Wireless Networks 1 (1995) 187–196

Dynamic mobile user location update for wireless PCS networks

Ian F. Akyildiz and Joseph S.M. Ho

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

Received November 1994

Abstract. The basic architecture of a personal communication network consists of a wireline network and mobile terminals. Each mobile terminal communicates with the wireline network through a nearby base station. In order to route incoming calls to a destination mobile terminal, the network must keep track of the location of each mobile terminal from time to time. This is achieved by *location update* such that each mobile terminal reports its current location to the network at specific time points. When an incoming call arrives, the network will page the mobile terminal starting from the last updated location. A trade-off, therefore, exists between the frequency of location update and the number of locations paged in order to track down the mobile terminal. This paper introduces a location update policy which minimizes the cost of mobile terminal location tracking. A mobile terminal dynamically determines when to update after moving to a new cell based on its mobility pattern and the incoming call arrival probability. The performance of this scheme is close to that of the optimal policy reported earlier. However, the processing time requirement of this scheme is very low. The minimal computation required by this scheme enables its usage in mobile terminals which has limited energy supply and computational power.

1. Introduction

One of the major constraints of wireless personal communication services (PCSs) is the need to support a large number of mobile terminals under a very limited bandwidth allocation. Current PCS networks employ a cellular architecture such that a large area, such as a city, is divided into smaller areas called *cells*. Mobile terminals within a cell communicate with other terminals (mobile or stationary) through a base station which is installed within the cell. This base station is connected to other base stations and stationary terminals (such as ordinary wired telephones) through an underlying wireline network. In order for the network to efficiently route incoming calls to a mobile terminal, each mobile terminal is required to report its location to the network at specific points in time. This reporting process is called *location update*. There are a number of ways to determine these time points. The most commonly used scheme is to group the cells into location areas (LAs). A mobile terminal performs location update whenever it enters a new LA. When an incoming call arrives, the network sends a signal to each cell of the LA in order to locate the mobile terminal. This signaling process is called *terminal paging*. It is clear that if each LA consists of only one cell, the network knows exactly the location of each mobile terminal. In this case, there is no cost incurred in terminal paging. However, the cost for location update will be very high as the mobile terminal has to report its location every time it enters a cell. A trade off therefore exists between the cost of location update and the cost of terminal paging. Given the respective cost for location update and terminal paging, a good mobile terminal tracking policy should select the location update time points such that the total cost (location update and terminal paging) is minimized. Moreover, the scheme must be simple enough such that the computational requirement is minimal. It is infeasible to implement a computational intensive policy in a mobile terminal (such as a handheld unit) which has a very limited energy supply and minimal computational power.

As the number of PCS subscribers keeps increasing, smaller sized cells have to be used in order to accommodate the large number of mobile terminals under a constant amount of bandwidth allocation. The current scheme of locating a mobile terminal by sending a signal to all cells within the LA is not feasible. This may result in excessive amount of bandwidth used only for paging the mobile terminals. More sophisticated schemes that can attain a much lower cost is needed.

In [1], a mobile terminal location update scheme is introduced. Under this scheme, a subset of the cells is selected as the *reporting cells*. A mobile terminal reports its location only when entering one of these reporting cells. When an incoming call arrives, the search starts from the cell where the mobile terminal last reported its location. As described in [1], the selection of the set of reporting cells such that the total cost is minimized is an NP-complete problem. Besides, this scheme cannot be adjusted based on the call arrival and movement patterns of individual mobile terminal. A mobile terminal that moves in and out of a reporting cell frequently will perform excessive unnecessary location updates if the arrival probability of incoming call is low. This scheme is static in the sense that the reporting cells are fixed. It cannot easily be adjusted dynamically in response to changes in system parameters. Recent research efforts are focused primarily on dynamic location update mechanisms such that reporting is based on the movement of the mobile terminals and the frequency of incoming calls. In [2], three dynamic location update strategies, *time-based*, *movement-based* and *distance-based*, are studied. Under these three schemes, location updates are performed based on the time elapsed, the number of movements performed, and the distance traveled, respectively, since the last location update. It is demonstrated that the distance-based strategy has the best performance but its implementation incurs the highest overhead. It is proposed in [8] that the size of the LA should be terminal dependent. An algorithm is introduced that determines the size of LA based on the mobility and the call arrival probabilities of a mobile terminal. Another attempt is given in [5] which tries to find the optimal distance the mobile terminal travels before a location update is necessary. An optimal policy is obtained in [5]. In [3], the optimal location update distance is determined for a distance-based location update policy subject to maximum delay constraints. However, the computational overhead of the mechanisms reported in [3,5] is quite significant. It may be infeasible to implement these mechanisms in mobile terminals with limited computational power.

There are other efforts, such as [4,6,7], that try to improve the mechanism used by the network to locate a mobile terminal. This paper, on the other hand, focuses primarily on what the mobile terminals can do to reduce the cost of terminal tracking. We believe that these schemes can be combined to generate an efficient and low cost mobile terminal tracking mechanism.

In this paper, we introduce a dynamic location update mechanism. Our scheme can be tailored to the particular mobility and call arrival pattern of each individual mobile terminal. The location update decision process is distributed over time. Changes in the call arrival and mobility pattern of a terminal are detected and are taken into account from time to time. The computational requirement is minimal which makes our scheme feasible for use in mobile terminals with limited computational power. Most of the previous schemes assume the mobility of the mobile terminals to follow specific models such as the memoryless random walk model [2,3,5] or the Markovian model [2]. Our mechanism is not based on a specific assumption on movement pattern. In addition, instead of assuming the call arrival probability to be given, we introduce a method for estimating the call arrival probability for each mobile terminal. As demonstrated by experiments, the performance of our mechanism stays close to that produced by an optimal policy given in [3] under various movement and incoming call arrival probabilities. The performance of our mechanism never falls below that of a no-update policy (location update is never performed). This means that the cost effectiveness of mobile terminal location tracking is improved by using the location update policy.

