Tree-Based Object Tracking Without Mobility Statistics in Wireless Sensor Networks

Li-Hsing Yen
Dept. Computer Science & Information Engineering
National University of Kaohsiung
Kaohsiung, Taiwan 811, R.O.C.
lhyen@nuk.edu.tw

Bang Ye Wu
Dept. Computer Science & Information Engineering
National Chung Cheng University
Chiayi County, Taiwan 621, R.O.C.

Chia-Cheng Yang
Dept. Computer Science & Information Engineering
Chung Hua University
Hsinchu, Taiwan 300, R.O.C.

Abstract

Object tracking in wireless sensor networks is to track mobile objects by scattered sensors. These sensors are typically organized into a tree to deliver report messages upon detecting object's move. Existing tree construction algorithms all require a mobility profile that characterizes the movement statistics of the target object. Mobility profiles are generally obtained based on historical running traces. The contribution of this work is twofold. We first show that the problem of finding an optimal message report tree that requires the least amount of report messages is NP-hard. We then propose analytic estimates of mobility profiles based on Markov-chain model. This profiling replaces an otherwise experimental process that generates and analyzes running traces. Simulation results show that the analytic profiling works well and can replace costly statistical profiling without noticeable performance degradation.

Keywords: Network topology, Sensor networks, Algorithm/protocol design and analysis, Trees, Stochastic processes.

1 Introduction

Rapid progress in wireless communications and micro-sensing MEMS technology have enabled the deployment of wireless sensor networks. A wireless sensor network (WSN) consists of a large number of sensor nodes deployed in a region of interest. Each sensor node is capable of collecting, storing, and processing environmental information, and communicating with other sensors via a radio transceiver.

Object tracking is an application of WSNs where the presence of particular mobile objects (animals, vehicles, etc.) can be detected by nearby sensors. End users can track the target

object's location through a particular node called *sink*. The tracking task can be conducted in various ways. In some model, the target can be detected by more than one sensor simultaneously, triggering multiple detection reports from disparate sources. A challenge is to coordinate these sensors to make the tracking process more accurate, dependable, or energy efficient. A feasible approach to energy efficiency is to predict and shut down sensors that are redundant in the tracking task [16]. In [17], sensors around a moving target collectively form a tree structure to facilitate data aggregations, which significantly reduces network traffic and energy consumption.

In this paper, we assume that only one node is in charge of the location detection task at any time. We in fact follow the model proposed by Lin *et al.* [11], a variation of STUN architecture [8]. Each sensor has its own duty area. These duty areas are non-overlapping and collectively cover the whole target region. All nodes are organized into a tree structure rooted at the sink. Every sensor maintains a *detection list*. The detection list for a sensor with b children is a $(b+1)$-tuple of object set. The first element in the tuple records the set of objects that are under the sensor's observation, while other elements record those observed by its descendants, one for each branch. All detection lists together pave a *query path* for each object, which allows end users to retrieve through the sink the current location of a target object. The location retrieval is realized by propagating a Query message along the query path to the sensor that currently observes the target object. If the current observer is a leaf node, its parent is able to respond the query as the observer is the only node in the branch it resides.

When a sensor detects a target object entering into its duty area, it sends an Enter message toward the sink (along the tree) to create or update the associated query path. When a target is observed out of a sensor's observation, a Leave message is sent by the sensor and destined for the sink node to remove the obsolete query path. Both Enter and Leave messages are report messages. When a target crosses the border of two neighboring sensor's duty areas, a pair of report messages (one Leave and the other Enter) will be triggered. We assume that both the old and new sensors in charge of the target know which border the target is crossing, so triggered report messages need only be delivered hop-by-hop to the nearest common ancestor of their sources to renew the query path (Fig. 1). Consequently, once an initial query path is established, subsequent report messages do not always need to reach the sink.

Such a tree-based object tracking approach incurs two types of message costs: *update cost* (the number of report messages transmitted) and *query cost* (the number of query messages transmitted). Both are governed by the shape of the underlying tree structure. In this work, we are given the freedom to derive a tree from the initial network topology. To construct a tree that minimizes update cost, a kind of mobility profile that describes object-detecting rate at each sensor [8] or border-crossing rates between each pair of neighboring sensors [11] is required. To improve overall (update plus query) message cost, however, we also need the knowledge of query rate, the rate at which each sensor is queried by the sink. We are therefore required to answer the following two questions:

- How to obtain these profile parameters (object-detecting rate, border-crossing rate, query rate, etc.)?
- How to construct an optimal tree based on these parameters?

Profile parameters can be acquired by statistically analyzing real traces generated by existing object tracking systems. However, historical data for an unexplored region may not be available. Even the data are available, they may not be representative of target movements or query patterns in the future (history does not necessarily repeat itself.) Alternatively, the profile can be learned from traces generated by simulating an object moving around the deployment field.

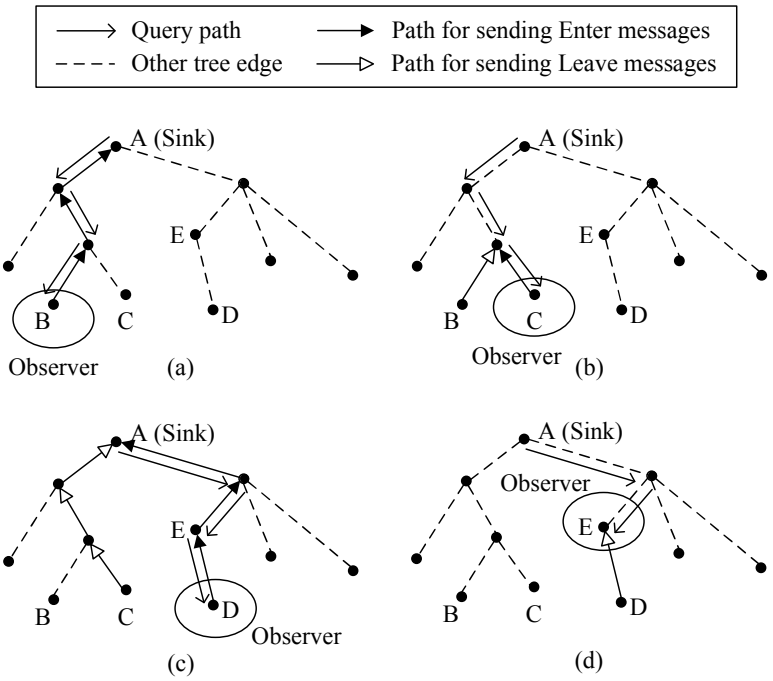

Figure 1: Transmissions of report messages and resulted query paths when (a) the target is under B's observation initially and in turn moves to (b) C's (c) D's, and finally (d) E's duty areas.

