ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®

Computer Networks xxx (2005) xxx–xxx

Computer Networks

www.elsevier.com/locate/comnet

A cost-minimization algorithm for fast location tracking in mobile wireless networks

Wenye Wang [a,*], Guoliang Xue [b]

[a] Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695, United States
[b] Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287-8809, United States

Received 10 March 2005; received in revised form 9 August 2005; accepted 26 September 2005

Responsible Editor: B. Baykal

Abstract

Location tracking is one of the most important issues in providing real-time applications over wireless networks due to its effect to quality of service (QoS), such as end-to-end delay, bandwidth utilization, and connection dropping probability. In this paper, we study cost minimization for locating mobile users under delay constraints in mobile wireless networks. Specifically, a new location tracking algorithm is developed to determine the position of mobile terminals under delay constraints, while minimizing the average locating cost based on a unimodal property. We demonstrate that the new algorithm not only results in minimum locating cost, but also has a lower computational complexity compared to existing algorithms. Furthermore, detailed searching procedures are discussed under both deterministic and statistic delay bounds. Numerical results for a variety of location probability distributions show that our algorithm compares favorably with existing algorithms.

Keywords: Wireless networks; Location tracking; Partition; Optimization

1. Introduction

The increasing demand for diverse mobile applications using public wireless networks has imposed many challenging issues because of the variations in mobile users' positions from time to time [14]. In order to deliver services in wireless networks, fast location tracking is critical to offering real-time mobile services, such as voice over IP. In cellular wireless networks, location update and paging are two fundamental operations for locating a mobile terminal (MT). According to the latest specification on third generation wireless communication systems such as universal mobile telecommunication system (UMTS) [1,9,23], location update depends on the design of location areas (LAs) and routing areas (RAs). Each LA consists of a group of cells, and mobile terminals send location update requests when they cross the boundary of two LAs. The RA is designed for packet switching domain, and each RA can be a subset of the location area. In

* Corresponding author. Tel.: +1 919 513 2549; fax: + 1 919 515 5523.
E-mail addresses: wwang@ncsu.edu (W. Wang), xue@asu.edu (G. Xue).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2005.09.035

other words, mobile terminals update their location information with the system based on location management mechanisms.

On the opposite, paging is a process to locate MTs, which is performed by the system instead of end users [16,19]. The main challenge in locating MTs is to reduce the signaling cost under delay constraints. Therefore, the minimization of cost and delay caused by locating mobile objects has been studied extensively by researchers [2,7,11,15–17,20,24]. In future wireless networks, many applications of multimedia services have various quality of service (QoS) requirements, including delay, transmission rate, pricing models and so on.

Among these parameters, delay is one of the most important metrics because it is directly related to the perceived QoS and is used to differentiate real-time and non-real-time applications. Therefore, traditional broadcast paging scheme used for telephony systems, in which polling messages are sent to every cell in the LA, is not appropriate for dual services in circuit-switched and packet-switched domains. Under this broadcast paging scheme, the paging delay is minimized since there is only one *polling cycle* required to find the called MT and all cells within the LA receive the paging request simultaneously, where a polling cycle is the round trip time from when a paging message is sent until the response is received. However, the cost of this paging scheme is high and the utilization of bandwidth is low since all cells in the LA are searched, which consumes a large amount of down-link radio resources for high mobility users.

As the demand for wireless services such as emails, transactions, and web-browse grows rapidly, the signaling traffic caused by location tracking increases accordingly, which consumes limited available radio resources. To improve the efficiency of bandwidth utilization, we explore the optimization of location tracking cost under delay constraints, based on a time-varying probability distribution of user location [5,13,15,22]. The probability distribution of user location depends on many factors such as mobility model, calling pattern, and so on. Many tracking schemes are designed to predict cell location probabilities and to estimate the next location of a MT accurately [2,3,6,8,10,18]. In this paper, we focus on optimal partitioning of searching areas.

The minimization of location tracking cost with delay constraints induces two fundamental problems:

1. Given the probability distribution and a deterministic delay bound, what is the minimum cost required to locate the target object? If there is such an optimal solution, how to design paging areas and how to proceed the searching procedure? What is the computation complexity for finding an optimal solution?
2. Given the probability distribution and a statistic delay constraint, what is the minimum cost required to locate the moving terminal? How can the statistic delay constraint be satisfied?

These two problems are challenging because they require the optimal solution to achieve minimum cost under delay constraints, whereas the computation complexity must be taken into account. Previous efforts have addressed these problems to some extent. For example, in [4,12,16], it is demonstrated that the minimum cost can be obtained if all cells are searched in a decreasing order of location probabilities in the absence of delay constraints. Similar results also show that the minimum cost can be achieved through dynamic programming. Since the number of ways to partition an N cell location area into $\mathscr{D}$ paging areas is exponential in N, searching through all possible partitions is an unrealistic approach to finding a cost-minimum scheme. In [20], it is proved that three necessary conditions are required to achieve minimum cost given a deterministic delay bound, thus, reducing the computation complexity. It is required that all cells must be searched in a non-increasing order of their location probabilities. This important property makes it sufficient to search only $O(N^{\mathscr{D}-1})$ partitions. Since it takes $O(N)$ time to compute the paging cost corresponding to a given partition, the necessary condition of [20] immediately implies an $O(N^{\mathscr{D}})$ time algorithm for computing an optimal paging scheme under delay bound $\mathscr{D}$.

In this paper, we prove a unimodal property of two-step locating cost as a function of the corresponding 2-partition of the location area. This unimodal property enables us to find an optimal 2-partition in $O(\log N)$ time, given $O(N \log N)$ preprocessing time. As a result, we have an $O(N \log N + N^{\mathscr{D}-2} \log N)$ time algorithm for computing a cost-minimum location tracking under delay constraint $\mathscr{D}$. Moreover, we investigate the cost minimization issue under statistic delay constraints. Based on the optimal algorithm developed for deterministic delay bound, we tackle this problem through a sequential matching algorithm. Con-

sidering multiple locating algorithms may cause failure in the searching process, we also analyze locating failure which is affected by the proposed algorithm.

Throughout the paper, the proposed algorithms and procedure are illustrated in the context of cellular networks. However, they are applicable to other mobile communication systems as well. For example, a hot-spot system such as a wireless local area network (WLAN) may need to locate a laptop which sends requests of services so as to deliver data or video to this terminal. Meanwhile, the locating algorithm can be used to assist storing information in those proxy servers which are close to the requesting terminals. Especially, if mobile users change their positions after they send a web-browsing or message request, locating these terminals is mandatory, but the delay constraints are flexible for non-real-time applications.

The rest of the paper is organized as follows. In Section 2, we formulate the problem of location tracking and discuss necessary conditions for cost minimization. In Section 3, we prove a unimodal property of two-step searching and use that property to design a fast cost-minimization algorithm subject to deterministic delay bounds. In Section 4, we present location tracking procedure under statistic delay bounds. We evaluate the performance of the proposed algorithm in terms of locating cost, delay, computation complexity, and searching failure in Section 5. We present numerical results over various location probability distributions in Section 6 and conclude the paper in Section 7.

2. Problem formulation

We assume that each LA (or RA) in a wireless system consists of the same number of cells, N. The worst-case delay of a successful searching is considered as the delay bound, $\mathscr{D}$, that is, the number of polling cycles. For instance, if $\mathscr{D}$ is equal to 1, the system should find the called MT in one polling cycle, requiring all cells within the LA to be polled simultaneously. The locating cost, C, which is the number of cells polled to find the called MT, is equal to N. In this case, the average locating delay is at its least value, which is one polling cycle, and the searching or paging cost is at its highest value, $C = N$. On the contrary, when $\mathscr{D}$ is equal to N, the system will poll one cell in each polling cycle and search all cells one by one until the called MT is found. Thus, the worst-case occurs when the called MT is found in the last polling cycle, which means the searching delay would be at its maximum and equal to N polling cycles. However, the locating cost may be minimized if the cells are searched in a decreasing order of location probabilities as demonstrated in [5,16].

