max\{|A(g_1)|, |A(g_2)|,\ldots,|A(g_m)|\}$, *then the complexity of* $M(T)$ *is* $O(m\alpha^2)$.

If the user traverses through m regions, then constructing the graph takes $O(m\alpha^2)$ steps. Solving the shortest path problem on the graph G_M takes $O(m\alpha^2)$ steps, since finding a shortest path in a dag is linear to the number of edges.

There is a subtle difference between what $M(r,T), T = g_1g_2..\ldots g_m$, means for $r \notin A(g_1)$ and what it means for $r \in A(g_1)$. In the first case, every registration in $M^*(r,T)$ is required to be performed. In the second case, $M(r,T)$ may return a registration sequence S in which $S(1) = r$. This registration is a *redundant* registration because it doesn't have to be performed. This is due to the fact that algorithm M returns a registration sequence $S = r_1r_2\ldots r_m$ denoting that for each region g_i in T the user "has to be registered" (as opposed to "has to register") with RA r_j in S. Therefore, the notation $|M^*(r,T)|$ denotes the number of RAs that appear in the sequence $M^*(r,T)$, including the redundant registration. If we want to know how many actual registration operations have to be made then we need to ignore the redundant registration, if it appears:
For $T = g_1g_2\ldots g_m$, if $r \in A(g_1)$ then the actual number of reg -istrations to be made is $|M^*(r,T)| - 1$. In the $M(T)$ case (no initial RA is given), there are no redundant registrations, similarly to the case of $r \notin A(g_1)$. Actually, the next property gives an insight into the equivalence of $M(T)$ and $M(r,T)$ for $r \notin A(g_1)$:

PROPERTY 2. *For a given trajectory* $T = g_1g_2\ldots g_m$, $\forall\, r_o \notin A(g_1)$ $M(r_o,T) = M(T)$.

In the case of $r_o \notin A(g_1)$, algorithm $M(r_o,T)$ is going to create a source vertex (r_o,g_1) and connect it to every vertex (r,g_1) in G_M, $r \in A(g_1)$, with a weight of 1 on each edge; Similarly, in the case of no initial RA, algorithm $M(r_o,T)$ is going to create a source vertex (r_o,g_1) and connect it to every vertex (r,g_1) in G_M, $r \in A(g_1)$, with a weight of 1 on each edge. Hence $M(r_o,T) = M(T)$, $\forall\, r \notin A(g_1)$.

The following property and lemma show the relation between $M(T)$ and $M(r,T)$, $T = g_1g_2\ldots g_m$ for $r \in A(g_1)$.

PROPERTY 3. *If* $T = g_1g_2\ldots g_m$ *then* $\exists r \in A(g_1)$ *such that* $M(T) = M(r,T)$.

$M(T)$ will always return a registration sequence $S = r_1r_2...r_m$ where $r_1 \in A(g_1)$. Therefore $M(T) = M(r_1,T)$.

LEMMA 1. *For any trajectory* $T = g_og_1...g_m$:
$\forall r \in A(g_o)$ $|M^*(T)| \leq |M^*(r,T)| \leq |M^*(T)| + 1$
(note that $M^*(T) \stackrel{\text{def}}{=} Condense(M(T))$.*)*

PROOF. *C the first inequality is trivial, as there cannot be a shorter path than the shortest. For the second inequality, assume that* $M(T) = S = r_1r_2\ldots r_m$, *without loss of generality. Then we can construct a registration sequence* $S_x = r_xr_2\ldots r_m$, *where* r_x *is any RA in* $A(g_o)$. *Note that* S_x *is consistent with T and also that* $|S_x^*| \leq |M^*(T)| + 1$. $M(r_x,T)$, *being optimal, has to be shorter or equal to* S_x *in the condensed form. Thus we have* $|M^*(r_x,T)| \leq |S^*| \leq |M^*(T)| + 1$. □

4. STOCHASTIC OPTIMAL REGISTRATION PROBLEM

4.1 Problem description

We saw in the previous section a way to compute an optimal registration sequence for a given user trajectory. In most real world cases, the user's trajectory is not known precisely by the system. In this case, an option is to make decisions on registrations about future moves by only knowing the user's mobility pattern. We assume that the system has knowledge of the mobility pattern of the user; the mobility pattern is represented as a random walk model, which is a directed graph $G_R(V_R,E_R)$ derived from $G_C(V_C,E_C)$. Each vertex in V_R denotes regions; each edge not only denotes adjacency between regions (as in G_C) they also denote transitions between vertices (states of the random walk model). Therefore, edges are weighted; an edge's weight is the transition probability from the starting vertex to the ending vertex (that is, a transition probability between the two corresponding states in the random walk model). The sum of weights of outgoing edges on every vertex is 1.