In this way, the resulting profile may heavily depend on the mobility model used to drive the object and the query-generating model, making the profile model-dependent. Another issue of the simulation-based profiling is to determine a sufficient length of simulation time to yield an archetypal statistic.

As to the second question, constructing an optimal tree that minimizes both update and query costs has been identified NP-complete [12]. However, most existing solutions consider a variant version of the tree construction problem: constructing a tree that minimizes *only* update costs (while disregarding query costs). We refer to this problem as the *Optimal Message Report Tree* (OMRT) problem in this paper. The computational complexity of the OMRT problem was previously unknown. DAB [8] and DAT [11] are two tree-construction heuristics for the OMRT problem. QCR [11] is a heuristic that attempts to restructure an existing tree to further reduce query cost for better overall performance.

The contribution of this work is twofold. First, we have developed a mathematical model that derives mobility profiles based on topological information without traces from real history or simulations. This profiling is valuable especially when the mobility pattern of target objects is unknown. We have conducted simulations, where two commonly-adopted mobility models, Random Waypoint [6, 2] and Gauss-Markov [10], were used to pilot a target object. The results indicate that the proposed analytical profiling can replace statistical counterparts without noticeable performance degradation.

Another contribution of this paper is on the analysis of tree constructions. We have proven that the OMRT problem, though less restrictive than the problem considered in [12], is still NP-hard. This result holds even if the problem is further simplified such that all border-crossing rates are identical. We also have shown the limitation of *deviation-free* heuristic used by DAT and investigated the potential of using the classical maximum spanning tree (MST) algorithm

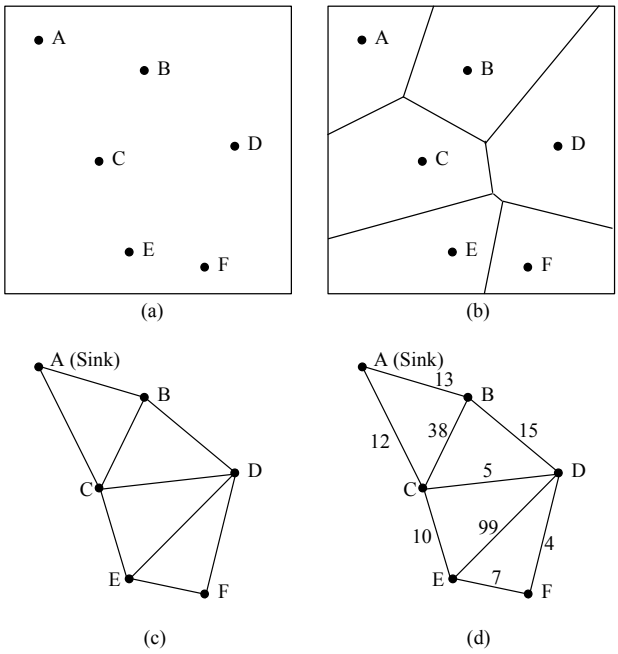

Figure 2: (a) A sensor deployment; (b) the corresponding Voronoi diagram; (c) the corresponding network profile; and (d) a sample mobility profile

as an alternative. Extended simulations indicates that with the aid of the analytic profiling, MST performs better than DAT in the amount of report messages used.

The remainder of this paper is organized as follows. The next section analyzes the computational complexity of the tree construction problem and discusses some heuristics. Sec. 3 details the analytic profiling. Experimental results are presented in Sec. 4. The last section concludes this paper.

2 Preliminaries and Problem Analysis

We assume a WSN consisting of N sensors placed in a closed region. The location of every sensor need not be engineered or predetermined but should be known after the deployment. This requirement can be met by equipping each sensor with special hardware such as a GPS (Global Positioning System) device or by other techniques [3, 13]. A sensor can detect and report an object that is within some range from it. To avoid redundant reports, we assume that only the sensor that is closest to the target object is in charge of the detection. We also assume that the whole deployment region is fully covered, so the duty area of every sensor can be captured by a Voronoi diagram on the deployment region (Fig. 2(b)). A Voronoi diagram can be transformed into a Delaunay graph $G(V, E)$, where V is the set of all sensors and edge $e(i, j) \in E$ for all $i, j \in V$ if i and j share a common border in the Voronoi diagram. The Delaunay graph serves as a *network profile* (Fig. 2(c)), which uniquely defines the adjacency relation between sensors without specifying object's movement characteristics.

Given a network profile $G = (V, E)$, $M = (V, E, w)$ is a *mobility profile* if w is a function defined on E such that $w(u, v)$ gives the handoff (border-crossing) rate between sensors u and v for all edge $e(u, v) \in E$. A mobility profile can be illustrated as a weighted graph with weights represent border-crossing rates. Fig. 2(d) shows an example.

2.1 Message Report Tree

Definition 1 *Given a mobility profile $M = (V, E, w)$, T is a message report tree (MRT) derived from M if T is a spanning tree derived from $G = (V, E)$.*

Given an MRT T, let $d_T(i, j)$ denote the *distance* between i and j on T, i.e., the edge count of the unique path in T that connects nodes i and j. Recall that when an object crosses the border of two sensors, say u and v, a pair of report messages (one Enter and the other Leave) has to be sent hop-by-hop to the nearest common ancestor of u and v in T. Such a crossing therefore causes $d_T(u, v)$ message transmissions. Consequently, the update cost between u and v can be computed as $w(u, v) \times d_T(u, v)$. For an MRT T derived from a mobility profile $M = (V, E, w)$, the update cost of T is given by

$$u(T) = \sum_{(u,v)\in E} \left(w(u, v) \times d_T(u, v)\right). \tag{1}$$

For better overall performance, query cost must also be considered. An end user may issue query messages from the sink with a regular or an irregular time interval, asking for the target object's current location. These messages are also delivered along the message report tree. Suppose that the target is currently under sensor i's surveillance. The query message must be delivered hop-by-hop to i (if i is a non-leaf node) or to i's parent (otherwise).