We consider the partition of an LA given that $1 < \mathscr{D} < N$, which requires grouping cells within an LA into smaller searching areas (SAs) under delay bound $\mathscr{D}$. The initial state $\mathbf{P}$ is defined as $\mathbf{P} = [p_1, p_2, \ldots, p_j, \ldots, p_N]$, where p_j is the probability of jth cell to be searched in decreasing order of probability. We use triplets SA*(i, q_i, n_i) to denote the SAs, in which i is the sequence number of the SA, q_i is the probability of the called MT being found within the ith SA, and n_i is the number of cells contained in this SA. Accordingly, the *location probability* q_i of the ith SA is

$$q_i = \sum_{j \in i\text{th SA}} p_j. \tag{2.1}$$

The *locating cost* under delay bound $\mathscr{D}$, $E[C(\mathscr{D})]$, is computed as follows:

$$E[C(\mathscr{D})] = \sum_{i=1}^{\mathscr{D}} q_i \cdot k_i, \quad \text{where } k_i = \sum_{k=1}^{i} n_k \tag{2.2}$$

and the *average delay*, $E[D(\mathscr{D})]$, is

$$E[D(\mathscr{D})] = \sum_{i=1}^{\mathscr{D}} i \cdot q_i. \tag{2.3}$$

Therefore, the cost-minimization problems for location tracking can be formulated as follows:

- The problem of optimal partition under deterministic delay constraint (PDDC) has an input parameter $\mathscr{D} \in \{1, 2, \ldots, N\}$ and asks for a partition of the LA into $\mathscr{D}$ SAs so that the locating cost is minimized.
- The problem of optimal partition under statistic delay constraint (PSDC) has an input parameter $d \in [1, N]$ and asks for an integer $\mathscr{D} \in \{1, 2, \ldots, N\}$ together with a partition of the LA into $\mathscr{D}$ SAs so that the locating cost is minimized among all such partitions with an average delay no more than d.

In [20], the following *necessary conditions* are proved for an optimal partition with minimum locating cost.

Lemma 2.1. *If a searching sequence $\mathscr{P}$ satisfies the following conditions of partition, the locating cost $E[C(\mathscr{D})]$ can be minimized*:

1. *Probability condition*: *The cells must be searched in a decreasing order of location probability. In other words, if u and v are cells with $p_u > p_v$, then the optimal searching sequence $SA_{\mathscr{P}}$ that minimizes $E[C(\mathscr{D})]$ must satisfy $u \in PA_{\mathscr{P}}(g, q_g, n_g)$ and $v \in PA_{\mathscr{P}}(h, q_h, n_h)$ for all $g \leqslant h$.*
2. *Forward boundary condition*: *It determines the largest location probability cell in the SA. We denote p^{l}_{i+1} as the largest probability cell in the $(i+1)$th searching area with n_{i+1} cells. Then, $p^{\mathrm{l}}_{i+1} \cdot (n_{i+1} - 1)$ must be less than or equal to q_i.*
3. *Backward boundary condition*: *It chooses the smallest location probability cell in the SA. The backward boundary condition demands that q_i should be less than or equal to $p^{\mathrm{s}}_i \cdot (n_{i+1} + 1)$, where p^{s}_i is the smallest probability in the ith searching area, and n_{i+1} is the number of cells in the $(i+1)$th SA. Thus, the smallest probability cell p^{s}_i cannot be moved "backward" to the $(i+1)$th SA, which comes after the ith SA.*

Similarly, we have the following facts.

Lemma 2.2. *There exists an optimal partition under statistic delay constraint d such that the cells are searched in a decreasing order of location probability. In other words, if u and v are cells with $p_u > p_v$, then the optimal searching sequence $SA_{\mathscr{P}}$ that minimizes $E[C(\mathscr{D})]$ subject to $E[D(\mathscr{D})] \leqslant d$ must search the SA containing u before the SA containing v, if u and v are in different SAs.*

Proof. If u and v are in the same SA, there is nothing to be proved. In the rest, we will assume that u and v are in different SAs. Since changing the order of two cells in the same SA does not affect the locating cost and the average delay, we may assume that the cells in each SA are ordered in a decreasing order of their location probabilities.

Suppose that SA^{L} and SA^{R} are two SAs such that the optimal partition method searches SA^{L} before SA^{R}. Assume that SA^{L} contains cells $c^{\mathrm{L}}_1, \ldots, c^{\mathrm{L}}_m$ with locating probabilities $p^{\mathrm{L}}_1 \geqslant \cdots \geqslant p^{\mathrm{L}}_m$. Assume that SA^{R} contains cells $c^{\mathrm{R}}_1, \ldots, c^{\mathrm{R}}_n$ with locating probabilities $p^{\mathrm{R}}_1 \geqslant \cdots \geqslant p^{\mathrm{R}}_n$. Suppose that $p^{\mathrm{L}}_m < p^{\mathrm{R}}_1$. Swapping c^{L}_m with c^{R}_1, we will get a different partition. It follows from the theory of [20] that the new partition will have a lower locating cost. From the definition of average delay, we know that the average delay of the new partition is also smaller than the average delay of the previous partition (by $p^{\mathrm{R}}_1 - p^{\mathrm{L}}_m$). This proves the lemma. □

Lemma 2.3. *Let $C_D(\mathscr{D})$ be the locating cost of the minimum cost paging scheme subject to deterministic delay bound $\mathscr{D}$ for $\mathscr{D} = 1, 2, \ldots, N$. Then $C_D(\mathscr{D})$ is a monotonically non-increasing function of $\mathscr{D}$.*

Proof. Suppose that $\mathscr{D} < N$ and that $C_D(\mathscr{D})$ is achieved by partitioning the LA into a number of $\mathscr{D}$ SAs, such as $\mathrm{SA}_1, \mathrm{SA}_2, \ldots, \mathrm{SA}_{\mathscr{D}}$, where the cells are paged with decreasing locating probability. Since $\mathscr{D}$ is smaller than N, there exists an SA, SA_i, which contains cells $c^i_1, c^i_2, \ldots, c^i_k$ for some $k \geqslant 2$ with locating probabilities $p(c^i_1) \geqslant p(c^i_2) \geqslant \cdots \geqslant p(c^i_k)$. If we replace SA_i with two SAs, one containing c^i_1 and the other containing cells c^i_2 through c^i_k, we will have a total of $\mathscr{D} + 1$ ways of partition whose corresponding paging cost is $C_D(\mathscr{D}) - p(c^i_1) \leqslant C_D(\mathscr{D})$. This proves the lemma. □

Lemma 2.4. *Let $C_S(d)$ be the locating cost of the minimum cost paging scheme subject to statistic delay bound d by partitioning the LA into SAs. Then $C_S(d)$ is a monotonically non-increasing function of d.*

Proof. Let $d_1 < d_2$ be two statistic delay bounds. Since there is a paging method whose average delay is no more than d_1 and whose locating cost is equal to $C_S(d_1)$, there is a paging method whose average delay is no more than d_2 and whose locating cost is at most $C_S(d_1)$. This proves the lemma. □

In the following section, we will present an $O(N \log N + N^{\mathscr{D}-2} \log N)$ time algorithm for computing an optimal partition. Our result is based on a *unimodal property* and a *necessary and sufficient condition* for an optimal 2-partition.

3. Optimal partition under deterministic delay constraints

3.1. Optimal partition into two SAs

In this section, we will focus on the case where $\mathscr{D} = 2$. For any integer $n \in \{1, 2, \ldots, N-1\}$, we can partition N cells into SA^n_1 and SA^n_2 so that SA^n_1 contains cells 1 through n (the first n cells) and SA^n_2 contains cells $n+1$ through N (the last $N-n$ cells). It follows from the *probability condition* of

Lemma 2.1 that for *some* integer n, SA_1^n and SA_2^n form an optimal partition.

Let $f(n) = n \cdot \sum_{i=1}^{n} p_i + N \cdot \sum_{i=n+1}^{N} p_i$. Then $f(n)$ is the expected cost of the corresponding partition. The following fact is important.