An example random walk model is given in Figure 3. A service area is divided into four registration areas R_1, R_2, R_3, R_4. The RAs overlap with each other, thus yielding 10 regions. Based on the layout of the regions, we create a graph with a vertex for each intersection and an edge for each common border of any two regions. The transition probabilities for each direction of each edge is shown on the same figure. Every move from a vertex g_i to a vertex g_j (with $i,j \in \{1,...,10\}$), has a specific probability $p_{ij} = w(g_i,g_j)$, where the weight function w is defined as $w(u,v) =$ Prob[user will move to region v | user is in region u]. For a path $T = g_ig_jg_k\ldots$ in G_R we denote its probability as the product of the transition probabilities of the edges in the path T: $\text{Prob}(T) = w(g_i,g_j)w(g_j,g_k)\ldots$.

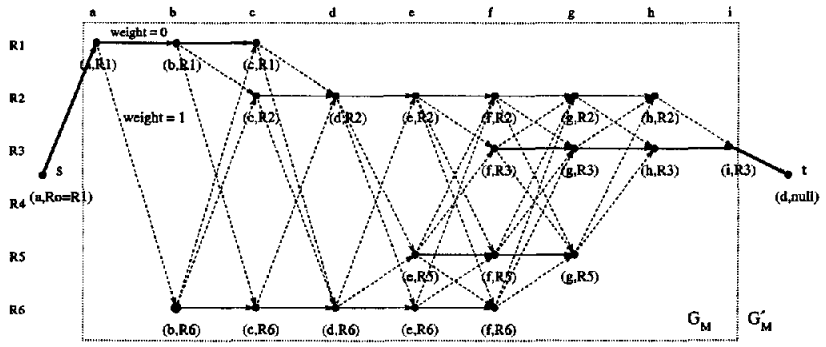

Figure 2: The original directed acyclic graph (dag) G_M and the augmented dag G'_M which is built by algorithm M. Note the source (s) and sink (t) vertices.

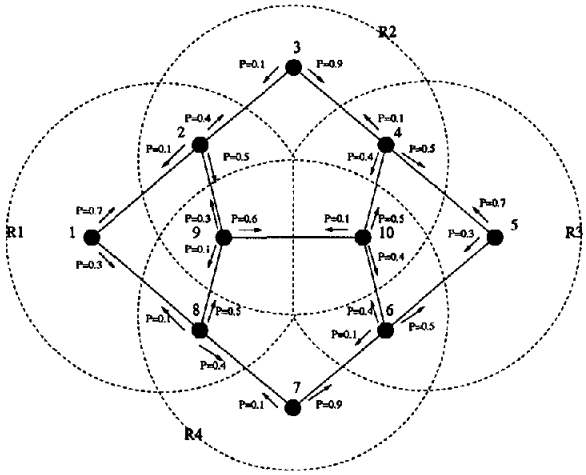

Figure 3: An example graph G_R that represents the random walk model assumed for the prediction algorithms in the stochastic case. We can see the 4 registration areas forming the 10 regions, each being a vertex in the random walk graph.

Consider a user that is in region g_o in the mobility model and last RA the user registered with is r_o. Let's assume for the moment that $r_o \notin A(g_o)$, therefore the user has to register to an RA in $A(g_o)$. We need to define a criterion for judging if a selection of an RA is "bad" or "good". The best criterion is whether the selected RA is part of an optimal registration sequence along the future path of the user, that is, if we knew the future path of the user, we could calculate a shortest path and chose the registration of the optimal path for the current state. Unfortunately, this metric cannot be used in an algorithm, because it uses information from the future. A registration that is part of a future path's optimal registration sequence is not necessarily part of any other future path's optimal registration sequence, as there may exist paths with totally different optimal registration sequences, and the user may follow any of the possible future paths. On the other hand, if the user is going to follow a path p_1 with a probability a and an alternative path p_2 with a probability $b < a$, it is reasonable to favor a registration area that is part of an optimal registration sequence for path p_1 over p_2. All in all, we need to make a selection of a registration area such that, based on the probabilities of future paths, is part of a "short" (if not shortest) registration sequence for the most probable future paths.

We define the *expected minimum number of registrations for RA r with look-ahead depth d starting at region g_o under model G_R* as the weighted sum of the lengths of the optimal registration sequence of every user path of length d starting at g_o into the probability that the user will take that path:

$$\text{Expct}(G_R, g_o, r, d) = \sum_{p \in \mathcal{P}(g_o, d)} \text{Prob}(p) |M^*(r, p)|$$

where $\mathcal{P}(g_o, d)$ is the set of all paths in G_R of length d that start at vertex g_o. The **stochastic optimal registration problem** is *given a mobility graph G_R, a starting vertex g_o in G_R and a look-ahead depth d, find an RA that yields minimal expected minimum number of registrations as expressed by Expct(G_R, g_o, r, d).*

4.2 Algorithm E

In this section we will present an online algorithm that computes the minimal Expct. We assume that we are given the last registration area and the current region. We are also given a mobility model graph G_R. With this information we can find for what paths every RA $r_i \in A(g_o)$ is part of an optimal registration sequence. It is logical that the RA that is part in the optimal registration sequences of the most probable paths, then that RA would minimize Expct (see next property). We define the following concept of *optimality grade* for an RA r:

$$\mathcal{R}_r \stackrel{\text{def}}{\equiv} \sum_{p \in \mathcal{Q}(r, g_o, d)} \text{Prob}(p),$$