To estimate entire query cost, we need the information of query rates. Let $\langle q_1, q_2, \ldots, q_N\rangle$ be a *query profile*, where q_i is the rate at which sensor i is queried by the sink. Let $L(T)$ be the set of leaf nodes in a tree T. For an MRT T rooted at S, the query cost is

$$q(T) = \sum_{u\in L(T)} q_u \times d_T(S, p_T(u)) + \sum_{u\notin L(T)} q_u \times d_T(S, u),$$

where $p_T(u)$ is the parent of node u in T.

To derive an MRT T from a given network profile that minimizes overall cost $(u(T) + q(T))$ requires both mobility and query profiles. This problem has been proven NP-complete [12]. In this work, we consider a less-restrictive problem that minimizes only update costs. We shall prove that even this problem is NP-hard.

Definition 2 *Optimal Message Report Tree (OMRT) problem is to find an MRT from a given mobility profile that minimizes the update cost.*

In the following, we recast the OMRT problem into a decision problem, and show that the decision problem cannot be solved in polynomial time.

Definition 3 *Given a mobility profile and a cost limit c, the Bounded Cost Message Report Tree (BCMRT) problem is to ask whether we can find an MRT from the mobility profile with update cost not exceeding c.*

Theorem 1 *The BCMRT problem is NP-hard.*

Proof: See Appendix. □

The OMRT problem in fact reduces to the *optimum communication spanning tree* (OCST) problem [5]. For a given undirected graph $G = (V, E, l)$ with non-negative edge length function

l and a matrix λ that gives the communication requirement between each node pair in G, the OCST problem is to derive a spanning tree T from G that minimizes

$$\sum_{u,v \in V} \left(\lambda(u,v) \times \hat{d}_T(u,v) \right),$$

where $\hat{d}_T(u,v)$ is the sum of all edge lengths on the path from u to v in T. The OMRT problem is a special case of the OCST problem with $l(e) = 1$ for all $e \in E$ and

$$\lambda(u,v) = \begin{cases} w(u,v) & \text{if } (u,v) \in E \\ 0 & \text{otherwise.} \end{cases}$$

The OCST problem has been proven NP-complete [7]. Although approximations and evolutionary algorithms have been proposed toward the OCST problem [14, 9], these methods are not considered practical in WSNs due to numerous nodes involved in the computation.

2.2 Tree Construction Heuristics

DAT [11] takes a heuristic that adds only deviation-free edges into the tree. An edge $e(u,v) \in E$ is deviation-free if it is the first edge of some shortest path from either u or v to the sink. Consequently, the resultant *DAT tree* possesses the property that the path from every node to the sink along the tree must be one of those in the underlying network profile that have the minimum edge count. As an illustration, Fig. 3(a) is a DAT tree derived from the mobility profile shown in Fig. 2(d). Observe that for any node i, the path from i to the sink in the tree has the minimum edge count among all possible.

We observed that the deviation-free property may not be a plus as a whole. Consider the tree shown in Fig. 3(b), which is derived from the same mobility profile but without the deviation-free property. The formation of edge (C, B) helps in reducing the update cost between B and C: the report messages issued whenever an object crosses the border between B and C are now delivered in only one hop (from C to B) instead of two (one from C to A and the other from B to A). Evidently, edge (C, B) breaks the deviation-free property because the hop count from C to the sink (which is two) is not the minimum among all in the underlying network profile (which is one). Here not having the deviation-free property has the negative effect of adding one hop to report messages sending from C to A. Nevertheless, the amount of cost reduction between C and B compensates the increase between C and A. In fact, the cost of the entire tree in Fig. 3(a) is 564, while that of the tree in Fig. 3(b) is only 244.

DAT implicitly minimizes the distance between each node and the sink S. It follows that DAT minimizes *average node level* (ANL). For an MRT T=(V, E) with sink node S, we have

$$\text{ANL}(T) = \frac{\sum_{u \in V} d_T(u,S)}{|V|}.$$

What we have learned from the above example is that to minimize $u(T)$, it is crucial to lower in-between distances for node pairs that have high border-crossing rates. However, two neighboring sensors with high border-crossing rates may not share a common edge in a DAT tree. We therefore argue that *average reporting length* (ARL) may be more important in designing heuristics for the OMRT problem. The ARL for a given MRT T=(V, E) is defined as

$$\text{ARL}(T) = \frac{\sum_{(u,v) \in E} d_T(u,v)}{|E|}.$$

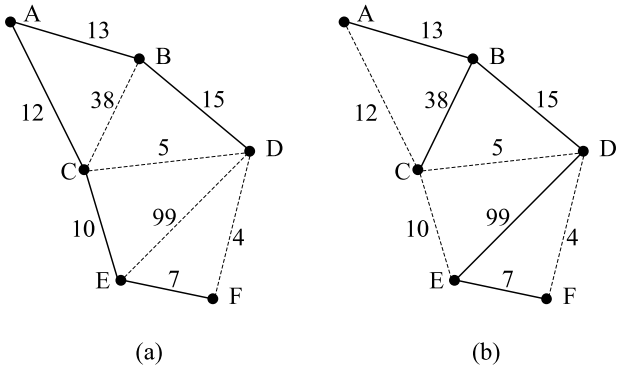

Figure 3: (a) The tree derived by DAT from Fig. 2(d); (b) another tree derived from the same mobility profile. Solid lines are tree edges while dashed lines are not.