Lemma 3.1. *Assume* $1 \leqslant n \leqslant N-2$. *We have*

- *If* $f(n) \leqslant f(n+2)$, *then* $f(n+1) < f(n+2)$.
- *If* $f(n) \geqslant f(n+2)$, *then* $f(n+1) < f(n)$.

Proof. Let

$$\mathscr{X} = \sum_{i=1}^{n} p_i + (n+1-N) \cdot p_{n+1} \tag{3.4}$$

and

$$\mathscr{Y} = \sum_{i=1}^{n} p_i + (n+1-N) \cdot p_{n+2} + p_{n+1} + p_{n+2}. \tag{3.5}$$

Since $N \geqslant n+2$ and $p_{n+1} \geqslant p_{n+2} > 0$, we have

$$\mathscr{Y} \geqslant \mathscr{X} + p_{n+1} + p_{n+2} > \mathscr{X}. \tag{3.6}$$

By definition, we have:

$$\begin{aligned} f(n+1) &= (n+1) \cdot \sum_{i=1}^{n+1} p_i + N \cdot \sum_{i=n+2}^{N} p_i \\ &= f(n) + \mathscr{X}, \end{aligned} \tag{3.7}$$

$$\begin{aligned} f(n+2) &= (n+2) \cdot \sum_{i=1}^{n+2} p_i + N \cdot \sum_{i=n+3}^{N} p_i \\ &= f(n) + \mathscr{X} + \mathscr{Y}. \end{aligned} \tag{3.8}$$

When $f(n) \leqslant f(n+2)$, we have $\mathscr{X} + \mathscr{Y} \geqslant 0$. It follows from (3.6) that $\mathscr{Y} > 0$. This implies that $f(n+1) = f(n) + \mathscr{X} < f(n) + \mathscr{X} + \mathscr{Y} = f(n+2)$. For $f(n) \geqslant f(n+2)$, we have $\mathscr{X} + \mathscr{Y} \leqslant 0$. In the same way aforementioned, $\mathscr{X} < 0$, which means $f(n+1) = f(n) + \mathscr{X} < f(n)$. This completes the proof. □

Lemma 3.2. [*unimodal property*] *Assume that* $1 \leqslant i < j < k \leqslant N$. *Then we have*

$$f(j) < \max\{f(i), f(k)\}. \tag{3.9}$$

Proof. If $i = j-1$ and $k = j+1$, it follows from Lemma 3.1 that

$$\begin{aligned} f(j) &< \max\{f(i), f(k)\} \\ &= \max\{f(j-1), f(j+1)\}. \end{aligned} \tag{3.10}$$

Assume that $f(j) < f(j+1)$. Applying Lemma 3.1 repeatedly, we can get $f(j) < f(j+1) < f(j+2) < \cdots < f(N)$. Therefore $f(j) < f(j+1)$ implies $f(j) < f(k)$.

Similarly, $f(j-1) > f(j)$ implies $f(1) > f(2) > \cdots f(j-1) > f(j)$, which in turn implies $f(j) < f(i)$. This proves the lemma. □

Theorem 3.1. *With delay bound* $\mathscr{D} = 2$, *there are at most two integers* $n \in \{1,2,\ldots,N-1\}$ *such that* $SA_1^n = \{1,2,\ldots,n\}$ *and* $SA_2^n = \{n+1, n+2, \ldots, N\}$ *form an optimal partition.*

1. *If* $f(j) = f(j+1)$ *for some* j, *then there are exactly two optimal partitions. They are defined by* $SA_1^n = \{1,2,\ldots,n\}$ *and* $SA_2^n = \{n+1, n+2, \ldots, N\}$ *for* $n = j$ *and* $n = j+1$.
2. *If* $f(j) < f(j+1)$ *for some* j, *then* $SA_1^n = \{1,2,\ldots,n\}$ *and* $SA_2^n = \{n+1, n+2, \ldots, N\}$ *being an optimal partition implies* $n \leqslant j$.
3. *If* $f(j) > f(j+1)$ *for some* j, *then* $SA_1^n = \{1, 2, \ldots, n\}$ *and* $SA_2^n = \{n+1, n+2, \ldots, N\}$ *being an optimal partition implies* $n \geqslant j+1$.

Proof. If $f(j) = f(j+1)$, it follows from Lemma 3.2 that $f(i) > f(j)$ for any $1 \leqslant i < j$ and $f(k) > f(j+1)$ for any $j+1 < k \leqslant N$. Therefore, for either $n = j$ or $n = j+1$, SA_1^n and SA_2^n form an optimal partition.

If $f(j) < f(j+1)$, it follows from Lemma 3.2 that $f(j) < f(k)$ for any $j < k \leqslant N$. Therefore whenever $SA_1^n = \{1,2,\ldots,n\}$ and $SA_2^n = \{n+1, n+2, \ldots, N\}$ form an optimal partition, we must have $n \leqslant j$.

If $f(j) > f(j+1)$, it follows from Lemma 3.2 that $f(i) > f(j+1)$ for any $1 \leqslant i < j+1$. Therefore whenever $SA_1^n = \{1,2,\ldots,n\}$ and $SA_2^n = \{n+1, n+2, \ldots, N\}$ form an optimal partition, we must have $n \geqslant j+1$. □

Corollary 3.1. *When* $\mathscr{D} = 2$, *the necessary conditions of Lemma 2.1 (probability condition, forward condition, backward condition) are also sufficient for an optimal partition.*

Proof. Assume that $SA_1^n = \{1,2,\ldots,n\}$ and $SA_2^n = \{n+1, n+2, \ldots, N\}$ satisfy all three conditions of Lemma 2.1. It follows from the proof of Lemma 2.1 [20] that $f(n)$ is not larger than $f(n-1)$ or $f(n+1)$. It follows from Theorem 3.1 that $SA_1^n = \{1,2,\ldots,n\}$ and $SA_2^n = \{n+1, n+2, \ldots, N\}$ form an optimal partition. □

Based on the unimodal property, an optimal 2-partition of an N-cell location area can be computed in $O(\log N)$ time, after $O(N \log N)$ preprocessing time. This is listed as Algorithm 1.

Theorem 3.2. *Algorithm part2* $(p, s, L, H, index)$ *correctly computes an integer index so that cells* $L+1$ *through index* -1 *and cells index through* H *form an optimal partition of the cells from* $L+1$ *through* H, *with the corresponding cost value returned. The worst case running time of the algorithm is* $O((H-L)\log(H-L))$. *In particular, part2* $(p, s, 0, N, n)$ *computes an integer n such that* SA_1^n *and* SA_2^n *form an optimal partition of the cells* 1 *through N.*

Proof. The correctness of the algorithm follows from Theorem 3.1 and Lemma 3.2.

step_1 takes $O((H-L)\log(H-L))$ time. step_2 takes $O((H-L))$ time. step_3 takes $O(1)$ time. step_4 takes $O(\log(H-L))$ time. Therefore the overall running time is $O((H-L)\log(H-L))$. □

Algorithm 1. double part2 (p, s, L, H, index)

Input: $p_1 \geqslant p_2 \geqslant \cdots \geqslant p_N$ and $s_i = \sum_{j=1}^{i} p_j$ for $i = 1,2,\ldots,N$. $s_0 = 0$. $0 \leqslant L < L+1 < H \leqslant N$.
Output: index is set so that $\{L+1,\ldots,\text{index}\}$ and $\{\text{index}+1,\ldots,H\}$ form an optimal partition of the cells $\{L+1,\ldots,H\}$. The corresponding cost is returned.step_1 $LB := L$; $UB := H$;
step_2 **while** $UB > LB + 1$ **do**
$j = (LB + UB)/2$;
$f(j) = (j-L)\cdot(s_j - s_L) + (H-L)\cdot(s_H - s_j)$;
$f(j+1) = (j+1-L)\cdot(s_{j+1} - s_L) + (H-L)\cdot(s_H - s_{j+1})$;
if case 1 of Theorem 3.1 happens **then**
index $:= j$; **return** $f(j)$;
elseif case 2 of Theorem 3.1 happens **then**
$UB := j$; **goto** step_4;
elseif case 3 of Theorem 3.1 happens **then**
$LB := j$; **goto** step_4;
endif
endwhile

3.2. Optimal partition into $\mathscr{D}$ SAs

In this section, we will use part2 as a subroutine in the development of an optimal partition algorithm and any $\mathscr{D}$. There are $O(N^{\mathscr{D}-2})$ choices of the first $\mathscr{D}-2$ groups. For each such choice, we can use $O(\log N)$ time to compute the optimal partition of the last two location groups subject to this setting. A total of $O(N \log N)$ preprocessing time is required for sorting and for computing prefix sums of the probabilities.