This paper is organized as follows. In section 2, we describe the system model. In section 3 we introduce the proposed dynamic location update policy. A method for estimating the incoming call arrival probability is given in section 4. In section 5 we present the performance evaluation for the proposed location update mechanism. The conclusion is given in section 6.

2. System model

2.1. Terminal mobility

We assume mobile terminals are residing in an M-dimensional space which is divided into cells. At every discrete time t, a mobile terminal may move to one of its neighboring cells. Our location update mechanism does not require specific assumptions on the mobility pattern of the mobile terminals and any knowledge of specific mobility parameters. Moreover, our mechanism is not constrained by any specific dimension, M, and it can be applied under both the one- and two-dimensional mobility model. However, in order to demonstrate the performance of our mechanism, we use the one-dimensional memoryless random walk mobility model throughout this paper. As will also be demonstrated later in this paper, the only information required by our mechanism is the call arrival distribution, the cost for terminal paging and the cost for location update.

Assume $X_t \in \Re^M$ is the coordinate of the cell where a mobile terminal is located at discrete time t. The mobile terminal will stay at cell X_t at time $t+1$ with probability q or it will move to one of the neighboring cells with probability p, where $p+q=1$. We restrict movement only to immediate neighboring cells. This is a reasonable restriction as it is unlikely that a mobile terminal will move across one or more cells during one discrete time period. If the mobile terminal decides to move, we assume that it has an equal probability of moving to any one of the neighboring cells. Fig. 1 gives an example of a one-dimensional PCS coverage area with 10 cells. As can be seen in Fig. 1, cells are arranged in a linear fashion. Each cell has exactly two neighboring cells (except the first and the last cells.) The neighbors of cell i are cells $i+1$ and $i-1$, respectively. The location of a mobile terminal at time $t+1$ is formally given as follows:

$$X_{t+1} = \begin{cases} X_t & \text{with probability } q, \\ X_t + 1 & \text{with probability } p/2, \\ X_t - 1 & \text{with probability } p/2. \end{cases} \quad (1)$$

1	2	3	4	5	6	7	8	9	10

Fig. 1. One-dimensional PCS coverage area.

The size of each cell is determined based on the number of mobile subscribers, the number of channels per cell and the channel allocation scheme used. In this paper, we introduce a dynamic location update mechanism assuming that the size of cells is given. Our scheme works both in the *macrocell* and the *microcell* environment. We note that if the size of cells is small, the movement probability, p, is relatively high and vice versa. We consider a wide range of values for the movement probability in the experiments given in section 5.

We denote the interarrival times of incoming calls by T. Assume that T is a generally distributed iid (independent and identically distributed) random variable with the probability distribution function $F(t)$ and the mean $1/\lambda$.

2.2. Mobility tracking

We assume that the location identifier of a cell is broadcast by the base station periodically such that every terminal knows exactly its own location at any given time. The network records the location of each mobile terminal whenever a location update is initiated by the mobile terminal. In this paper, we do not look into the handoff mechanism and its associated cost. We assume that the network has the up-to-date location information of a mobile terminal throughout the progress of a call. The network records the location information of a mobile terminal in a database whenever up-to-date location information is available. This recording process is called *location registration*.

In order to determine if a mobile terminal is located in a particular cell, the network performs the following steps:

1. Sends a polling signal to the target cell and waits until a timeout occurs.
2. If a reply is received before timeout, the destination terminal is located in the target cell.
3. If no reply is received, the destination terminal is not in the target cell.

We define the *center cell* to be the cell where the mobile terminal's last location registration occurred. When an incoming call arrives, the PCS network polls the cells in the vicinity of the center cell in a shortest-distance-first order. In the PCS coverage area given in Figure 1, assume that the destination mobile terminal is located at cell 2 when the call arrives. Also assume that cell 6 is the center cell of the mobile terminal. The PCS network will poll the cells in the following order: 6, 5-7, 4-8, 3-9, 2-10 (x-y means that cells x and y are polled at the same time). In general, given that a mobile terminal is currently located in cell i and the center cell of the mobile terminal is cell j, the distance traveled since the mobile terminal's last location registration is then

$$k = |i - j| . \qquad (2)$$

The number of cells polled before the mobile terminal is successfully located is

$$h(k) = 2k + 1 . \qquad (3)$$

We assume that when an incoming call arrives, the network is able to locate the destination mobile terminal (by terminal paging) within the same discrete time period. The following is an ordered list of tasks that a mobile terminal performs within each discrete time unit:

1. Determine if the location of the mobile terminal is changed as compared to its location during the previous discrete time period.
2. If the location of the mobile terminal is changed, calculate the next location update time.
3. If an incoming call is pending, receive the call and set the location update time to infinity (no location update is necessary until the next movement).
4. If current time is bigger than or equal to the next location update time, perform location update and set next location update time to infinity.

A method for determining the next location update time is given in section 3. As discussed before, the operation of the mobile terminal during a call is not considered here. The above list is valid only when no call is currently in progress. For the rest of this paper, we assume that the time is always expressed in the number of discrete time periods. We will not explicitly indicate the unit of time hereafter.