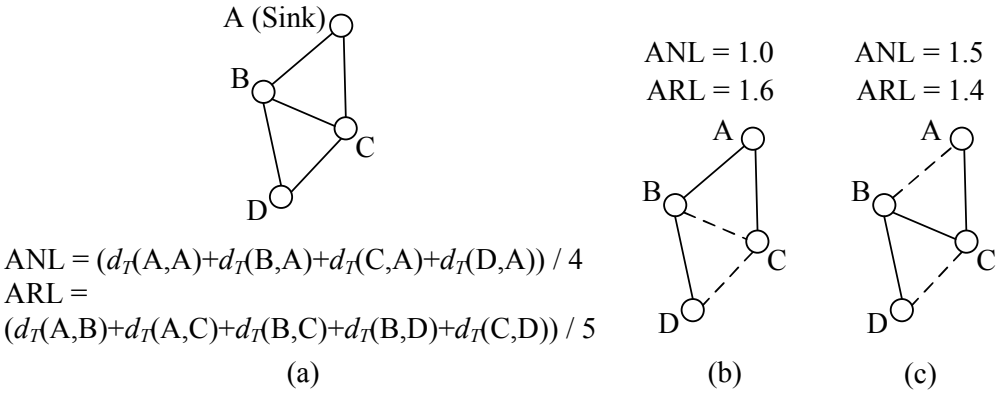

Figure 4: (a) A sample network profile; (b) A deviation-free tree T_1 derived from (a); (c) Another tree T_2 derived from (a).

Consider the case when all edges of a mobility profile have a uniform weight W. The update cost of any derived tree T then becomes

$$u(T) = W \times \sum_{(u,v)\in E} d_T(u,v). \tag{2}$$

A tree that minimizes ARL therefore also minimizes (2) and is optimal. In contrast, a tree that minimizes ANL does not necessarily minimize (2) at the same time. As an example, consider two trees shown in Fig. 4, where one tree T_1 possesses the deviation-free property and the other T_2 does not. Compared with the counterpart, T_1 has a lower ANL (1.0 versus 1.5) yet a higher ARL (1.6 versus 1.4). If each edge in Fig. 4(a) has equal weight, T_1 has a higher update cost than T_2.

Finding a spanning tree that minimizes ARL is an instance of the OMRT problem with pure unit-weight edges. In the proof of Theorem 1 (Appendix), L_1 and L_2 are polynomial in p and k. We can replace each such edge by a path of L_1, and L_2 respectively, edges of unit weight. If any edge in the paths is not in the solution, at least one pair of nodes will be separated by a path of at least L_1 hops. Therefore, we can show that the OMRT with unit requirements is still NP-hard.

Since node pairs with high border-crossing rates are more "expensive" than those with low rates, we may design a greedy approach by examining node pairs in a non-increasing order of their

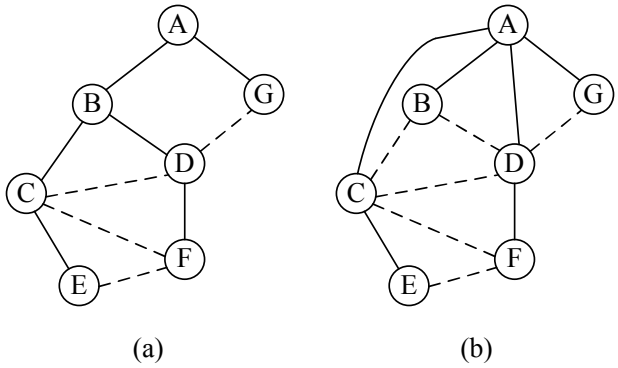

Figure 5: (a) An MRT; (b) The MRT after rewiring all of B's children to A. Solid lines are tree edges while dashed lines are edges in network profile

border-crossing rates and minimizing the distance of the node pair (u, v) under consideration by adding $e(u, v)$ into T (providing that the introduction of $e(u, v)$ does not create a cycle in T). This heuristic turns out to be the classical Maximum Spanning Tree (MST) algorithm, which derives a spanning tree from a given graph that maximizes the total weight. MST does not guarantee an optimal solution to our problem, as distances of node pairs are strongly correlated: shortening the distance of one important pair often lengthens others. However, it is simple yet effective. As a remark, the tree shown in Fig. 3(b) can be derived by MST.

Although MST seems appealing in these synthesized examples, its performance in general depends on the distribution of weights, which in turn depends on mobility patterns of target objects. We shall explore this issue by simulations in Sec. 4.

2.3 Query Cost Reduction

QCR [11] is a heuristic that attempts to restructure an existing tree to further reduce query cost for better overall performance. QCR takes two tree-adjustment techniques. The first considers turning an intermediate node into a leaf by rewiring all its children to its parent. The second technique attempts pulling leaf nodes up one level. The adjustment is performed logically; report and query messages are still propagated along the original MRT. The resulting logical tree serves only for two purposes:

- Determining the data sink for each sensor pair, which is the nearest common ancestor of the pair in the logical tree. This is where report messages of the pair are delivered.
- Determining the set of leaf nodes, for which query messages are delivered to their respective parents.

As an example, consider the original MRT shown in Fig. 5(a). If we rewire each of B's children to B's parent, A, the resultant MRT will be that shown in Fig. 5(b). Now report messages associated with neighbor pair (C, D) have to be delivered to A, the new nearest common ancestor of C and D. Since messages are still propagated along the original MRT, this adjustment causes an increase of two hops in the reporting length associated with C and D. Nevertheless, it also decreases the query cost of B as B becomes a leaf node in the logical tree.

In general, the query cost of the resultant tree T' after tree adjustments becomes

$$q(T') = \sum_{u \in L(T')} q_u \times d_T(S, p_{T'}(u)) + \sum_{u \notin L(T')} q_u \times d_T(S, u)$$

while the update cost changes to

$$u(T') = \sum_{(u,v)\in E} \left(w(u,v) \times \left(d_T(u, \rho_{T'}(u,v)) + d_T(v, \rho_{T'}(u,v))\right)\right),$$

where $\rho_{T'}(u,v)$ denotes the nearest common ancestor of nodes u and v in tree T'. It follows from the operation of QCR that $L(T) \subseteq L(T')$ and $d_T(S, p_{T'}(u)) \leq d_T(S, p_T(u))$ for all node u. Therefore, we have $q(T') \leq q(T)$. It also follows from the operation of QCR that

$$d_T(u, \rho_{T'}(u,v)) + d_T(v, \rho_{T'}(u,v)) > d_T(u, \rho_T(u,v)) + d_T(v, \rho_T(u,v)) = d_T(u,v).$$

So $u(T') > u(T)$. Consequently, adjusting a tree reduces query cost but increases update cost. Determining the overall (update plus query) cost change after a specific adjustment requires both the mobility profile and the query profile. QCR executes an adjustment only if the result of the cost computation reveals overall cost reduction. In Sec. 4, we shall investigate the quality of the proposed profiling and the performance comparison between DAT and MST in terms of query and overall costs with the help of QCR.