Theorem 3.3. *Algorithm* part $\mathscr{D}$ *correctly computes an optimal* $\mathscr{D}$-partition *of the cells* $\{L+1,\ldots,H\}$. *The time complexity of algorithm is* $O(N \log N + N^{\mathscr{D}-2}\log N)$.

Algorithm 2. double part$\mathscr{D}$(p, s, L, H, index)

Input: $p_1 \geqslant p_2 \geqslant \cdots \geqslant p_N$ and $s_i = \sum_{j=1}^{i} p_j$ for $i = 1,2,\ldots,N$. $s_0 = 0$. $0 \leqslant L < L + \mathscr{D} - 1 < H \leqslant N$.
Output: The $\mathscr{D}-1$ element array index is set so that $\{L+1,\ldots,\text{index}[1]\}$, $\{\text{index}[1]+1,\ldots,\text{index}[2]\},\ldots,\{\text{index}[\mathscr{D}-1]+1,\ldots,H\}$ form an optimal partition of the cells $\{L+1,\ldots,H\}$. The corresponding cost is returned.
step_1 **for** each $L < \text{index}[1] < \text{index}[2] < \cdots < \text{index}[\mathscr{D}-2] < H-2$ **do** compute the corresponding cost as $\sum_{i=1}^{\mathscr{D}-2}(\text{index}[i] - \text{index}[i-1])\times(s_i - s_{i-1}) + \text{part2}(p, s, \text{index}[\mathscr{D}-2], H, \text{index}[\mathscr{D}-1])$;
step_2 Set the array index to be the one with the minimum corresponding cost.
return the corresponding minimum cost.

Proof. The correctness of the algorithm follows from that of Algorithm 1. We need to compute the cost for $O(N^{\mathscr{D}-2})$ times, spending $O(\log N)$ time for each such computation. $O(N \log N)$ is spent on preprocessing. Therefore the worst-case running time of part$\mathscr{D}$ is $O(N \log N + N^{\mathscr{D}-2}\log N)$. □

4. Optimal partition algorithm under statistic delay constraint

Based on the optimal partition algorithm for deterministic delay bound, in this section, we tackle the cost-minimization problem under the constraints of statistic delay.

4.1. Location tracking algorithm under statistic delay constraint

Statistic delay constraint can be considered as an average delay during service delivery. The problem of minimizing locating costs under statistic delay constraints is a difficult one. It may be mathematically complicated to provide a concrete solution.

However, we can tackle this problem by considering the intrinsic relationship between the average delay and upper delay bound. The basic idea is to find a corresponding upper delay bound given a statistic delay constraint. Then we can obtain an optimal partition based on the algorithm developed in Section 3.

Assume a set of delay bounds is denoted as **D**, and the cardinality of this set is N, which is the maximum number of cells in an LA. Each specific delay bound, $\mathscr{D} \in \mathbf{D}$, results in an average searching delay $D(\mathscr{D})$. We also denote **d** as the set of statistic delay constraints and each statistic delay constraint, $d \in \mathbf{d}$. Thus, upper delay bound, $\mathscr{D}$, is distinguished from statistic delay constraint d. Note that it is possible that we can find a delay bound $\mathscr{D} \in \mathbf{D}$ corresponding to multiple statistic delay $d \in \mathbf{d}$ as shown in Fig. 1, depending on how statistic delays are specified in mobile services.

Therefore, the minimization of locating costs with statistic delay constraint can be formulated as finding the matching delay bound to a specific statistic delay constraint. Then, the optimal partition algorithm developed in Section 3 can be deployed in determining the final partitions for optimal searching procedure under the constraints of average searching delay.

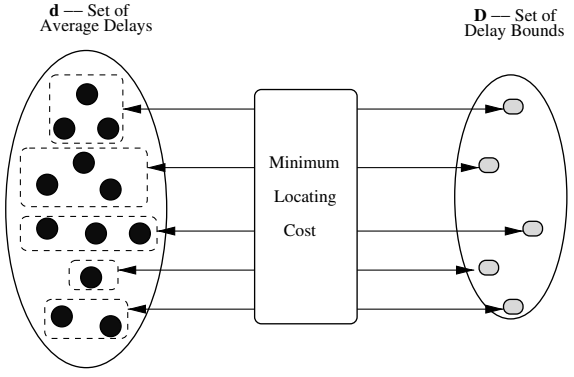

Fig. 1. Deterministic delay and statistic delay.

Our locating procedure under statistic delay constraint is as follows:

1. Determine the range of average delays, $R_d = [1, \mathbf{d}_{\max}]$, with upper delay bounds varying from 1 to N, i.e., $\mathscr{D} = 1, 2, \ldots, N$.
2. Obtain the set of average searching delays and costs according to upper delay bounds by using the optimal partition algorithm in Section 3.
3. Match the requirement of average delay, $d \in \mathbf{d}$ to a specific delay bound $\mathscr{D} \in \mathbf{D}$.
4. Find the partition of searching areas with corresponding upper delay bound, $\mathscr{D}$; thus, the average locating cost with average delay constraint $d(\mathscr{D})$ can be obtained.

If there are more than one delay bound which result in average delays that satisfy the requirement of statistic constraint, by default, the one that produces less locating costs will be preferred because the objective of the optimal locating is to reduce the overhead of locating process.

For example, considering an example as in Fig. 2. First, we determine the range of average delays with varying delay bounds, i.e., $R_d = [1,3.00]$. This means the maximum average delay of location tracking for this example is 3.00 polling cycles with the optimal partition algorithm. On the other hand, the service requirements, based on the range of average delay obtained from R_d, are represented by the average searching delays, which can be specified as $\mathbf{d} = [1.0,1.1,1.2,1.3,\ldots,1.9,2.0,2.1,\ldots,2.9,3.0]$. This is the second row shown in Fig. 2. Next, we match the elements in **d** with $\mathbf{D} = [1,2,\ldots,10]$. Then the

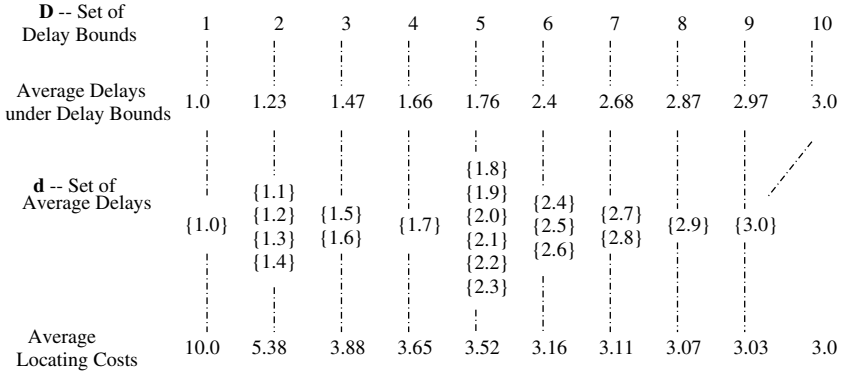

Fig. 2. Example of location tracking under average delay constraint.

locating cost for each delay constraint can be obtained as in the last row in Fig. 2.