3. A dynamic location update mechanism

The dynamic location update policy determines the next location update time based on the distance traveled since last location registration and the incoming call arrival probability. We assume that the cost for each location update, denoted by U, is independent of the location of the mobile terminal. We also assume that the cost for terminal paging is proportional to the number of cells polled before the mobile terminal is found. Given that the mobile terminal is k cells away from its center cell, the cost for terminal paging is

$$Q(k) = V(2k + 1) , \qquad (4)$$

where V represents the cost for polling each cell during the terminal paging process. It is clear that if the arrival time of the next incoming call is known in advance, the mobile terminal should perform a location update right before the call arrival if the cost for location update is less than the paging cost ($U < Q(k)$) at that time. Otherwise, no location update should be performed because the paging cost is lower than the location update cost. In reality, we do not know in advance the arrival time of each incoming call. However, given the incoming call interarrival times distribution function, $F(t)$, and the time elapsed since the last call arrival, we can determine the probability distribution of the residual call arrival time. Assume that T is the incoming call interarrival time with distribution $F(t)$ and t_e is the time elapsed since last call arrival. The probability of an incoming call arrival within the next Δt time units is

$$G(\Delta t, t_e) = Pr[T \leq \Delta t + t_e | T > t_e] \tag{5}$$

$$= \frac{Pr[t_e < T \leq \Delta t + t_e]}{Pr[T > t_e]} \tag{6}$$

$$= \frac{Pr[T \leq \Delta t + t_e] - Pr[T \leq t_e]}{Pr[T > t_e]} \tag{7}$$

$$= \frac{F(\Delta t + t_e) - F(t_e)}{1 - F(t_e)} \, . \tag{8}$$

Since the cumulative probability of call arrival increases with time, the risk of not updating the location also increases with time. We define the weighted paging cost Δt time units from the current time as

$$W(\Delta t, t_e, k) = G(\Delta t, t_e) Q(k) \, . \tag{9}$$

The weighted paging cost at a given time interval after a movement is the terminal paging cost multiplied by the probability of call arrival during that time interval. As it can be seen in eq. (9), the weighted paging cost increases with time. Its value approaches that of the paging cost, $Q(k)$, as the time interval, Δt, increases. The weighted paging cost, therefore, takes into account of the increasing probability of having to pay the terminal paging cost as time advances. A location update should be performed at the time when the weighted paging cost exceeds the location update cost. We propose a location update mechanism which determines the time of the next location update based on the call interarrival time distribution function, $F(t)$, the distance traveled since the mobile terminal's last location registration, k, and the time elapsed since the last call arrival, t_e. The following equation gives the weighted paging cost on the left hand side and the location update cost on the right hand side:

$$G(\Delta t, t_e) Q(k) = U \, . \tag{10}$$

Given the values of k and t_e, the time until the next location update, $\tau(k)$, can be determined by solving eq. (10) for Δt:

$$\tau(k) = \{\Delta t \mid G(\Delta t, t_e) Q(k) = U\} \, . \tag{11}$$

Since the probability $G(\Delta t, t_e)$ can never exceed 1, eq. (10) is valid only when $U \leqslant Q(k)$. When $U > Q(k)$, the weighted paging cost, $W(\Delta t, t_e, k)$, can never exceed the location update cost, U. No location update is necessary under this situation, i.e. $\tau(k) = \infty$. As the distance, k, changes when the mobile terminal moves from cell to cell, $\tau(k)$ has to be updated. Assume the last movement of the mobile terminal occurred at time t_m when the distance of the mobile terminal from its center cell is k. The next location update time is

$$t_u = t_m + \tau(k) \, . \tag{12}$$

If the mobile terminal moves to a new location before t_u, a new value for t_u will be determined based on the up-to-date distance k and the time elapsed since the last call arrival t_e. The proposed location update mechanism is summarized as follows:

1. The following information is collected whenever the mobile terminal enters a cell:
   - The mobile terminal's last movement time, t_m (this is the same as the current time.)
   - The mobile terminal's cell location during the last location registration, j.
   - The mobile terminal's current cell location, i.
   - The distance traveled since the last location registration, $k = |i - j|$.
   - The cost for the network to page for the mobile terminal, $Q(k) = V(2k + 1)$.
   - The time elapsed since the last call arrival, t_e.
2. Using the data obtained in the previous step, determine the time until next location update, $\tau(k)$, by solving equation (10) for Δt.
3. Record the $\tau(k)$ value obtained in the previous step and initiate a location update at time $t_u = t_m + \tau(k)$.

3.1. Geometric call interarrival time distribution

Here we consider the special case where the call interarrival time distribution is geometric. Based on this assumption, we develop the expression for the time until next location update, $\tau(k)$. The result obtained in this section is useful for comparison with previous location update schemes [3,5] as many of which employ the geometric call interarrival time assumption. The probability distribution of call interarrival time is

$$F(t) = 1 - (1 - \lambda)^t \, . \tag{13}$$

This is the probability distribution function for the geometric distribution with probability λ. We assume at this point the arrival probability of incoming calls, λ, is given. A method for estimating λ for each mobile terminal is described in section 4. Because of the memoryless property of the geometric distribution, the residual call arrival time is independent of the time elapsed since the last call arrival. As a result, the probability distribution of the residual call arrival time given the time elapsed since the last call arrival, t_e, is the same as $F(t)$ given in eq. (13):

$$G(\Delta t, t_e) = 1 - (1 - \lambda)^{\Delta t} \, . \tag{14}$$

Substituting eq. (14) in eq. (10) and solving for Δt, the next location update time, $\tau(k)$, is

$$\tau(k) = \begin{cases} \infty & \text{if } k = 0\,, \\ \infty & \text{if } U > Q(k)\,, \\ \lfloor \log_{(1-\lambda)}(1 - \frac{U}{Q(k)}) \rfloor & \text{otherwise}\,. \end{cases} \quad (15)$$

A $\tau(k)$ value of ∞ indicates that no location update is necessary. As can be seen from eq. (15), no location update is needed either when $k = 0$ or when $U > Q(k)$. The first case represents the situation right after a location registration. The network knows exactly the location of the mobile terminal. No location update is therefore needed until after the next movement. In the second case, the location update cost is always higher than the weighted paging cost no matter what the probability $G(\Delta t, t_e)$ is. It is therefore more economical not to perform any location update. The next location update time, t_u, is computed from eq. (12).