3 Analytic Mobility Profiling

The basic idea behind our analytic profiling is that object movements can be modeled as a stochastic process $X(t)$. The state space of $X(t)$ corresponds to the set of all sensors in the WSN. When an object is observed by sensor i at time t, we say that the object is in state i at time t, denoted by $X(t) = i$. Since we are primarily concerned with the sequence of state transitions, not the sojourn time spent in each state, we assume that time values are all discrete and increased by one only when state changes. In this way, we model object movements as a discrete-time stochastic process. We call $X(t)$ an *object movement process* (OMP). The objective is to estimate the probabilities of state transitions, from which expected border-crossing rates can be derived.

For any discrete time $n \geq 0$, let $p_i^n = \Pr[X(n) = i]$ be the probability that the process is in state i, $1 \leq i \leq N$, at time n. Define $\mathbf{p}(n) = \langle p_1^n, p_2^n, \cdots, p_N^n \rangle$ to be the n-th state vector. The first step to our goal is to evaluate $\mathbf{p}(0)$. As we have no prior knowledge of object's appearance, let us assume that the probability of the target object being in state i initially is proportional to the size of sensor i's duty area. That is,

$$p_i^0 = \frac{A_i}{\sum_{j=1}^{N} A_j},$$

where A_i is the size of sensor i's duty area.

The next step is to obtain state transition probabilities for $X(t)$. Let $\mathbf{M}(m,n)$ be the transition probability matrix for times m and n, where $m < n$. Specifically,

$$\mathbf{M}(m,n) = \begin{pmatrix} p_{1,1}^{m,n} & p_{1,2}^{m,n} & \cdots & p_{1,N}^{m,n} \\ p_{2,1}^{m,n} & p_{2,2}^{m,n} & \cdots & p_{2,N}^{m,n} \\ \vdots & \vdots & \cdots & \vdots \\ p_{N,1}^{m,n} & p_{N,2}^{m,n} & \cdots & p_{N,N}^{m,n} \end{pmatrix},$$

where

$$p_{i,j}^{m,n} = \Pr[X(n) = j | X(m) = i].$$

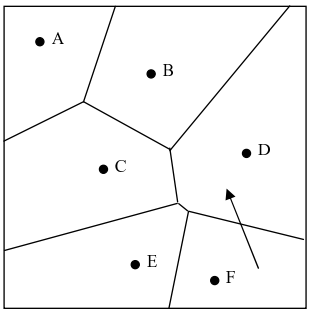

Figure 6: An object has moved from F's duty area to D's. The object is more likely to enter B's duty area than to either C's, E's, or F's.

By definition, we have $\mathbf{p}(n) = \mathbf{p}(m) \times \mathbf{M}(m, n)$.

If $X(t)$ possesses the Markov property, i.e., the conditional probability that $X(n) = j$ given $X(m) = i$, $X(m+1) = i_{m+1}$, ..., $X(n-1) = i_{n-1}$ depends only on the most recent observation $X(n-1) = i_{n-1}$, not on other history, then the state transition probabilities can be easily decomposed by applying the Chapman-Kolmogorov equation

$$p_{i,j}^{m,n} = \sum_{k=1}^{N} \left(p_{i,k}^{m,r} \times p_{k,j}^{r,n} \right)$$

for any time $m < r < n$. Unfortunately, the target's movements may not follow a mobility model that holds the Markov property. Some synthesized mobility model like Random Waypoint [6, 2] tends to maintain an object's current moving direction for an extended period of time. Consequently, if an object changes states from i to j at time t, the object is more likely to enter the duty area pointed to by the straight line connecting i to j than to any other at time $t+1$. Fig. 6 shows an example.

Including historical information in the stochastic model would make it extremely complicated and scenario-dependent. To make the analytical results meaningful and sufficiently general, we model the object movement process as a Markov chain (which possesses the Markov property). Consequently,

$$\mathbf{p}(n) = \mathbf{p}(0) \times [\mathbf{M}(0,1) \times \mathbf{M}(1,2) \times \cdots \times \mathbf{M}(n-1,n)]. \tag{3}$$

We assume that sensor nodes never move after deployment and operate long enough to accomplish their tracking mission. This assumption makes the chain a stationary process, which implies $\mathbf{M}(0,1) = \mathbf{M}(1,2) = \cdots = \mathbf{M}(n-1,n)$. Eq. (3) therefore becomes

$$\mathbf{p}(n) = \mathbf{p}(0) \times \mathbf{M}^n, \tag{4}$$

where $\mathbf{M} = \mathbf{M}(0,1)$ is a one-step transition probability matrix. Specifically,

$$\mathbf{M} = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,N} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,N} \\ \vdots & \vdots & \cdots & \vdots \\ p_{N,1} & p_{N,2} & \cdots & p_{N,N} \end{pmatrix},$$

where $p_{i,j} = \Pr[X(t+1) = j | X(t) = i]$ for any $t \geq 0$. The values of $p_{i,j}$'s can be computed as follows. Any sensor shares common borders with its neighbors, one for each. Suppose sensor

i has k neighbors with common border lengths $l_1, l_2, \ldots, l_k$, respectively. The probability that an object moves to the duty area of the j-th neighbor ($1 \leq j \leq k$), given the fact that it is current under sensor i's surveillance, is $l_j/(l_1 + l_2 + \cdots + l_k)$. In the way, the one-step transition probability matrix can be computed.

We are now required to prove that $\mathbf{M}^n$ converges as $n \to \infty$. That property enables us to obtain $X(t)$'s steady-state probability distribution.

Theorem 2 *If an OMP $X(t)$ is modeled as a Markov chain, then $\pi_j = \lim_{n\to\infty} \Pr[X(n) = j]$ exists for all state j. Let $\mathbf{M}$ be the one-step transition probability matrix for $X(t)$, we have*

$$\lim_{n\to\infty} \mathbf{M}^n = \begin{pmatrix} \pi \\ \pi \\ \vdots \\ \pi \end{pmatrix},$$

where $\pi = \langle \pi_1, \pi_2, \ldots, \pi_N \rangle$.