4.2. Sequential matching algorithm

Here, we introduce a systematic method to accomplish the procedure explained in the previous section, namely, *Sequential matching*. This is used to match the average delay with the delay bound. The framework of the sequential matching algorithm starts from the lowest delay constraint in set **d**. The flowchart of this matching procedure is shown in Fig. 3. The idea is to compare the delay constraint with the *possible* average delays. Since the average delay may be derived from service-level agreements or quality of service specification, the required average delay may not be the exact value that can be obtained by any locating algorithm. Therefore, we need to *satisfy* the delay requirements as close as possible. If the average searching delay meets the delay constraint, then the corresponding delay bound is determined, which further yields the minimum locating cost. This procedure continues until all elements in delay constraint set **d** are examined and categorized into specific delay bounds. Therefore, the optimal partition can be completed based on the algorithm with deterministic delay bound proposed in Section 3.

5. Performance evaluation

The performance of the proposed *Fast* location tracking algorithm for *Cost-Minimization*, denoted as *FCM* in short, will be evaluated by following parameters:

- Average locating cost and average delay of searching: Once a location area is partitioned into SAs and searching sequence is determined based on a location tracking algorithm, the system is able to send paging signals to each SA one by one in an order of the searching sequence. The average locating cost and searching delay can be obtained by using formulas in (2.2) and (2.3) for any given delay bound.

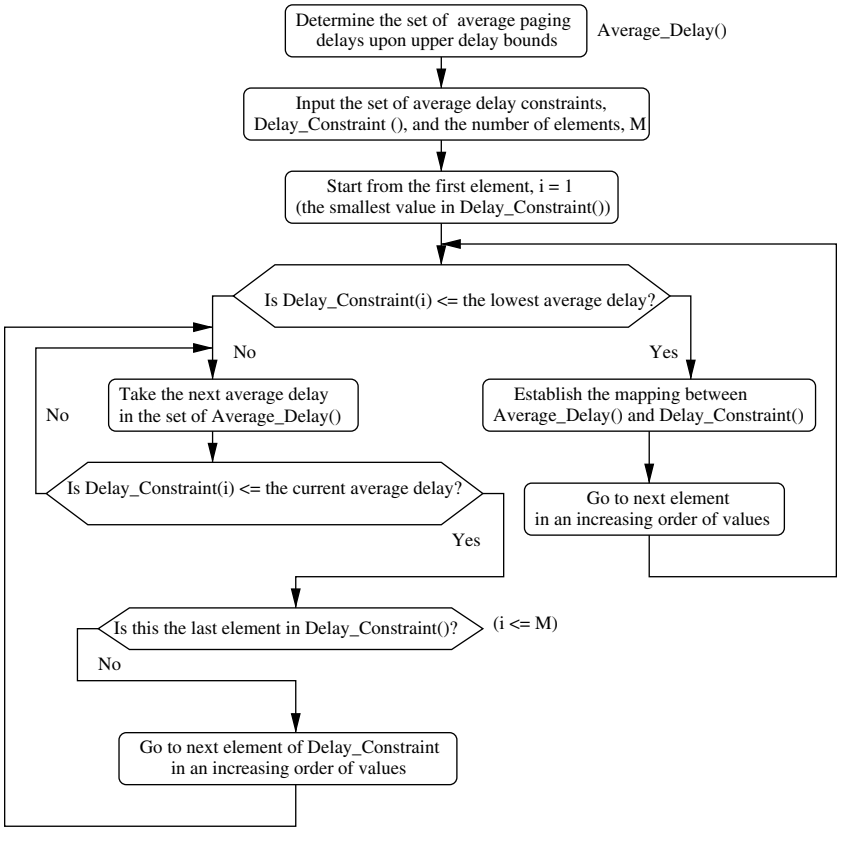

Fig. 3. Sequential matching algorithm.

- Computation complexity: We are interested in comparing the computation complexity of three cases: without any partition algorithm, with only necessary conditions, and with fast locating algorithm, which are necessary and sufficient.

Without using the probability condition, one needs to consider all possible partitions of the LA into $\mathscr{D}$ SAs. The number of such partitions is the Stirling number of the second kind [21]. By taking the searching order into consideration, we have

$$D! \times S(n,D) = \sum_{i=0}^{D-1} (-1)^i \binom{D}{i} (D-i)^n. \qquad (5.11)$$

If we evaluate all possible partitions satisfying the probability condition, the time complexity would be $O(N \log N + N^D)$, since there are $O(N^{D-1})$ partitions to evaluate and evaluation takes $O(N)$ time.

The time complexity of the necessary conditions only algorithm is $O(N^D)$, whereas our fast locating algorithm is $O(N \log N + N^{D-2} \log N)$.

6. Numerical results

The numerical results of the proposed *fast cost-minimization* (FCM) algorithm with regards to uniform distribution, truncated discrete Gaussian distribution, and irregular distribution are presented in this section. We compare the average locating cost and searching delay with highest probability first (HPF) and paging with necessary conditions only algorithms (NOA) [16,20]. It is necessary to mention that in the original paper of the HPF scheme, a non-increasing probability density function $g(x)$ must be found for a non-increasing discrete distribution of location probabilities [16]. In order to obtain the numerical results and make complete comparison, we design the paging procedure of the original HPF scheme, which is called the enhanced-HPF (E-HPF) scheme. In addition, we evaluate the computation complexity of different partition algorithms with regards to delay bounds and number of cells.

Although location probabilities are given as a prior information in this paper, they can be obtained through network simulation experiments [3] in which network scenarios, buffers, and computation complexity are provided. These location probabilities can then be used as input to our tracking algorithms. In order to avoid redundancy, we focus on the numerical results in this section.

6.1. Locating cost and delay for uniform probability distribution

First we study the relationship between the average locating cost, C, deterministic delay bound, $\mathscr{D}$, and statistic delay bound, d, under different searching schemes. Fig. 4(a) shows the average searching cost $C(\mathscr{D})$ as a function of $\mathscr{D}$ for an LA with 10 cells ($N = 10$). It can be seen that the locating costs decrease with the increasing delay bounds for three partition algorithms, except broadcast scheme. As a matter of fact, we can observe only one plot in the figure because the three plots overlap, which means the fast tracking algorithm, FCM, necessary conditions-based algorithm, NOA, and HPF, result in the same average locating cost for uniform distribution,

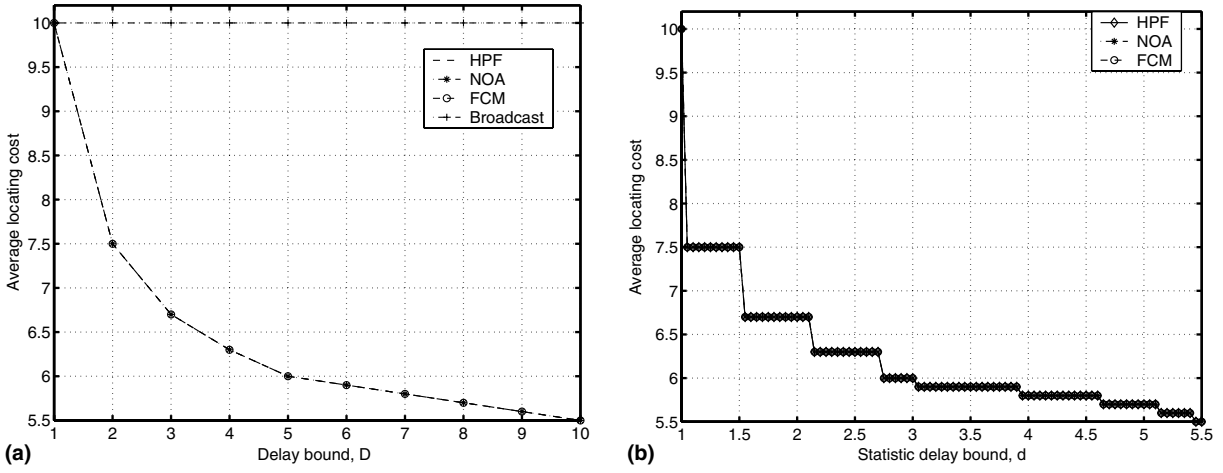

Fig. 4. Locating cost for uniform distribution: (a) deterministic delay constraints; (b) statistic delay constraints.

which fall very fast as the delay bound increases. This is because, for uniform distribution, all three algorithms yield the same partition of searching areas. In particular, when the delay bound is 5, the locating costs achieve the small asymptotic value. Therefore, there is a trade-off between the locating cost and delay constraints, which is important in handling mobile services with different requirements.