$\mathcal{Q}(r, g_o, d) \stackrel{\text{def}}{=} \{p \in \mathcal{P}(g_o, d) : |M^*(p)| = |M^*(r, p)|\}$. We compute $\mathcal{R}_{r_i}$ for each $r_i \in A(g_o)$ as follows: An initial value of 0 is assigned to every RA's optimality grade; if r_i is part of an optimal registration sequence on a path T_j, then the optimality grade $\mathcal{R}_{r_i}$ is increased by the probability of T_j (basically we find if a path T_j belongs to $\mathcal{Q}(r, g_o, d)$. At the end we choose the RA with maximal $\mathcal{R}$.

LEMMA 2. *If $r \in A(g_o)$ then*

$$Expct(G_R, g_o, r, d) = \sum_{p \in \mathcal{P}(g,k)} (Prob(p)|M^*(p)|) + (1 - \mathcal{R}_r)$$

PROOF. *C Expanding $Expct(G_R, g_o, r, d)$ we have:*
$Expct(G_R, g_o, r, d) = Prob(p_1)|M^*(r, p_1)| + Prob(p_2)|M^*(r, p_2)| + \ldots + Prob(p_k)|M^*(r, p_k)|$.
We group the terms $Prob(p_i)|M^*(r, p_i)|$ into two groups, group (A) which has all paths $p_i^{(A)}$ for which $|M^*(r, p_i^{(A)})| = |M^*(p_i^{(A)})|$, and group (B) which has all paths $p_i^{(B)}$ for which $|M^*(r, p_i^{(B)})| \neq |M^*(p_i^{(B)})|$:

$$\begin{aligned} &Expct(G_R, g_o, r, d) \quad = \\ &\quad Prob(p_1^{(A)})|M^*(r, p_1^{(A)})| + \ldots + Prob(p_m^{(A)})|M^*(r, p_m^{(A)})|) \\ &\quad + (Prob(p_1^{(B)})|M^*(r, p_1^{(B)})| + \ldots + Prob(p_n^{(B)})|M^*(r, p_n^{(B)})|) \end{aligned}$$

For the group (A) it is true that $|M^*(r, p_i^{(A)})| = |M^*(p_i^{(A)})|$ and for group (B) it is true that $|M^*(r, p_i^{(B)})| = |M^*(p_i^{(B)})| + 1$ (see lemma 1):

$$\begin{aligned} &Expct(G_R, g_o, r, d) \quad = \\ &\quad (Prob(p_1^{(A)})|M^*(p_1^{(A)})| + \ldots + Prob(p_m^{(A)})|M^*(p_m^{(A)})|) \\ &\quad + (Prob(p_1^{(B)})(|M^*(r, p_1^{(B)})| + 1) + \ldots + Prob(p_n^{(B)})(|M^*(p_n^{(B)})| \\ &\quad + 1)) \end{aligned}$$

The above gives:

$$\begin{aligned} &Expct(G_R, g_o, r, d) \quad = \\ &\quad Prob(p_1^{(A)})|M^*(p_1^{(A)})| + \ldots + Prob(p_m^{(A)})|M^*(p_m^{(A)})|) \\ &\quad + Prob(p_1^{(B)})|M^*(p_1^{(B)})| + \ldots + Prob(p_m^{(B)})|M^*(p_n^{(B)})| \\ &\quad + Prob(p_1^{(B)}) + \ldots + Prob(p_n^{(B)}) \end{aligned}$$

which gives

$$\begin{aligned} &Expct(G_R, g_o, r, d) \quad = \\ &\quad Prob(p_1)|M^*(p_1)| + Prob(p_2)|M^*(p_2)| \ldots + Prob(p_d)|M^*(p_d)|) \\ &\quad + Prob(p_1^{(B)}) + \ldots + Prob(p_n^{(B)}) \end{aligned}$$

Note that $Prob(p_1^{(B)}) + \ldots + Prob(p_n^{(B)})$ is the sum of probabilities of the paths for which there is no optimal registration sequence containing RA r as the first RA. This sum is $1 - \mathcal{R}_r$, therefore we have:

$$Expct(G_R, g_o, r, d) = \sum_{p \in \mathcal{P}(g,d)} (Prob(p)|M^*(p)|) + (1 - \mathcal{R}_r)$$

□

A formal description of the algorithm is given in Appendix A named Algorithm E. Algorithm E is a generalization of the "optimality grade method" that takes into account the RA the user last registered to (known as "initial" RA); E takes 3 parameters, the "initial" registration r_o, a current region g_o and a depth value d. For every path $p \in \mathcal{P}(g_o, d)$ and for each RA $r \in A(g_o)$ it tries to find whether $|M^*(r_o, p)| = |M^*(r, p)| + 1$, unless $r = r_o$ where the test is $|M^*(r_o, p)| = |M^*(r, p)|$; if the equality test is true, then it increments $\mathcal{R}_r$ by Prob(p). When it finishes all the path-RA pairs, it chooses the RA with maximal optimality grade. If no initial r_o is given, then we denote the algorithm as $E(g_o, d)$.