Proof: This theorem can be proven by showing particular properties of $X(t)$. First, $X(t)$ is *irreducible*, which asserts that an object starting at any state has a non-zero probability to eventually visit any other state. This property holds since for an arbitrary sensor deployment, the corresponding Delaunay graph is connected and our model precludes any obstacle that prevents objects from crossing a border from either direction. Second, since $X(t)$ is irreducible and has a finite state space, $X(t)$ is *recurrent*, meaning that the number of times any state is entered is infinite if $n \to \infty$. In fact, because $X(t)$ is not an infinite Markov chain, $X(t)$ is *positive recurrent*, i.e., the expected return time (the time interval between any two consecutive visits to the same state) is finite. Finally, $X(t)$ is *aperiodic*, which asserts that the return time is not fixed for any state. This can be seen by the unfeasibility of any state transition loop in which objects can be trapped infinitely. The theorem therefore follows from these properties [15]. □

It also follows [15] that $\lim_{n\to\infty} \mathbf{p}(n) = \mathbf{p}(0) \times \lim_{n\to\infty} \mathbf{M}^n = \pi$. Therefore, with the contents of $\mathbf{M}$ and π, the weight associated with each edge (i, j) in the mobility profile can be calculated as

$$w(i, j) = (\pi_i \times p_{i,j} + \pi_j \times p_{j,i}). \tag{5}$$

Note that our estimate requires only topology information, *i.e.*, the location of each sensor. The computation task, however, involves computing a Voronoi diagram, the area and border lengths of each region in the Voronoi diagram, and the limit value of the one-step transition probability matrix.

Although the above analysis assumes only one object, the derived result applies to cases with multiple objects as well provided that we assume no particular mobility patterns on these objects. The point is that under the adopted object-tracking model, each object is independently tracked. Therefore, the mobility of c objects can be modeled by c independent OMPs, where the aggregated weight associated with each edge (i, j) in the mobility profile essentially becomes $c \cdot w(i, j)$. Tree construction algorithms are not affected by such aggregations, as only relative quantities of weights among edges matter.

4 Simulation Results

4.1 Experiment Setup

We conducted simulations to investigate the performance of the analytic profiling. We assume a $100 \times 100\,\mathrm{m}^2$ deployment field with the number of deployed sensors varied from 100 to 1000. Two mobility models were used to drive object's movements, Random Waypoint and Gauss-Markov. In Random Waypoint model (RWM), an object is randomly placed initially. It then randomly selects a destination location to move with a randomly determined speed. The object waits a random paused time when it arrives at the destination and then moves to another location following the same random distributions. In Gauss-Markov model (GMM), a randomly placed object determines its moving speed and direction randomly. After a fixed period of time (5 sec. in our setting), the object alters the speed and direction according to Gaussian distributions with the current values of speed and direction used as the respective means. To preclude extreme values, any randomly generated value outside the range $[\mu - 2\sigma, \mu + 2\sigma]$ was discarded and regenerated, where μ and σ are mean and standard deviation of the distribution, respectively. In our setting, the standard deviations of speed and direction are 2.5 m/sec. and 20 degrees, respectively. We further limit the speed value to the range between 1 and 50 m/sec.

RWM and GMM were considered because these two mobility models do not prefer any particular location. Some other mobility models, such as the City Mobility Model [8] and the Manhattan Grid mobility model [1] do exhibit certain locality. We excluded these mobility models because we do not assume locality on object movements.

For a given sensor population, 100 sensor deployments were randomly generated. A Voronoi diagram was generated for each sensor deployment, from which three types of mobility profiles were created. The first one, MP_Markov, was obtained by applying the proposed Markov modeling. To produce statistical mobility profiles MP_RWM and MP_GMM, we conducted a pre-run by running an object around the deployment field according to RWM and GMM, respectively. Each type of mobility profile was used by both DAT and MST to construct trees, resulting in six possible combinations of resultant MRTs. Fig. 7 illustrates how these MRTs were constructed from a single deployment setting.

To ensure that statistical profiles are statistically significant, we investigated experimentally the relationship between the number of moves and the ratio of zero-weight edges (borders that have never been passed by the object). We used the latter to gauge the degree of unexplored region within a simulation run. The results (Fig. 8) indicate that no more than 0.8% edges were zero-weighted when over $100,000$ moves were taken in a run. We therefore let the object take 150,000 moves in each run.

4.2 Node Level and Reporting Length

We first measured average node level and reporting length for all the six MRTs. Fig. 9 shows the results. As expected, all trees formed by DAT have an identical average node level. The three MST trees all have higher average node levels than the family of DAT trees. We observed the same order in average reporting lengths for trees using statistical profilings (Fig. 9(b)). DAT(Markov) and MST(Markov), however, both have shorter average reporting lengths. The results reveal that the proposed Markov-based profiling leads to shorter reporting lengths, regardless of which tree construction method was used.