When compared to other location update mechanisms, the overhead incurred by our proposed mechanism is low. The last part of eq. (15) can be rearranged to obtain

$$\tau(k) = \left\lfloor \log_{(1-\lambda)}\left(1 - \frac{U}{V}\frac{1}{2k+1}\right)\right\rfloor . \quad (16)$$

It can be seen in eq. (16) that it takes precisely one addition, one subtraction, one multiplication, one division, and one logarithmic (with base $(1-\lambda)$) operations to determine the time until next location update each time the mobile terminal enters a cell (assume that the U/V division is done in advance). Compared to other location update policies, such as [5], which employs an iterative approach to determine the location update distance, our mechanism is significantly simpler. Besides, the overhead of our mechanism is not affected by the values of the parameters, such as λ. Apart from its simplicity, it is demonstrated in section 5 that the cost of the proposed mechanism is close to the optimal cost of a distance based location update scheme. The minimal and predictable cost makes it a good choice of location update mechanism.

4. Estimating incoming call arrival probability

The location update schemes proposed here as well as those given in [3,5,8] assume that the value of the call arrival probability is readily available. Two methods can be used to keep track of the call arrival probability for a mobile terminal. In the first method, the network is responsible for keeping track of the call arrival probability for each mobile terminal. The network calculates the call arrival probabilities and stores the results in a database from time to time. The call arrival probability values are transmitted to the mobile terminals when requests for data are received. This method requires significant memory storage space in the network as the number of mobile terminals keeps increasing. Besides, additional network and wireless bandwidth are consumed for transmitting these arrival probability data from the database to the mobile terminals. This will significantly increase the cost of the mobile terminal location tracking mechanism. The alternative method is to have each mobile terminal calculates its own call arrival probability locally. This method consumes no extra network and wireless bandwidth. However, some processing is required at each mobile terminal to keep track of the number of call arrivals within a certain period of time. In this section we introduce a method for estimating the call arrival probability at the mobile terminals. One basic strategy is to record the number of call arrivals, n, starting from a given time, t_1, up to the current time, t_c, in a register inside the mobile terminal. This arrival count register is increased by one when an incoming call arrives. The arrival probability is then estimated by dividing this arrival count by the length of the time interval:

$$\lambda = \frac{n}{t_c - t_1}\,. \quad (17)$$

There are a number of problems to be addressed if this strategy is used:

- The arrival count cannot be arbitrarily large. The register must be cleared periodically to avoid register overflow.
- Clearing the call arrival count register will suddenly lower the estimated call arrival probability to zero. A considerable amount of time is needed before a steady arrival probability value can be obtained. A mechanism is needed so that a continuous call arrival probability value can be obtained.

We propose a modified scheme such that two call arrival count registers are used. Define a *long interval*, t_l, to be the maximum time interval used for estimating the call arrival probability. Similarly, a *short interval*, $t_s = \frac{1}{2}t_l$, is defined as the minimum time interval used for estimating the call arrival probability. When an incoming call arrives, both registers are incremented by one. Only one of the registers is used for estimating the call arrival probability at any given time. We call this register the active register and the other one the inactive register. The role of the two registers is switched every t_s time interval. The active register is cleared before it changes to the inactive state. Assume t_{rc} and t_c to be the times the roles of the registers were last switched and the current time, respectively. Also assume r_a and r_i to be the values in the active and the inactive registers, respectively. The incoming call arrival probability, λ, is estimated by

$$\lambda = \frac{r_a}{t_c - t_{rc} + t_s}\,. \quad (18)$$

This method estimates the call arrival probability using a time interval with length between t_s and t_l. Selecting the appropriate values for t_s and t_l should enable this mechanism to capture long term arrival probability changes but at the same time it is not overly reactive to short term variations. A register is cleared every t_l time units. However, clearing the register will not lead to an arrival probability of zero because of the availability of the alternative register. The overhead of this mechanism is very low. Processing is required only when an incoming call arrives. The processing is limited to incrementing the two registers and the subtraction and division operations needed to calculate the call arrival probability. As described before, this mechanism does not increase the bandwidth overhead of the network.

5. Performance evaluation

We developed an analytical model to study the performance of the proposed dynamic location update mechanism. We use both the random walk mobility model and the geometric call interarrival time distribution.

5.1. Analytical model

Here we develop an exact analytical model for the dynamic location update mechanism. We model the activity of a mobile terminal using an imbedded Markov chain of which the state i ($i \geq 0$) is defined as the distance between the current location of the mobile terminal and its center cell. State transitions of the imbedded Markov chain occurs right after the following time epoches:

- The movement of the mobile terminal to one of the neighboring cells.
- The arrival of incoming call.
- The performance of location update by the mobile terminal.