Under statistic delay constraints, average locating costs also decrease as the delays increase, but they maintain the same for some varying values of delays as shown in Fig. 4(b). In fact, this is a reasonable phenomena because paging messages are sent one by one, i.e., the paging can be finished only in integer number of polling cycles. In reality, no searching process can be terminated at halfway of a polling cycle. Each paging response must be sent within a complete polling cycle. Therefore, these numerical results provide a very good reference for determining QoS levels with reasonable delay constraints. Again, the results of three partition algorithms are overlapped for the same reason for deterministic delay bounds. The average searching delays are shown in Fig. 5. It is observed that the average delays increase as the delay bound increases. We conclude that the fast tracking algorithm matches the optimal partition algorithm based on necessary conditions, and enhanced HPF very well.

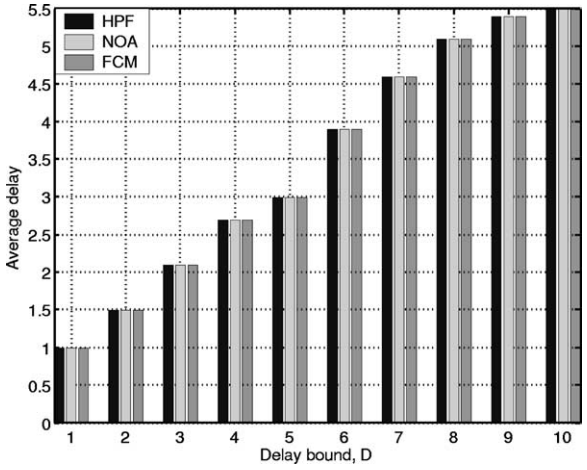

Fig. 5. Searching delay for uniform distribution.

6.2. Locating cost and delay for truncated Gaussian distribution

In this section, the average locating cost and delay versus delay bounds are investigated for truncated Gaussian distribution. When the mobile users are moving toward their destinations, they are likely to keep going along a specific path. The location probabilities of the cells covering the path will be higher than those regions that are away from the path. Thus, the truncated Gaussian distribution is more appropriate for modeling this type of movement.

The average locating cost and delay of different schemes are revealed in Figs. 6 and 7 for truncated discrete Gaussian distribution with mean zero and variance one. When an MT's location probability p_j is a truncated discrete Gaussian distribution, the average locating cost, C, decreases very quickly as

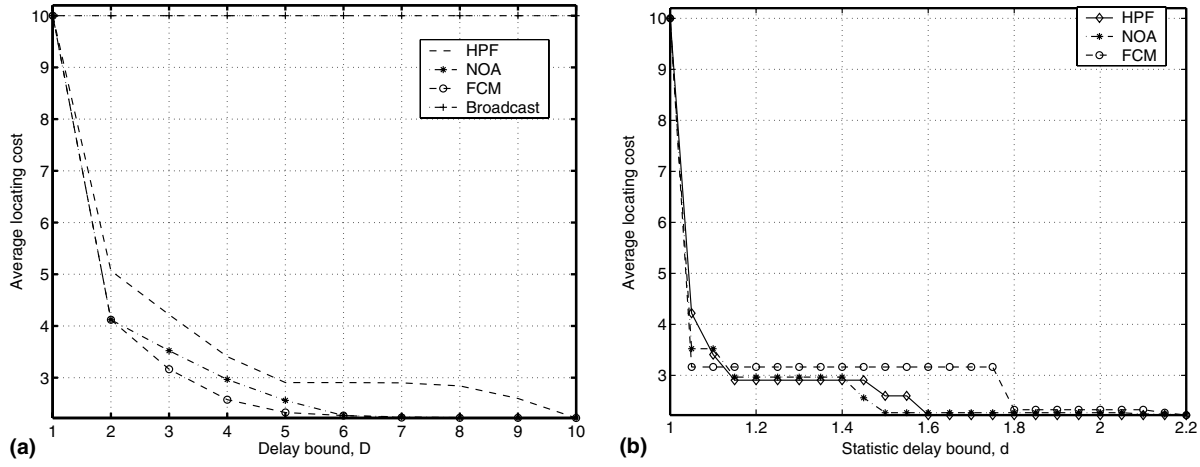

Fig. 6. Locating cost for truncated Gaussian distribution: (a) deterministic delay constraints; (b) statistic delay constraints.

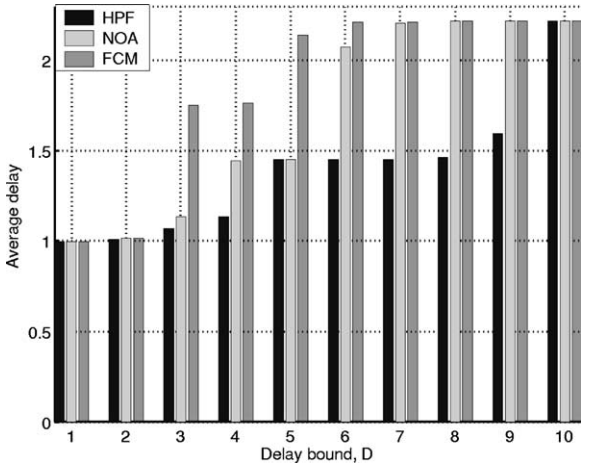

Fig. 7. Searching delay for truncated Gaussian distribution.

$\mathscr{D}$ changes from 1 to 4, then it converges to the minimum value. Note that the FCM algorithm yields the minimum cost compared to E-HPF and NOA. In particular, the improvement of locating cost of FCM over NOA is up to 20% at $\mathscr{D} = 4$ as shown in Fig. 6(a), which indicates that the necessary conditions for cost minimization is not sufficient. Under the statistic delay constraints, again, locating costs decrease as the delays constraints increase. Moreover, we find that with a small amount of increase in average delay, the locating cost can be reduced tremendously. For example, in Fig. 6(b), when the delay constraint is 1 polling cycle, the cost is 10. However, when the delay constraint is 1.2 polling cycles, the cost is around 3. Thus, the locating cost is reduced more than 3 times while the average paging delay is increased by 20%. These results provide a very good reference for systems to establish the services class with regard to the mobile users' QoS expectation.

In addition, the maximum average delay is 2.2 of truncated Gaussian distribution, which is much smaller than that for uniform distribution with 5.5. This demonstrates that the assumption of the uniform distribution will yield the most disadvantageous result although it is still being widely used. In reality, it is very unlikely to have a uniform distribution of location probabilities because of dynamic environments. Thus, we can consider the cost and delay of uniform distributions as the worst bounds, i.e., our algorithm would not affect results negatively even in the worst case. The deployment of cost-minimization location tracking algorithms are more effective for the truncated Gaussian distribution. In many mobile environments, especially in public access wireless networks, the movement patterns of mobile users would be destination-oriented. The location probabilities are thus more likely to be truncated Gaussian distribution than uniform. The average searching delays are illustrated in Fig. 7. As the delay bound increases, the average searching delay also increases. FCM algorithm generates the minimum cost among three algorithms, whereas the average delays of FCM are higher than those of the other two algorithms. In other words, the benefits of locating cost is gained at the expense of a relatively small increase in the average delay.

6.3. Locating cost and delay for irregular probability distribution

We also investigate the average locating cost and searching delay for irregular location probability distributions when the location probability distribution may not be represented by a particular function. We conduct our simulations based on three irregular distribution cases created randomly in Table 1. The *locating costs* and *average delays* of cases A, B, and C are shown in Table 2. The details of each SA, such as partitions, searching sequence, the location probability, and the number of cells under $\mathscr{D} = 3, 4, 5$, are also illustrated. The minimum locating costs for each delay bound are indicated in bold and the minimum average delays are indicated in italics in Table 2.

The locating costs decrease as the delay bound increases. For instance, the costs of case A are changed from $3.88 \rightarrow 3.52 \rightarrow 3.29$ when $\mathscr{D}$ is changed from 3 to 5. It can be seen that the FCM algorithm results in the minimum locating costs for all cases.