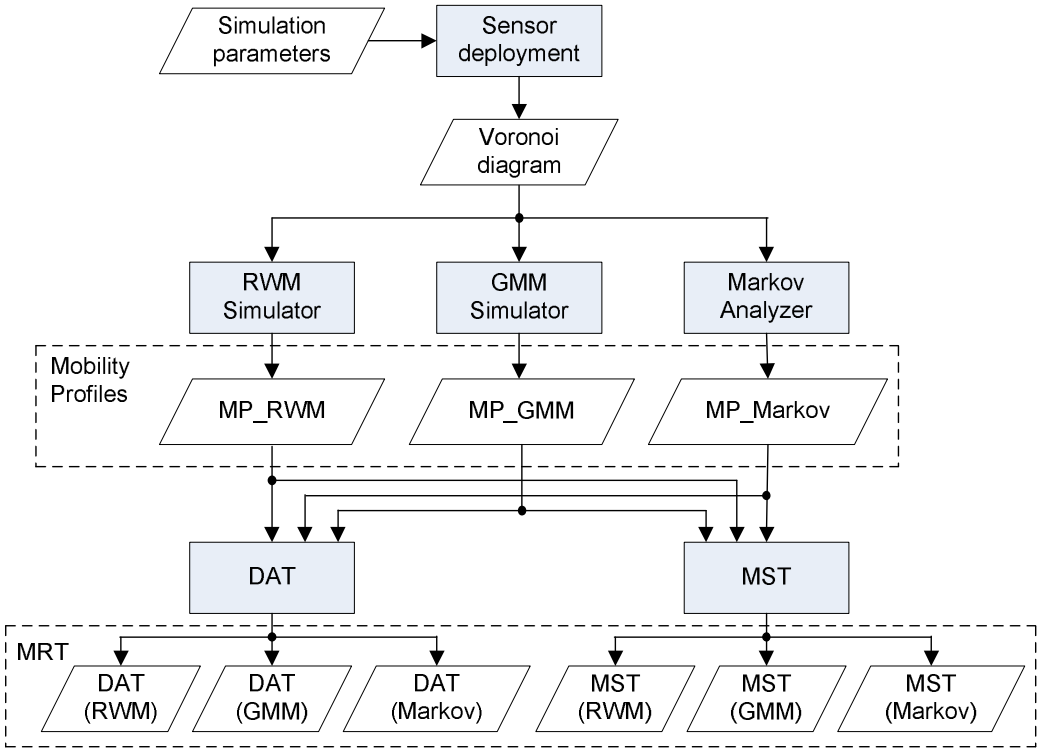

Figure 7: The process of MRT constructions

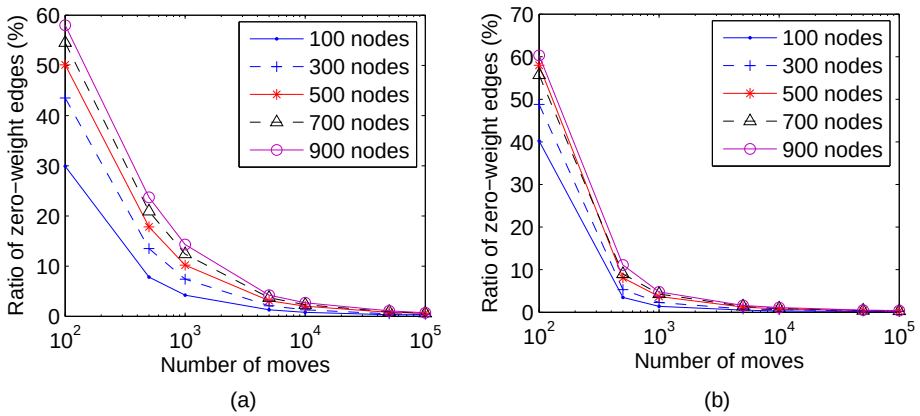

Figure 8: Ratio of zero-weight edges in (a) MP_RWM and (b) MP_GMM. All values were averaged over 100 runs.

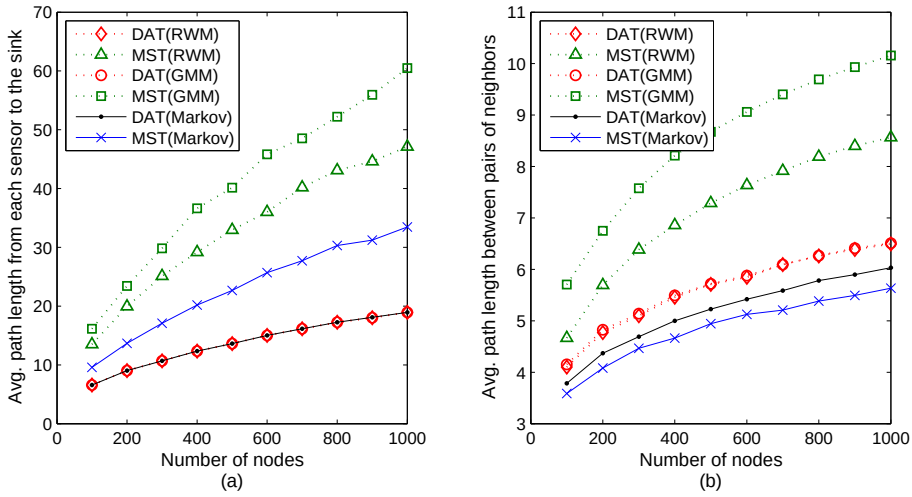

Figure 9: (a) Average node level. (b) Average reporting length. All values were averaged over 100 runs.

4.3 Update Cost

We measured update costs associated with the six types of MRTs. After all six MRTs had been constructed for a given sensor deployment, the simulator used RWM or GMM to move an object around the deployment field and counted the number of update messages triggered by the movement. The movement lasted 10,000 units of time and repeated 10 times for a given sensor deployment. The final result for a given sensor population and a specific MRT was averaged over 100 random sensor deployments.

We conducted two types of experiments: *regular* and *mix-up* (Fig. 10). In regular experiments, the mobility model used in the pre-run that generated statistical mobility profiles was also used to move an object for cost measurements. Fig. 11 shows the results of regular experiments. In mix-up experiments, however, we used different mobility models in generating statistical mobility profiles and in directing object movements in cost measurements. This is to test the robustness of profilings. Fig. 12 shows the results of mix-up experiments.

From these results we observed the following findings.

1. MST(Markov) performs the best in all settings.

2. DAT seems to be profile-insensitive. That is, trees generated by DAT have nearly identical costs, regardless of the mobility profile in use. Therefore, in case of using DAT, MP_Markov can replace statistical profiling (MP_RWM or MP_GMM) without any performance degradation.

3. MST is profile-sensitive, and its cost is lower with MP_Markov than with either MP_RWM or MP_GMM.

To confirm the profile-insensitivity of DAT, for each topology setting a uniform-weight mobility profile was used to generate a DAT tree called DAT(Uniform). Since edges have identical weights, DAT simply chooses one of the deviation-free edges for every node. DAT(Uniform) was then compared with DAT(Markov), DAT(RWM), and DAT(GMM). See Fig. 13. Interestingly, the use of uniform-weight mobility profile may slightly raise or lower update costs, depending on