PROPERTY 4. *Algorithm $E(g_o, d)$ finds the registration r with minimal $Expct(G_R, g_o, r, d)$.*

Algorithm E returns the RA r that maximizes $\mathcal{R}$. For maximal $\mathcal{R}$, according to lemma 2, Expct(G_R, g_o, r, d) is minimized.

PROPERTY 5. *If $\alpha = \max_{g_i}\{|A(g_i)|\}$, and if the degree of the mobility model graph G_R is δ, Algorithm E takes $O(d\delta^d\alpha^3)$ steps to finish.*

It takes $O(\delta^d)$ set unions to find all paths of length d. It tries to find $\delta^d\alpha$ optimal paths with $O(d\alpha^2)$ steps to find an optimal path. Therefore it takes $O(d\delta^d\alpha^3)$ to finish.

Two example executions of Algorithm E on the model of Figure 3 are given in Table 1 (a) for $r_o = R_3, k = 2, A(9) = \{R_1, R_2, R_4\}$, and (b) for $r_o = R_1, k = 2, A(9) = \{R_1, R_2, R_4\}$. Table 1a shows the execution of the nested for loops of algorithms E for initial RA $r_o = R_3$, starting region $g_o = 9$ and depth 2 in the example model of Figure 3. Outer loop is the second column and inner loop is the third column. At each step, the pair (T_i, r_j) is taken and the case of $M^*(r_o, p) = |M^*(r, p)| + 1$ is tested. At the end of the execution we get $\mathcal{R}_{R_1} = 0.24$, $\mathcal{R}_{R_2} = 0.63$,$\mathcal{R}_{R_4} = 0.41$, so we should register with R_2. As a second example, we assume that the user is at the same vertex 9, but this time it is registered with R_1, d being 2 again. Table 1b shows the execution of E. From the last column, we get $\mathcal{R}_{R_1} = 1$,$\mathcal{R}_{R_2} = 0.48$ and $\mathcal{R}_{R_4} = 1$. Therefore, this time it is a good move to stay registered with R_1. In a second example, the user is again at vertex 9, but this time the vertex is registered to R_3. We notice that for $r \in A(g)$, algorithm E will choose r.

The next property states that if a user is registered to an RA, it will continue to be registered with the same RA as long as it is available in the traversed regions:

PROPERTY 6. *If $r_o \in A(g_o)$ then Algorithm $E(r_o, g_o, d)$ chooses r_o with $\mathcal{R}_{r_o} = 1$.*

Algorithm E is called to find if there are optimal registration sequence $M(r, T)$ for every $r \in A(g_o)$. If $r_o \in A(g_o)$, then for every trajectory $p \in \mathcal{P}(g_o, d)$ algorithm E will find that $|M^*(r_o, T)| = |M^*(r_o, T))|$, thus yielding $\mathcal{R}_(r_o)$=1.

Intuitively, it is not worth searching too deep: if the user's mobility shows a lot of variation from the model, then it is highly probable that the answer will be wrong; if the user's mobility shows little variation, then a short look-ahead should suffice. A practical bound to d would be a value d_M such that there exists trajectory of length d_M such that there exists $m_o < d_M$ $A(T(m_o)) \cap A(T(m_o+1)) = \emptyset$. Another value for d_M would be a value for which there exists trajectory T such that $A(T(1)) \cap A(T(d_M)) = \emptyset$. In such case, any choice that is made at T(1) is not going to change. Another value for d_M would be a value for which there exists trajectory T such

Table 1: Two example executions of Algorithm E on model of Figure 3: (a) $r_o = R_3, k = 2, A(9) = \{R_1, R_2, R_4\}$ (b) $r_o = R_1, k = 2, A(9) = \{R_1, R_2, R_4\}$.