The state transition diagram for the imbedded Markov chain is given in Fig. 2. The transitions $a_{i,i+1}$ ($0 \leq i \leq N$) and $b_{j,j-1}$ ($0 < j \leq N$) represent the movement of the mobile terminal to a neighboring cell such that its distance from the center cell is increased or decreased, respectively. The transitions $c_{i,0}$ ($0 \leq i \leq N$) represent

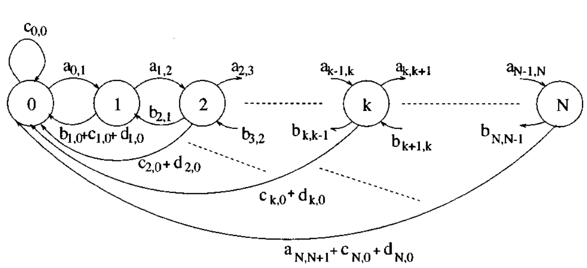

Fig. 2. State transition diagram for the imbedded Markov chain.

the arrival of incoming calls when the terminal is i cells away from the center cell. The transitions $d_{i,0}$ ($0 < i \leq N$) represent location updates performed when the mobile terminal is i cells away from its center cell.

It can be seen in eq. (15) that the location update time interval $\tau(k)$ decreases as the distance k increases. The average time at a state, k, is zero if $\tau(k) = 0$. This is because every transition into this state will result in an instantaneous transition to state 0 because of a location update. The highest state of the imbedded Markov chain, N, can be determined from the following inequalities:

$$\tau(N) > 0\,, \tag{19}$$

$$\tau(N+1) = 0\,. \tag{20}$$

The transition $a_{N,N+1}$ in Fig. 2 represents an increase of the mobile terminal's distance from N to $N+1$. This movement results in an immediately location update which changes the state of the mobile terminal from $N+1$ to 0 instantly.

Let M and T be two geometrically distributed iid random variables which represent the inter-movement time and the call interarrival time, respectively. As described before, p and λ denote the movement probability and the call arrival probability, respectively. We define γ, q, α and β as follows:

$$\gamma = 1 - \lambda\,, \tag{21}$$

$$q = 1 - p\,, \tag{22}$$

$$\alpha = q\gamma\,, \tag{23}$$

$$\beta = 1 - \alpha\,. \tag{24}$$

The probability that a movement occurs before the arrival of a call ($Pr[M < T]$) is

$$\theta = \frac{p\gamma}{1 - q\gamma}\,. \tag{25}$$

According to the ordered list of tasks performed by a mobile terminal given in section 2.2., we assume that if both a movement and a call arrival occur at the same discrete time period, the movement is performed before the call is received. This means that if a mobile terminal is at state k when both movement and call arrival occur, the next state of the mobile terminal is 0. We further assume that if either the movement time or the call arrival time coincide with the location update time, the location update will not be performed. A new location update time will be selected after the movement or the call is complete. Since the probability that more than two events occur at the same time is low when the discrete time period is small, the effect of the above assumptions to the analytical result is minimal. The state transition probabilities of the imbedded Markov chain are

For $\tau(k) = \infty$:

$$a_{k,k+1} = \begin{cases} \theta & \text{for } k = 0\,, \\ \frac{1}{2}\theta & \text{for } k > 0\,, \end{cases} \tag{26}$$

$$b_{k,k-1} = a_{k,k+1}\,, \tag{27}$$

$$c_{k,0} = 1 - \theta\,, \tag{28}$$

$$d_{k,0} = 0\,. \tag{29}$$

For $\tau(k) < \infty$:

$$a_{k,k+1} = \frac{1}{2}\Bigg\{\gamma^{\tau(k)}(1 - q^{\tau(k)}) + \lambda\left[\frac{\gamma(\gamma^{\tau(k)-1} - 1)}{\gamma - 1} - \frac{\alpha(\alpha^{\tau(k)-1} - 1)}{\alpha - 1}\right]\Bigg\}\,, \tag{30}$$

$$b_{k,k-1} = a_{k,k+1}\,, \tag{31}$$

$$c_{k,0} = q^{\tau(k)-1}(1 - \gamma^{\tau(k)}) + \frac{p}{q} \times \left[\frac{q(q^{\tau(k)-1} - 1)}{q - 1} - \frac{\alpha(\alpha^{\tau(k)-1} - 1)}{\alpha - 1}\right]\,, \tag{32}$$

$$d_{k,0} = (q\gamma)^{\tau(k)}\,. \tag{33}$$

Assume p_k is the equilibrium probability of state k. Given the above state transition probabilities, we can set up the following balance equations:

$$a_{0,1}p_0 = b_{1,0}p_1 + a_{N,N+1}p_N + \sum_{j=1}^{N}(c_{j,0} + d_{j,0})p_j \quad \text{for } N > 0\,, \tag{34}$$

$$p_k = a_{k-1,k}p_{k-1} + b_{k+1,k}p_{k+1} \quad \text{for } 0 < k < N\,, \tag{35}$$

$$p_N = a_{N-1,N}p_{N-1} \quad \text{for } N > 0\,. \tag{36}$$

Given the above balance equations, we can recursively express the equilibrium probability of each state in terms of the equilibrium probability of state N such that

$$p_k = \eta_k p_N \quad \text{for } 0 \le k < N\,, \tag{37}$$

where η_k is the ratio of the equilibrium probabilities of states k and N. The probability of state N can then be solved by applying the law of total probabilities as

$$p_N = \frac{1}{1 + \sum_{j=0}^{N-1}\eta_j} \quad \text{for } N > 0\,. \tag{38}$$