Table 1
The irregular location probability distributions

Cell sequence	1	2	3	4	5	6	7	8	9	10
Case A	0.36	0.31	0.05	0.05	0.045	0.045	0.04	0.04	0.03	0.03
Case B	0.28	0.26	0.08	0.08	0.05	0.05	0.05	0.05	0.05	0.05
Case C	0.16	0.16	0.16	0.16	0.06	0.06	0.06	0.06	0.06	0.06

Table 2
The comparison of locating cost and delay

Partitions (SAs)	SA(i, q_i, n_i)	$E[C(\mathscr{D})]$	$E[D(\mathscr{D})]$
Case A: $\mathscr{D} = 3$			
NOA	(1, 0.36, 1); (2, 0.36, 2); (3, 0.28, 7)	4.24	1.92
FCM	(1, 0.67, 2); (2, 0.19, 4); (3, 0.14, 4)	**3.88**	1.47
E-HPF	(1, 0.77, 4); (2, 0.13, 3); (3, 0.10, 3)	4.99	*1.33*
Case A: $\mathscr{D} = 4$			
NOA	(1, 0.36, 1); (2, 0.31, 1); (3, 0.19, 4); (4, 0.14, 4)	**3.52**	2.11
FCM	(1, 0.36, 1); (2, 0.31, 1); (3, 0.19, 4); (4, 0.14, 4)	**3.52**	2.11
E-HPF	(1, 0.72, 3); (2, 0.14, 3); (3, 0.08, 2); (4, 0.06, 2)	4.28	*1.48*
Case A: $\mathscr{D} = 5$			
NOA	(1, 0.36, 1); (2, 0.31, 1); (3, 0.10, 2); (4, 0.13, 3); (5, 0.10, 3)	**3.29**	2.30
FCM	(1, 0.36, 1); (2, 0.31, 1); (3, 0.10, 2); (4, 0.13, 3); (5, 0.10, 3)	**3.29**	2.30
E-HPF	(1, 0.67, 2); (2, 0.10, 2); (3, 0.09, 2); (4, 0.08, 2); (5, 0.06, 2)	3.52	*1.76*
Case B: $\mathscr{D} = 3$			
NOA	(1, 0.54, 2); (2, 0.26, 4); (3, 0.20, 4)	4.64	1.66
FCM	(1, 0.54, 2); (2, 0.21, 3); (3, 0.25, 5)	**4.63**	1.71
E-HPF	(1, 0.70, 4); (2, 0.15, 3); (3, 0.15, 3)	5.35	*1.45*
Case B: $\mathscr{D} = 4$			
NOA	(1, 0.54, 2); (2, 0.16, 2); (3, 0.15, 3); (4, 0.15, 3)	**4.27**	1.91
FCM	(1, 0.54, 2); (2, 0.16, 2); (3, 0.15, 3); (4, 0.15, 3)	**4.27**	1.91
E-HPF	(1, 0.62, 3); (2, 0.18, 3); (3, 0.10, 2); (4, 0.10, 2)	4.74	*1.68*
Case B: $\mathscr{D} = 5$			
NOA	(1, 0.54, 2); (2, 0.16, 2); (3, 0.10, 2); (4, 0.10, 2); (5, 0.10, 2)	4.12	*2.06*
FCM	(1, 0.28, 1); (2, 0.26, 1); (3, 0.16, 2); (4, 0.15, 3); (5, 0.15, 3)	**3.99**	2.63
E-HPF	(1, 0.54, 2); (2, 0.16, 2); (3, 0.10, 2); (4, 0.10, 2); (5, 0.10, 2)	4.12	*2.06*
Case C: $\mathscr{D} = 3$			
NOA	(1, 0.48, 3); (2, 0.22, 2); (3, 0.30, 5)	5.54	1.82
FCM	(1, 0.32, 2); (2, 0.32, 2); (3, 0.36, 6)	**5.52**	2.04
E-HPF	(1, 0.64, 4); (2, 0.18, 3); (3, 0.18, 3)	5.62	*1.54*
Case C: $\mathscr{D} = 4$			
NOA	(1, 0.32, 2); (2, 0.32, 2); (3, 0.18, 3); (4, 0.18, 3)	**4.98**	2.22
FCM	(1, 0.32, 2); (2, 0.32, 2); (3, 0.18, 3); (4, 0.18, 3)	**4.98**	2.22
E-HPF	(1, 0.48, 3); (2, 0.28, 3); (3, 0.12, 2); (4, 0.12, 2)	5.28	*1.88*
Case C: $\mathscr{D} = 5$			
NOA	(1. 0.32, 2); (2, 0.32, 2); (3, 0.12, 2); (4, 0.12, 2); (5, 0.12, 2)	**4.80**	2.40
FCM	(1. 0.32, 2); (2, 0.32, 2); (3, 0.12, 2); (4, 0.12, 2); (5, 0.12, 2)	**4.80**	2.40
E-HPF	(1. 0.32, 2); (2, 0.32, 2); (3, 0.12, 2); (4, 0.12, 2); (5, 0.12, 2)	**4.80**	2.40
Broadcast scheme	(1,1,10)	10	1

The NOA can also yield the minimum costs for some scenario, such as $\mathscr{D} = 5$ for *Case A* and $\mathscr{D} = 4$ for *Case B*. Occasionally, all three algorithms may result in the minimum cost, such as $\mathscr{D} = 5$ for *Case C*. According to the *locating costs* of *Case A*, the FCM algorithm can improve up to 22% compared to E-HPF and 8.5% compared to NOA algorithms. Therefore, the new cost-minimization algorithm is very effective in achieving the minimum locating costs regardless of location probability distributions and delay bounds.

6.4. Computation complexity

Fig. 8 shows the computation complexity of fast tracking algorithm (“FCM”), necessary conditions algorithm as “NOA”, and no probability constraints, which is referred to as “ALL” for various number of cells under different delay constraints. When the number of cells is 10, the dotted line in Fig. 8 shows the result of not using any partition constraints. The dashed line is for using necessary conditions, and the solid line is based on FCM algo-

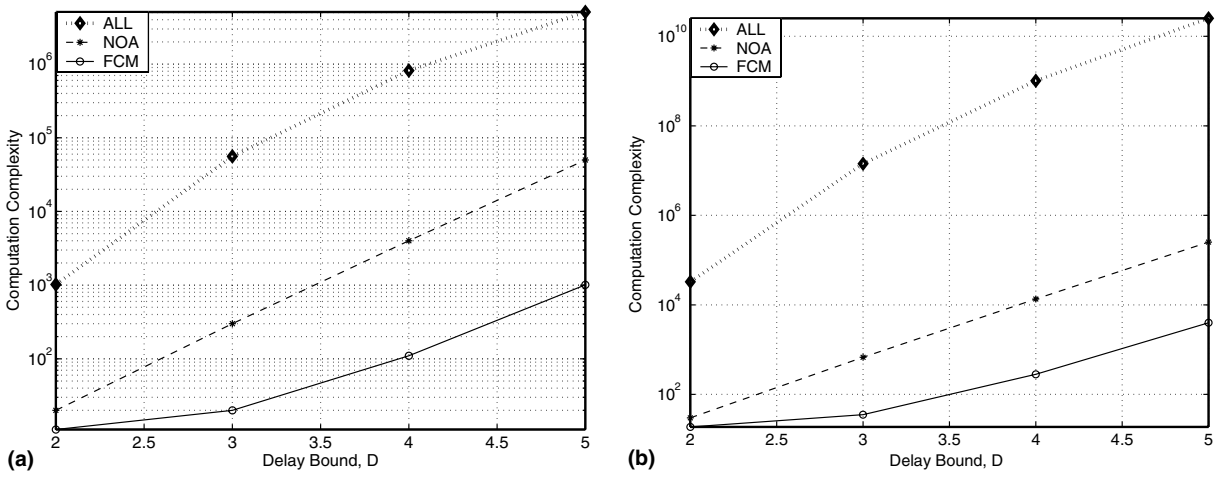

Fig. 8. Computation complexity: (a) number of cells, $N = 10$; (b) number of cells, $N = 15$.

rithms, where delay bounds are increased from 2 to 5. The figures show that the proposed fast algorithm saves up to three orders of magnitude computation than that for not using optimal partition and up to 50 times than that of using NOA. The more the number of cells, the more significant is the improvement. When the number of cells is increased to 15, the computation can further be reduced from six orders of magnitude to two orders of magnitude compared to partition without using NOA and using NOA, respectively. It indicates that the fast location tracking algorithm is very efficient in reducing the computation for cost-minimization location tracking in mobile wireless networks.