Step	T_i	r_j	$r_o.M(r_o, T_i)$	$r_o.M(r_j, T_i)$	r_j accepted?	Prob(T_i)	$\mathcal{R}_{r_j}$
1	9,10,4	R_1	$R_3.R_2R_2R_2$	$R_3.R_1R_2R_2$	no		
2		R_2	$R_3.R_2R_2R_2$	$R_3.R_2R_2R_2$	yes	0.30	$\mathcal{R}_{R_2} = 0.30$
3		R_4	$R_3.R_2R_2R_2$	$R_3.R_4R_4R_2$	no		
4	9,10,6	R_1	$R_3.R_4R_4R_4$	$R_3.R_1R_4R_4$	no		
5		R_2	$R_3.R_4R_4R_4$	$R_3.R_2R_2R_4$	no		
6		R_4	$R_3.R_4R_4R_4$	$R_3.R_4R_4R_4$	yes	0.24	$\mathcal{R}_{R_4} = 0.24$
7	9,10,9	R_1	$R_3.R_2R_2R_2$	$R_3.R_1R_2R_2$	no		
8		R_2	$R_3.R_2R_2R_2$	$R_3.R_2R_2R_2$	yes	0.06	$\mathcal{R}_{R_2} = 0.36$
9		R_4	$R_3.R_2R_2R_2$	$R_3.R_4R_4R_4$	yes	0.06	$\mathcal{R}_{R_4} = 0.30$
10	9,8,7	R_1	$R_3.R_4R_4R_4$	$R_3.R_1R_4R_4$	no		
11		R_2	$R_3.R_4R_4R_4$	$R_3.R_2R_4R_4$	no		
12		R_4	$R_3.R_4R_4R_4$	$R_3.R_4R_4R_4$	yes	0.06	$\mathcal{R}_{R_4} = 0.36$
13	9,8,1	R_1	$R_3.R_1R_1R_1$	$R_3.R_1R_1R_1$	yes	0.01	$\mathcal{R}_{R_1} = 0.01$
14		R_2	$R_3.R_1R_1R_1$	$R_3.R_2R_1R_1$	no		
15		R_4	$R_3.R_1R_1R_1$	$R_3.R_4R_1R_1$	no		
16	9,8,9	R_1	$R_3.R_1R_1R_1$	$R_3.R_1R_1R_1$	yes	0.05	$\mathcal{R}_{R_1} = 0.06$
17		R_2	$R_3.R_1R_1R_1$	$R_3.R_2R_1R_1$	no		
18		R_4	$R_3.R_1R_1R_1$	$R_3.R_4R_4R_4$	yes	0.05	$\mathcal{R}_{R_4} = 0.41$
19	9,2,1	R_1	$R_3.R_1R_1R_1$	$R_3.R_1R_1R_1$	yes	0.03	$\mathcal{R}_{R_1} = 0.09$
20		R_2	$R_3.R_1R_1R_1$	$R_3.R_2R_1R_1$	no		
21		R_4	$R_3.R_1R_1R_1$	$R_3.R_4R_1R_1$	no		
22	9,2,3	R_1	$R_3.R_2R_2R_2$	$R_3.R_1R_1R_2$	no		
23		R_2	$R_3.R_2R_2R_2$	$R_3.R_2R_2R_2$	yes	0.12	$\mathcal{R}_{R_2} = 0.48$
24		R_4	$R_3.R_2R_2R_2$	$R_3.R_4R_2R_2$	no		
25	9,2,9	R_1	$R_3.R_1R_1R_1$	$R_3.R_1R_1R_1$	yes	0.15	$\mathcal{R}_{R_1} = 0.24$
26		R_2	$R_3.R_1R_1R_1$	$R_3.R_2R_2R_2$	yes	0.15	$\mathcal{R}_{R_2} = 0.63$
27		R_4	$R_3.R_1R_1R_1$	$R_3.R_4R_1R_1$	no		

(a)

Step	T_i	r_j	$r_o.M(r_o, T_i)$	$r_o.M(r_j, T_i)$	r_j accepted?	Prob(T_i)	$\mathcal{R}_{r_j}$
1	9,10,4	R_1	$R_1.R_2R_2R_2$	$R_1.R_1R_2R_2$	yes	0.30	$\mathcal{R}_{R_1} = 0.30$
2		R_2	$R_1.R_2R_2R_2$	$R_1.R_2R_2R_2$	yes	0.30	$\mathcal{R}_{R_2} = 0.30$
3		R_4	$R_1.R_2R_2R_2$	$R_1.R_4R_4R_2$	no		
4	9,10,6	R_1	$R_1.R_4R_4R_4$	$R_1.R_4R_4R_4$	yes	0.24	$\mathcal{R}_{R_1} = 0.54$
5		R_2	$R_1.R_4R_4R_4$	$R_1.R_2R_2R_4$	no		
6		R_4	$R_1.R_4R_4R_4$	$R_1.R_4R_4R_4$	yes	0.24	$\mathcal{R}_{R_4} = 0.24$
7	9,10,9	R_1	$R_1.R_2R_2R_2$	$R_1.R_1R_2R_2$	yes	0.06	$\mathcal{R}_{R_1} = 0.60$
8		R_2	$R_1.R_2R_2R_2$	$R_1.R_2R_2R_2$	yes	0.06	$\mathcal{R}_{R_2} = 0.36$
9		R_4	$R_1.R_2R_2R_2$	$R_1.R_4R_4R_4$	yes	0.06	$\mathcal{R}_{R_4} = 0.30$
10	9,8,7	R_1	$R_1.R_4R_4R_4$	$R_1.R_1R_4R_4$	yes	0.04	$\mathcal{R}_{R_1} = 0.64$
11		R_2	$R_1.R_4R_4R_4$	$R_1.R_2R_4R_4$	no		
12		R_4	$R_1.R_4R_4R_4$	$R_1.R_4R_4R_4$	yes	0.06	$\mathcal{R}_{R_4} = 0.34$
13	9,8,1	R_1	$R_1.R_1R_1R_1$	$R_1.R_1R_1R_1$	yes	0.01	$\mathcal{R}_{R_1} = 0.65$
14		R_2	$R_1.R_1R_1R_1$	$R_1.R_2R_1R_1$	no		
15		R_4	$R_1.R_1R_1R_1$	$R_1.R_4R_1R_1$	no		
16	9,8,9	R_1	$R_1.R_1R_1R_1$	$R_1.R_1R_1R_1$	yes	0.05	$\mathcal{R}_{R_1} = 0.70$
17		R_2	$R_1.R_1R_1R_1$	$R_1.R_2R_1R_1$	no		
18		R_4	$R_1.R_1R_1R_1$	$R_1.R_4R_1R_1$	no		
19	9,2,1	R_1	$R_1.R_1R_1R_1$	$R_1.R_1R_1R_1$	yes	0.03	$\mathcal{R}_{R_1} = 0.73$
20		R_2	$R_1.R_1R_1R_1$	$R_1.R_2R_1R_1$	no		
21		R_4	$R_1.R_1R_1R_1$	$R_1.R_4R_1R_1$	no		
22	9,2,3	R_1	$R_1.R_2R_2R_2$	$R_1.R_1R_1R_2$	yes	0.12	$\mathcal{R}_{R_1} = 0.85$
23		R_2	$R_1.R_2R_2R_2$	$R_1.R_2R_2R_2$	yes	0.12	$\mathcal{R}_{R_2} = 0.48$
24		R_4	$R_1.R_2R_2R_2$	$R_1.R_4R_2R_2$	no		
25	9,2,9	R_1	$R_1.R_1R_1R_1$	$R_1.R_1R_1R_1$	yes	0.15	$\mathcal{R}_{R_1} = 1.00$
26		R_2	$R_1.R_1R_1R_1$	$R_1.R_2R_2R_2$	no		
27		R_4	$R_1.R_1R_1R_1$	$R_1.R_4R_1R_1$	no		