The equilibrium state probability of states k can be obtained by substituting eq. (38) into equation (37). Eq. (34) through (38) are valid only when $N > 0$. When there is only one state (i.e. $N = 0$), the equilibrium state probability of state 0 is equal to 1 ($p_0 = 1$). Assume S'_u and S'_v are the cost for location update and terminal paging per state transition, respectively. The expressions for S'_u and S'_v are then

$$S'_u = \begin{cases} Ua_{0,1} & \text{if } N = 0\,, \\ U(p_N a_{N,N+1} + \sum_{j=1}^{N} p_j d_{j,0}) & \text{if } N > 0\,; \end{cases} \tag{39}$$

$$S'_v = \begin{cases} c_{0,0}[pQ(1) + qQ(0)] & \text{if } N = 0\,, \\ p_0 c_{0,0}[pQ(1) + qQ(0)] + \sum_{j=1}^{N} p_j c_{j,0} Q(j) & \text{if } N > 0\,. \end{cases} \tag{40}$$

Let π_k be the *average sojourn time* of state k which is the smallest of three time intervals: the deterministic update time $\tau(k)$, the residual incoming call arrival time and the time until next movement. The expression for π_k is

$$\pi_k = \begin{cases} \tau(k)\alpha^{\tau(k)} + \frac{\beta}{(\alpha-1)^2}\left[\tau(k)\alpha^{\tau(k)+1} - (\tau(k)+1)\alpha^{\tau(k)} + 1\right] & \text{if } \tau(k) < \infty\,, \\ \frac{1}{\beta} & \text{if } \tau(k) = \infty\,, \end{cases} \tag{41}$$

where $\tau(k)$, α and β are computed from equations (15), (23) and (24), respectively.

The *average time between state transitions*, μ, is

$$\mu = \sum_{j=0}^{N} p_j \pi_j\,. \tag{42}$$

The average *location update, S_u, and terminal paging, S_v, costs* are

$$S_u = \frac{S'_u}{\mu}\,, \tag{43}$$

$$S_v = \frac{S'_v}{\mu}\,. \tag{44}$$

The *total location update and terminal paging cost*, denoted by S_T, is then

$$S_T = S_u + S_v\,. \tag{45}$$

5.2. Performance bounds

The performance of the proposed location update mechanism (referred to as the dynamic policy hereafter) is measured by the total location update and terminal paging cost incurred per unit time. We will compare the performance of the dynamic policy with a *no-update policy* and an *optimal policy* as described below:

No-update policy: The no-update policy is the same as the dynamic policy except that no location update is performed. When an incoming call arrives, the network searches for the mobile terminal starting from the cell where the last call arrival occurs. Since the purpose of location update is to reduce the cost for tracking down the mobile terminal, an effective location update policy should reduce the average cost as much as possible compared to the no-update policy. Otherwise, it is better off not to perform any location update.

Optimal policy: The optimal policy is defined as a distance-based location update policy such that location update is performed whenever the mobile terminal's distance from its center cell exceeds an optimal distance, ℓ^*. The optimal location update distance is selected based on the mobility and call arrival parameters such that the total location update and terminal paging cost is minimized.

Given the no-update and the optimal policies, our expectation on any mobility tracking mechanism is that the average cost should be close to that of the optimal policy while, at the same time, it must never exceed that of the no- update policy. We denote the average costs of the no-update and the optimal policies by S_T^n and S_T^*, respectively. As described in the previous subsection, the average cost of the dynamic policy, denoted by S_T, is given by eq. (45).

In order to obtain the average costs of the no-update and the optimal policies, we use the result reported in [3]. Given the location update cost, U, the polling cost, V, the call arrival probability, λ, and the movement probability, p, we developed algorithms in [3] to obtain the average cost for a distance-based location update mechanism given the location update distance, ℓ and the maximum terminal paging delay m. This average cost is denoted by $C_T(\ell, m)$ (eq. (66) in [3]). Since terminal paging delay is not constrained in this paper, we set $m = \infty$ (which means that delay is not constrained) in the following analysis. The average cost of the no-update policy, S_T^n, can be obtained when the location update distance ℓ approaches infinity:

$$S_T^n = \lim_{\ell \to \infty} C_T(\ell, \infty) . \tag{46}$$

In practice, the average cost, $C_T(\ell, \infty)$, converges to S_T^n when ℓ is sufficiently large. In the experiments given in section 5.3, we assume $S_T^n = C_T(100, \infty)$. The average cost of the optimal policy, S_T^*, is defined as

$$S_T^* = \min_{\ell} C_T(\ell, \infty) . \tag{47}$$

An iterative algorithm for determining the minimum average cost S_T^* as given in eq. (47) is reported in [3].

5.3. Experiments

We first present the numerical results using some typical parameters. Figs. 3(a), 3(b) and 3(c) show the average cost, S_T, for the location update costs, U, of 0.1, 1 and 10, respectively. The call arrival probability, λ, and the polling cost, V, are fixed at 0.01 and 1, respectively, while the probability of moving, p, varies from 0.001 to 0.5. When $U = 0.1$ (Fig. 3(a)) there is a big difference between the average cost of the optimal, S_T^*, and that of the no-update policies, S_T^n. This means that the gain from performing location update is significant. It can be seen that S_T is close to S_T^* while it is significantly lower than S_T^n. When $U = 1$ (Fig. 3(b)) the difference between S_T^* and S_T^n reduces. The values of S_T are very close to S_T^* for most movement probability, p, values. When p is close to 0.5, S_T is about half way between S_T^* and S_T^n. When $U = 10$ (Fig. 3(c)) the three curves overlap each other except when the movement probability is approaching 0.5. This means that the optimal policy is the no-update policy. The average cost of the dynamic policy is equal to that of the optimal policy except when p is high. When p is close to 0.5, S_T is slightly higher than S_T^*. Figs. 4(a), 4(b) and 4(c) show similar results when p is fixed at 0.05 while λ varies from 0.001 to 0.1. When $U = 0.1$ (Fig. 4(a)) the difference between S_T^* and S_T^n is large. The value of S_T is only