In addition to computation complexity, it is also important to consider the buffers required for location probabilities in real systems. Here we evaluate only the numerical results of computation complexity. In our previous work [3], we presented a detailed procedure to calculate location probabilities and buffer requirement in the third generation cellular wireless systems based on technical specifications. According to our previous results, the number of cells to be considered in computation is less than 10 in most of the scenarios. Therefore, our numerical results here are applicable to real systems.

7. Conclusions

We have presented a new algorithm for cost-minimization location tracking in mobile wireless networks. Based on a unimodal property of two-step locating cost as a function of the corresponding 2-partition of the location area, the new algorithm reduces the computation complexity significantly while achieving an optimal partition. In addition to the investigation of minimum cost searching problem under deterministic delay bound, we also addressed the searching problem under statistic delay constraints by using a sequential matching algorithm. Our simulation results show that the proposed algorithms are applicable to different location probability distributions. This is an important merit because uniform or Gaussian distributions of location probabilities are not present in real systems. As a matter of fact, these probabilities change over time and are irregularly distributed. By combining with location estimation algorithms, the proposed algorithms can easily be implemented in wireless systems.

Acknowledgements

The research of Wang was supported in part by NSF grant ANI-0322893. The research of Xue was supported in part by ARO grant W911NF-04-1-0385 and NSF grants CCF-0431167 and ANI-0312635.

References

[1] ETSI TS 129 010 V4.0.0 (2001–03). Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); Signalling procedures and the Mobile Application Part (MAP), March, 2001.

[2] A. Abutaleb, V.O.K. Li, Paging strategy optimization in personal communication system, ACM-Baltzer Journal of Wireless Networks (WINET) 3 (3) (1997) 195–204.

[3] I.F. Akyildiz, W. Wang, The predictive user mobility profile framework for wireless multimedia networks, IEEE/ACM Transactions on Networking 12 (6) (2004) 1021–1035.
[4] B-N. Amotz, I. Kessler, M. Sidi, Mobile users: to update or not to update? in: IEEE INFOCOM' 94, vol. 2, June, 1994, pp. 570–576.
[5] L.P. Araujo, J.R.B. de Marca, A comparative analysis of paging and location update strategies for PCS networks, in: IEEE ICC'98, June, 1998, pp. 1390–1394.
[6] A. Bhattacharya, S.K. Das, LeZi-update: an information-theoretic approach to track mobile users in PCS networks, in: ACM/IEEE MobiCom'99, August, 1999, pp. 1–12.
[7] P.P. Demestichas, V.P. Demesticha, E.C. Tzifa, M.G. Kazanitzakis, M.E. Anagnostou, Efficient location and paging area planning in future cellular systems, Wireless Personal Communications 12 (1) (2000) 83–109.
[8] Y. Fang, I. Chlamtac, Y.-B. Lin, Portable movement modeling for PCS networks, IEEE Transactions on Vehicular Technology 49 (4) (2000).
[9] J. Korhonen, Introduction to 3G Mobile Communications, Artech House, Boston, MA, 2003.
[10] T. Liu, P. Bahl, I. Chlamtac, Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks, IEEE Journal on Selected Areas in Communications (JSAC) 16 (6) (1998) 389–400.
[11] S. Mishra, O.K. Tonguz, New metric for analyzing multi-step paging schemes in mobile networks, in: IEEE VTC 2001, vol. 4, May, 2001, pp. 2590–2594.
[12] A. Misra, A. Roy, S.K. Das, An information-theoretic framework for optimal location tracking in multi-system 4G wireless networks, in: Proceedings of IEEE INFOCOM'2004, June, 2004.
[13] Z. Naor, H. Levy. Minimizing the wireless cost of tracking mobile users: an adaptive threshold scheme, in: IEEE INFOCOM'98, March, 1998, pp. 720–727.
[14] A.-C. Pang, Y.-B. Lin, H.-M. Tsai, P. Agarwal, Serving radio network controller relocation for umts all-ip networks, IEEE Journal on Selected Areas in Communications 22 (4) (2004) 617–629.
[15] C. Rose, State-based paging/registration: a Greedy technique, IEEE Transactions on Vehicular Technology 48 (1) (1999) 166–173.
[16] C. Rose, R. Yates, Minimizing the average cost of paging under delay constraints, ACM-Baltzer Journal of Wireless Networks (WINET) 1 (1995) 211–219.
[17] T. Tung, A. Jamalipour, A Kalman-filter based paging strategy for cellular networks, in: Proceedings of the 36th Annual Hawaii Internal Conference on System Science 2003, January, 2003, pp. 308–317.
[18] G. Wan, E. Lin, Cost reduction in location management using semi-realtime movement information, ACM-Baltzer Journal of Wireless Networks (WINET) 5 (4) (1999) 245–256.
[19] W. Wang, I.F. Akyildiz, G. Stüber, B.-Y. Chung, Effective paging schemes with delay bounds as QoS constraints in wireless systems, ACM-Kluwer Wireless Networks (WINET) 7 (10) (2001) 455–466.
[20] W. Wang, I.F. Akyildiz, G.L. Stüber, An optimal paging scheme of minimizing signaling costs under delay bounds, IEEE Communications Letters 5 (2) (2001) 42–45.
[21] Wolframe Research: MathWorld, Stirling Number of the Second Kind, February, 2004.
[22] T.Y.C. Woo, T.F.L. Porta, J. Golestani, N. Agarwal, Update and search algorithms for wireless two-way messaging: design and performance, in: IEEE INFOCOM'98, March, 1998, pp. 737–747.
[23] Y. Xiao, Y. Pan, J. Li, Design and analysis of location management for 3G cellular networks, IEEE Transactions on Parallel and Distributed System 15 (5) (2004) 339–349.
[24] T. Zhang, S-W. Li, Y. Ohba, N. Hakajima, A flexible and scalable IP paging protocol, in: IEEE GLOBECOM'2002, vol. 1, November, 2002, pp. 630–635.

Wenye Wang (M'98/ACM'99) received the B.S. and M.S. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 1986 and 1991, respectively. She also received the M.S.E.E. and Ph.D. degree from Georgia Institute of Technology, Atlanta, Georgia in 1999 and 2002, respectively. She is now an Assistant Professor with the Department of Electrical and Computer Engineering, North Carolina State University. Her research interests are in mobile and secure computing, quality-of-service (QoS) sensitive networking protocols in single- and multi-hop networks. She has served on program committees for IEEE INFOCOM, ICC, ICCCN in 2004. She has been a member of the Association for Computing Machinery since 2002.

Guoliang Xue is a Professor in the Department of Computer Science and Engineering at Arizona State University. He received the Ph.D. degree in Computer Science from the University of Minnesota in 1991 and has held previous positions at the Army High Performance Computing Research Center and the University of Vermont. His research interests include efficient algorithms for optimization problems in networking, with applications to fault tolerance, robustness, and privacy issues in networks ranging from WDM optical networks to wireless ad hoc and sensor networks. He has published over 100 papers in this area. His research has been continuously supported by federal agencies including NSF, ARO and DOE. He is the recipient of an NSF Research Initiation Award in 1994, and an NSF-ITR Award in 2003. He is an Associate Editor of the IEEE Transactions on Circuits and Systems-I, an Associate Editor of the Computer Networks Journal, and an Associate Editor of the IEEE Network Magazine. He has served on the executive/program committees of many IEEE conferences, including Infocom, Secon, Icc, Globecom and QShine. He is a TPC co-chair of IEEE Globecom'2006 Symposium on Wireless Ad Hoc and Sensor Networks.