(b)

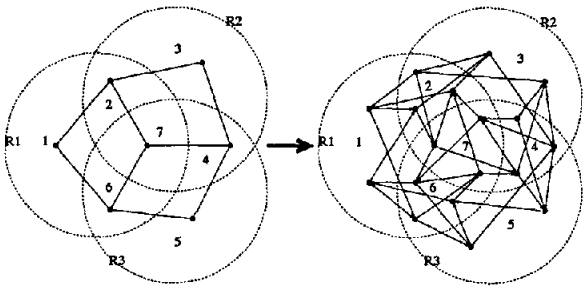

Figure 4: An enhancement of the original random walk model G_R: every vertex in the graph is representing a (*previous region, current region*) pair. This model captures the property of direction of movement.

that there exists $m_o < d_M$ $|A(T(m_o))| = 1$. On the other hand, we need d to be at least as big as the number of regions in any RA. Nevertheless, the real bound to d is how much time there is available to run the algorithm. Execution time increases exponentially with respect to d, so d can't be large.

4.3 Soft and Hard Registrations

Based on Property 6 we define two approaches to the problem: the *preemptive* ($E(g_o, d)$), which may make registration prior to the user moving out of the servicing RA, in view of a certain trajectory of the user; and the *non-preemptive* ($E(r_o, g_o, d)$), which doesn't let the user register to a new RA unless the user moves into an region where the RA is not available. The non-preemptive *always* saves registrations (see lemma 3). We understand that all the registrations of the non-preemptive approach are *hard*; under a hard registration, the user is exiting the servicing RA and a registration *has* to be performed immediately. Based on the Property 6, we don't have to run the non-preemptive scheme upon each change of a region, we only run Algorithm E when the last registered RA is no longer available. On the other hand, the preemptive scheme requires that algorithm E is run every time the user enters a new region. The preemptive approach has the ability to make soft registrations. Actually, when it is right about the prediction, transforms a *hard* registration to a *soft* registration; if it is wrong, then it pays with an extra *hard* registration.

The following lemma shows that the preemptive produces at least as many total registrations as the non-preemptive scheme:

LEMMA 3. *Given a user path, the preemptive scheme will produce at least as many registrations as the non-preemptive scheme for the same look-ahead depth.*

PROOF. *C The output of the non-preemptive scheme is a sequence* $s_N = x_1 x_2 \ldots x_n$. *Due to the non-preemptive nature of the algorithm, the sequence is of the type* $s_N = a_1 a_2 \ldots a_{n_a} b_{n_a+1} b_{n_a+2} \ldots b_{n_b} \ldots$, *where a subsequence* $a_1 a_2 \ldots a_{n_a}$ *is of length* n_a *of the same character a (RA id). The output of the preemptive scheme is* $s_A = y_1 y_2 y_3 \ldots y_{n'}$. *We claim that at locations* 1, $n_a + 1$, $n_b + 2$, ... *both sequences have the same character, that is* $x_1 = y_1$, $x_{n_a+1} = y_{n_a+1}$. *This is easy to understand, based on the Property 2 of Algorithm M, when we realize that at those locations the non-preemptive scheme acts like preemptive.* □

Under a cost model where the optional registrations cost less than the hard ones, the preemptive approach can be more efficient, depending on the *predictability* of the user's movement. A justification for assigning different costs on hard and soft registrations is that hard are time-critical: under a hard registration the user is moving into a cell not serviced by the current RA, and a registration, including resource allocation, has to be performed with real-time constraints. A soft registration doesn't have such real-time constraints, the resource allocation and the registration generally can be easily done before the user exits the current region.