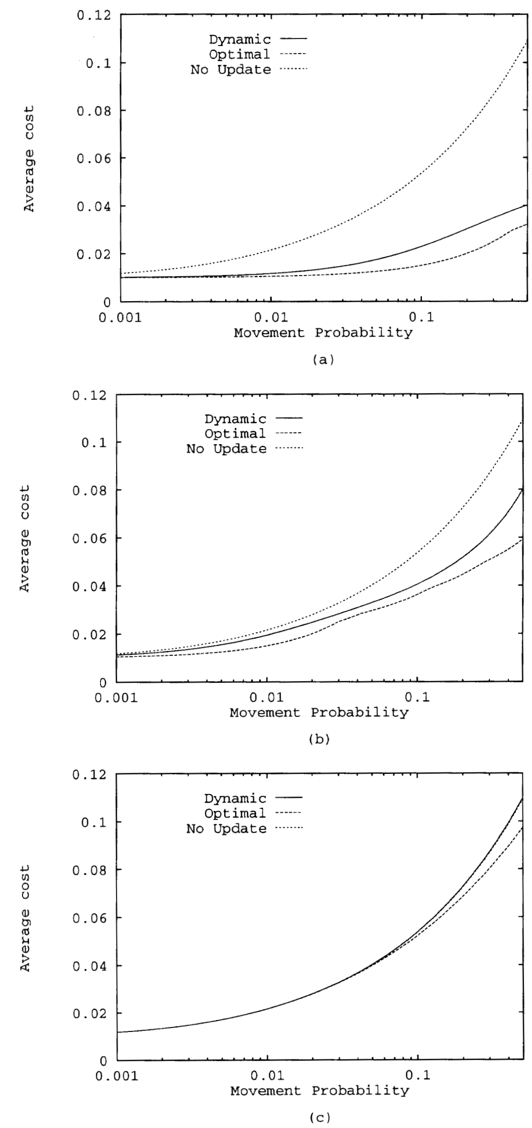

Fig. 3. Average cost vs. movement probability for (a) $U = 0.1$ (b) $U = 1$ and (c) $U = 10$.

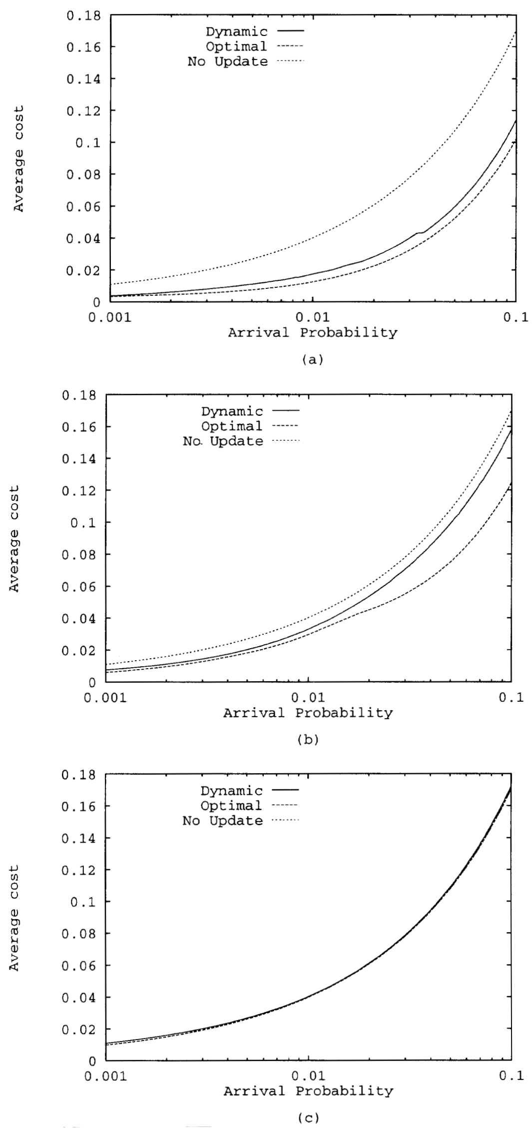

Fig. 4. Average cost vs. call arrival probability for (a) $U = 0.1$ (b) $U = 1$ and (c) $U = 10$.

slightly higher than that of S_T^* for all values of λ considered. When $U = 1$ (Fig. 4(b)), S_T is very close to S_T^* when λ is below 0.02. At high λ values, S_T increases but it is still lower than S_T^n. When $U = 10$ (Fig. 4(c)) the three curves overlap each other for all arrival probabilities considered. This means that it is better off not to perform location update because of the high location update cost U.

Fig. 5 shows the average cost as the location update cost, U, varies from 0.1 to 10. The movement probability, p, the call arrival probability, λ, and polling cost, V, are set to 0.05, 0.01 and 1, respectively. It can be seen that the average cost of the no-update policy, S_T^n, is constant for all values of U. As no location update is performed by the no-update policy, S_T^n is independent from changes in U. The average costs for both the dynamic and the optimal policies increase as U increases. For $U = 10$, the costs for all three policies are approximately the same. For other values of U, S_T is close to S_T^*, especially for $U > 1$.