5. CONCLUSIONS AND FUTURE WORK

We have presented a way to minimize the number of registrations that have to be performed for a mobile user. We formulated and proposed solutions for two versions of the problem: deterministic and stochastic. The solution for the deterministic version involves constructing a weighted directed acyclic graph (dag) and finding a shortest path in the dag. We have used the solution of the deterministic case to find a "good" registration for the stochastic version. Recently, Wu *et al.* have presented a mobility management scheme that is based on mobile's moving behavior [9]. It is observed that many mobile users have a very predictable moving behavior i.e. given a time of the day and location of the user, the system can very accurately predict based on the past history the destination of the user and his/her prefered route. Based on this knowledge about user's mobility behavior, the deterministic and stochastic techniques presented in this paper can be used to optimize the number of registrations.

The goodness metric which we have uses is the expected number of minimum registrations. We have proved the optimal of the stochastic algorithm under this metric. Further, we have proposed two versions of the stochastic solution: pre-emptive and non-preemptive. These versions differ in the number of hard and soft registrations which are needed. Future work will involve developing optimal solutions for stochastic case under different metrics. And experimentally studying the difference in pre-emptive and non-preemptive versions under different cost models for hard and soft registrations.

The quality of the output of algorithm E highly depends on the detail of the input model G_R. The more detailed the model is, the better the solution of the algorithm E will be. For example, the random model, as shown in Figure 3 doesn't capture an important property of the users mobility: direction. In order to make the model capture direction, we need to refine it. Figure 4 shows an example mobility model inspired from the model in Figure 3. Every region is mapped to as many vertices as the neighboring regions, thus each vertex implying a (*previous region, current region*) pair. we can divide a region into smaller parts (zones), so that the model can capture more details about the location of the user (see Figure 5). The subdivision of regions can help the model capture a user's proximity to another region. As a drawback, the size of the graph increases dramatically. This suggests a trade-off between level of detail in the random walk model (accuracy of the algorithm's result) versus the running time of the algorithm and overhead imposed for maintaining the mobility model. A future work would involve the study of the algorithm under different or more detailed mobility models. There can be other reasons to chose an available registration such as for balancing load across different RAs. We also plan to investigate these different policies in future.

5.1 Acknowledgments

The authors would like the anonymous refrees for their constructive comments which helped to improve the paper. This work is supported by NSF grant # ANI-0196156.

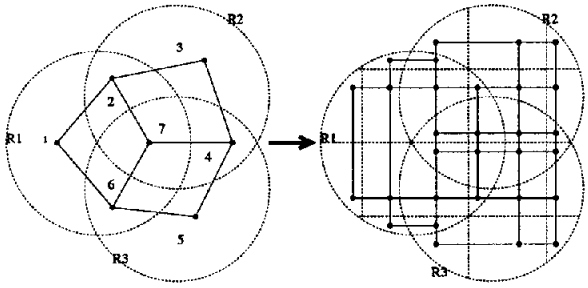

Figure 5: An enhancement of the original random walk model G_R: every region is divided into zones, thus giving a more detailed model of the user's mobility.

6. REFERENCES

[1] Y. Bejerano and I. Cidon. An Efficient Mobility Management Strategy for Personal Communication Systems. In *Proc. Fourth Annual IEEE/ACM Int'l Conf on Mobile Computing and Networking (MobiCom'98)*, pages 215–222, Dallas, TX, April 1998.

[2] S. K. Das and S. K. Sen. Adaptive Location Prediction Strategies Based on a Hierarchical Network Model in Cellular Mobile Environment. Proceedings of Second International Mobile Computing Conference (IMCC), March 1996.

[3] EIA/TIA. Cellular radio-telecommunication intersystem operations. Technical Report IS-41, Revision B, EIA/TIA, 1991.

[4] J. S. M. Ho and I. F. Akyildiz. Dynamic Hierarchical Location Management in PCS Networks. *IEEE/ACM Transactions on Networking*, 5(5):646–660, October 1997.

[5] S. Kryukova, B. Massingill, and B. Sanders. An Algorithm for Distributed Location Management in Networks of Mobile Computers. Technical report, California Institute of Technology, Computer Science department, 1996.

[6] S. Rajagopalan and B. R. Badrinath. An Adaptive Location Management Strategy for Mobile IP. In *Proc. ACM/IEEE MobiCom*, 1995.

[7] G. Varsamopolous and S. K. S. Gupta. On dynamically adapting registration areas to user mobility patterns in PCS networks. In *Proc. Int'l Workshop on Collaboration and Mobile Computing (IWCMC'99)*, Aizu, Japan, September 1999.

[8] I. F. Akyildiz and W. Wang. Intersystem Location Update and Paging Schemes for Multitier Wireless Networks. In *Proc. ACM/IEEE MobiCom*, Boston, MA, August 2000.

[9] H.-K. Wu, M.-H. Jin, J.-T. Horng, and C.Y. Ke Personal Paging Area Design Based on Mobile's Moving Behaviors. In *Proc. IEEE Infocom'01*, Anchorage, Alaska, April, 2001.

APPENDIX

A. ALGORITHMS M AND E

Algorithm M (see pseudo-code in Figure 6) takes a registration id r_o and a trajectory T. It also uses the availability vector A of every region the trajectory T. With the aid of A, M constructs a weighted graph G_M of alternative registration sequences and finds a shortest path in G_M, thus finding an optimal registration sequence.

```
Algorithm M(RA r_o, region array T)
begin
(assume the knowledge of A_T, which if it is indexed by
a region g gives an array of all the RAs available in that region,
and if it is indexed by a region g and a number i ∈ A(g),
gives the i^th RA available in that region. W(e) is the weight of edge e)

(construct a graph G_M, every vertex is essentially a pair (RA,region))
  vertex s,t ; RA u,v ; region r,g,h ;string z ; integer i ;
  E_M := ∅; s := (r_o, null); V_M := {s};
  for r := A(T(1), 1)toA(T(1), |A(T(1))|) do
    begin
      V_M := V_M ∪ (r, T(1)); E := E ∪ {(s, (r, T(1))};
      if (r_o = r) then W(s, (r, T(1))) := 0;
      else W(s, (r, T(1))) := 1; endif
    end

  for i := 1 to |T| − 1 do
    begin
      g := T(i); h := T(i + 1) ;
      for u := A(g, 1) to A(g, |A(g)|) do
        begin
          if (i > 1) then V_M := V_M ∪ {(u, g)}; endif
          for v := A(h, 1) to A(h, |A(h)|) do
            begin
              V_M := V_M ∪ {(v, h)};
              E := E ∪ {((u, g), (v, h))};
              if (u = v) then W((u, g), (v, h)) := 0;
              else W((u, g), (v, h)) := 1; endif
            end
        end
    end

  t := (null, null); V_M := V_M ∪ {t};
  for r := A(T(|T|), 1) to A(T(|T|), |A(T(1))|) do
    begin
      E := E ∪ ((r, T(|T|)), t); W((r, T(|T|)), t) := 0;
    end

(solve the shortest path problem on the constructed graph G_M)
  - find a shortest path from s to t in the form (s, q_1, q_2, ..., q_m, t).
  for i := 1 to m do
    begin
      y := RA_of(q_1);
      z := zy; (concatenate z, y)
    end
  return z;
end
```

Figure 6: Pseudo-code for Algorithm M.

```
All-Paths(graph G_R, region g_o, integer d, set P_o, path p)
returns set P
  begin
    foreach outgoing edge e_i = (g_o, g_i) at vertex g_o in G_R do
      begin
        p:=pg_i; (concatenate p and g_i)
        if (d = 1) then P := P_o ∪ {p};
        else P := All-Paths(G_R, g_i, d − 1, P, p) endif;
      end
    return P;
  end
```

```
Algorithm E(RA r_o, Region g, integer d)
begin
(assume the knowledge of the random-walk model
graph G_R of the mobile unit)
  integer opt; set Paths;real R;
  Paths := All-Paths(G_R, g, r_o, d, ∅, ε);
  for each path T_i ∈ Paths do
    begin
      opt := |M*(r_o, T_i)|;
      for each r_j ∈ A(g_o) do
        begin
          if (r_j ≠ r_o) then
            if (opt = 1 + |M*(r_j, T_i)|)
            then R_{r_j} := R_{r_j} + Prob(T_i);
          else
            R_{r_i} := R_{r_i} + Prob(T_i);
          endif
        end
    end
  select next RA r := argmax_{r_j}(R_{r_j}), r_j ∈ A(g_o)
end
```

Figure 7: Pseudo-code for Algorithms E and All-Paths.

Algorithm E (see pseudo-code in Figure 7) takes a initial registration id r_o, a current region g and a search depth value d. It assumes the knowledge of the random walk model, which is viewed as a digraph G_M in which every vertex is a region and every directed edge is a transition from a state to another state. Edges are weighted, denoting probability of transition, and the sum of the weights of all outgoing edges of a vertex is exactly 1. With the aid of G_M, algorithm E finds all possible trajectories of length d that start at r_o.

The algorithm All-Paths(G_R,g_o,d,$\emptyset$,ϵ) in Figure 7, where $\emptyset$ is the empty set and ϵ is the empty string, is a recursive algorithm that does a depth-first search and returns a set of all paths of length d starting at g_o.