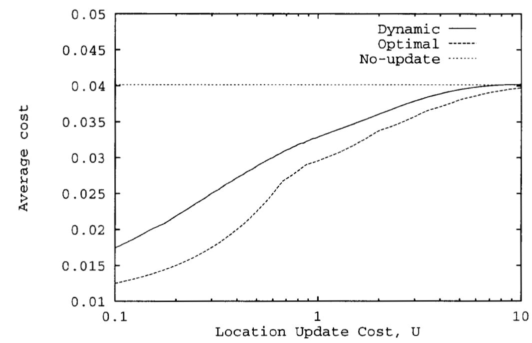

Fig. 5. Average cost vs. location update cost, U.

These three sets of experiments demonstrate the ability of the dynamic location update mechanism to adjust to changes in call arrival and movement probabilities in order to minimize cost. In most cases, the average cost of the dynamic policy is close to that of the optimal policy. While it requires significant computation to achieve the optimal cost (as demonstrated in [3,5]), the dynamic policy generates near-optimal performance at much lower computational cost. Under all the parameters considered, the average cost of the dynamic policy is always lower than that of the no-update policy.

6. Conclusions

A dynamic mobile terminal location update policy is introduced in this paper. This policy is dynamic in the sense that it can adjust to changes in system parameters, including incoming call arrival probability and the mobility pattern of the mobile terminal, so as to attain better cost effectiveness. The computational overhead of our policy is very low and the required processing is distributed over time. A decision process is initialized to generate the next location update time whenever the system parameters are changed. Moreover, the complexity of this mechanism is independent from the values of the system parameters. This results in a very stable and predictable computational power consumption. Experiments demonstrated that the dynamic policy can generate similar performance as the optimal policy given in [3]. Moreover, the dynamic policy is not restricted to a particular assumption on the mobility pattern of the mobile terminals. For ease of demonstration, we conducted the experiments using the one-dimensional mobility model. Application of our policy is not limited to this mobility model. Extension to the two-dimensional mobility model is straightforward by modifying the terminal paging cost, $Q(k)$, to reflect the polling cost required to track down a mobile terminal in the two-dimensional environment.

While all of the previous results omit the computational cost required by the location update policy, we introduced a cost efficient location update policy in this paper. The minimal computational requirement and the demonstrated performance of this policy make it a good choice of location update policy for mobile terminals with very limited processing power and energy supply.

References

[1] A. Bar-Noy and I. Kessler, Tracking Mobile Users in Wireless Networks, *Proc. IEEE INFOCOM '93* (1993) pp. 1232–1239.
[2] A. Bar-Noy and I. Kessler, Mobile users: To update or not to update? *Proc. IEEE INFOCOM '94* (1994) pp. 570–576.
[3] J.S.M. Ho and I.F. Akyildiz, A mobile user location update and paging mechanism under delay constraints, submitted for publication.
[4] Y.-B. Lin, Determining the user locations for personal communications services networks, IEEE Trans. Veh. Tech. 43 (1994) 466–473.
[5] U. Madhow, M.L. Honig and K. Steiglitz, Optimization of wireless resources for personal communications mobility tracking, *Proc. IEEE INFOCOM '94* (1994) pp. 577–584.
[6] R. Jain, Y.-B. Lin, C. Lo and S. Mohan, A caching strategy to reduce network impact of PCS, IEEE J. Select. Areas Commun. 12 (1994) 1434–1444.
[7] R. Jain and Y.-B. Lin, An auxiliary user location strategy employing forwarding pointers to reduce network impacts of PCS, Wireless Networks 1 (1995) 197–210.
[8] H. Xie, S. Tabbane and D. Goodman, Dynamic location area management and performance analysis, *Proc. IEEE VTC '93* (1993) 536–539.

Ian F. Akyildiz received his BS, MS, and PhD degrees in computer engineering from the University of Erlangen-Nuernberg, Germany, in 1978, 1981 and 1984, respectively. Currently, he is a Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology. He has held visiting professorships at the Universidad Tecnica Federico Santa Maria, Chile, Université Pierre et Marie Curie (Paris VI) and Ecole Nationale Superieure Telecommunications in Paris, France. He has published over hundred technical papers in journals and conference proceedings. He is an editor for "Computer Networks and ISDN Systems Journal", "IEEE Transactions on Computers", and "ACM-Baltzer Journal of Wireless Networks".

He guest-edited several special issues, such as on "Parallel and Distributed Simulation Performance" for "ACM Transactions on Modeling and Simulation"; and on "Networks in the Metropolitan Area" for "IEEE Journal of Selected Areas in Communications".

Dr. Akyildiz is a senior member of IEEE since 1989, member of ACM (SIGCOMM) and is a National Lecturer for ACM since 1989. He received the "Don Federico Santa Maria Medal" for his services to the Universidad Federico Santa Maria in Chile. Dr. Akyildiz is listed in "Who's Who in the World (Platinum Edition)". He received the ACM Outstanding Distinguished Lecturer Award for 1994.

His current research interests are in B-ISDN (ATM) networks, wireless networks, optical local area networks, performance evaluation, parallel simulation, high performance computing and network security.
E-mail: ian@armani.gatech.edu

Joseph S.M. Ho received the B.S.E.E. and M.S.E.E. degrees from the University of Washington at Seattle in 1987 and 1989, respectively. He is currently a Ph.D. student in the School of Electrical and Computer Engineering at Georgia Institute of Technology.

His current research interests include design and analysis of PCS networks, performance evaluation and parallel simulation. He is a student member of IEEE and Tau Beta Pi.
E-mail: joseph@ee.gatech.edu