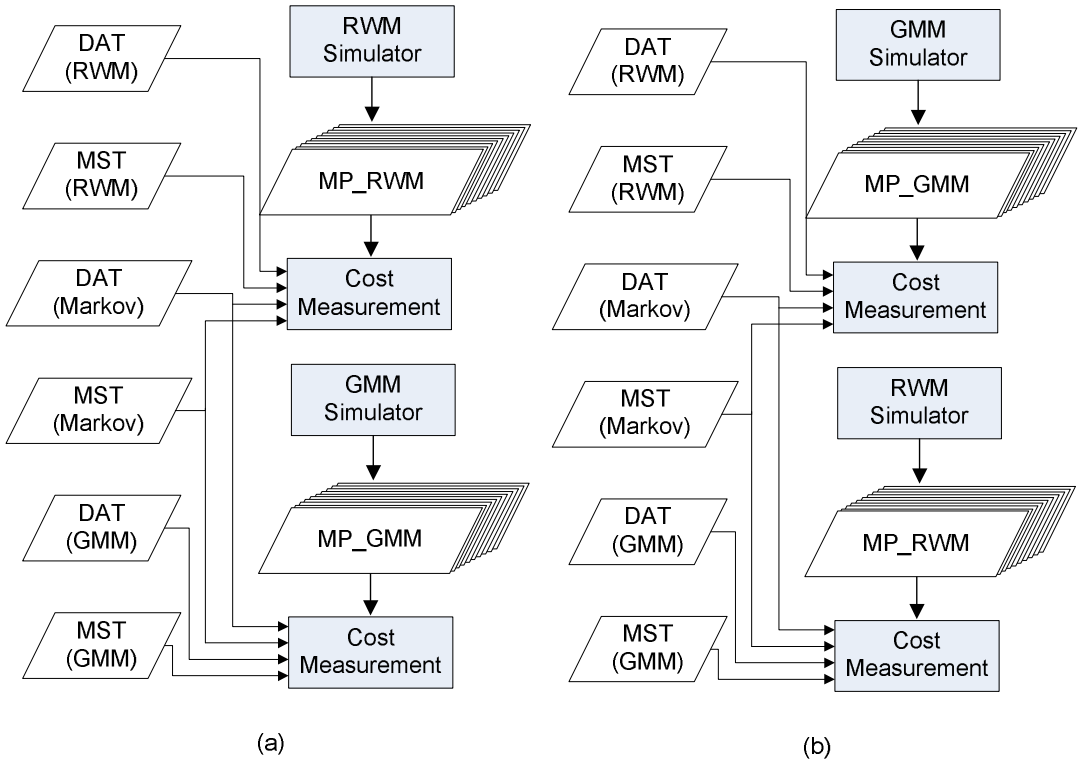

Figure 10: Two types of experiments. (a) regular (b) mix-up

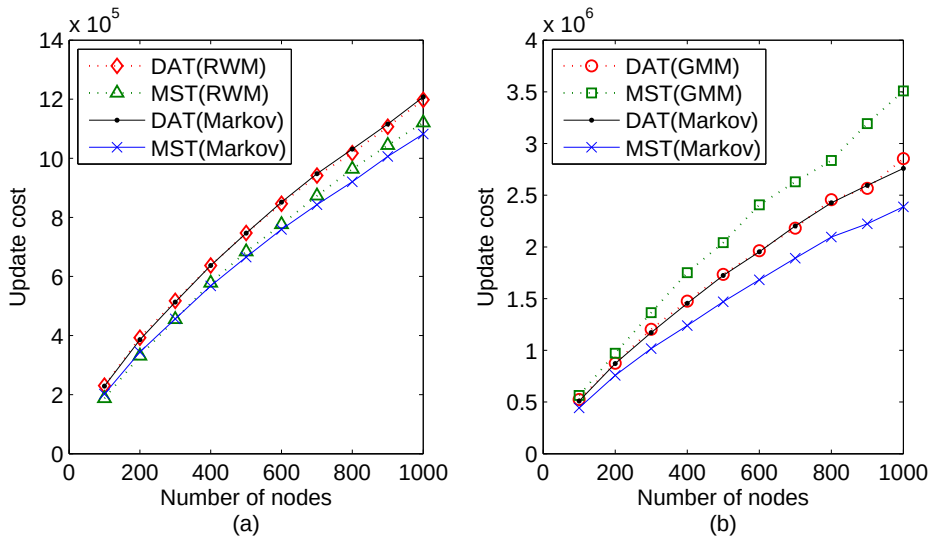

Figure 11: Update costs of the regular experiments with the mobility model used in cost measurements being (a) RWM and (b) GMM.

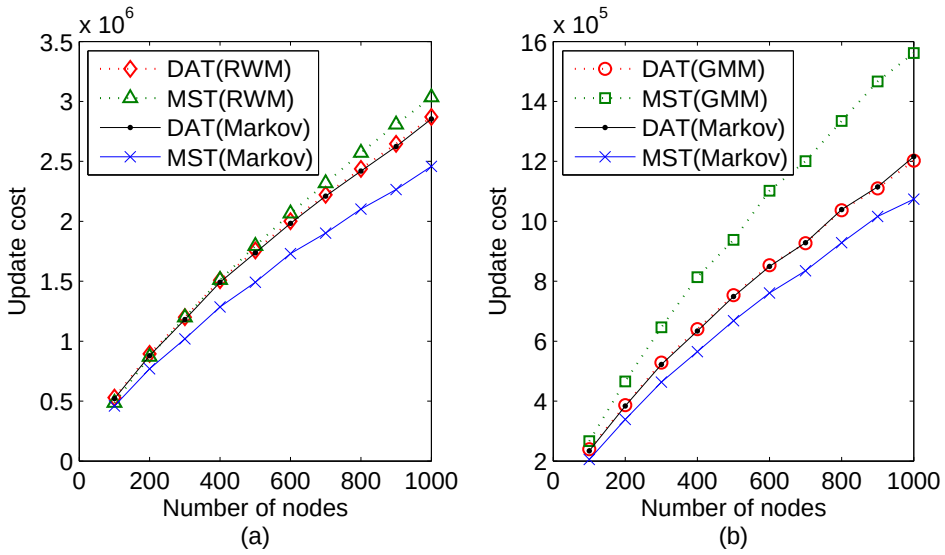

Figure 12: Update costs of the mix-up experiments with the mobility model used in cost measurements being (a) GMM and (b) RWM.

which mobility model was used in measuring costs. When RWM was used, DAT(Uniform) had higher costs than the others. On the other hand, DAT(Uniform) had the lowest costs among all when GMM was used. Theses results indicate that DAT is not purely profile-insensitive, though it performed similarly with either statistical or analytical profiling. Moreover, sophisticated profilings may not have advantage over the uniform-weight profiling in certain cases.

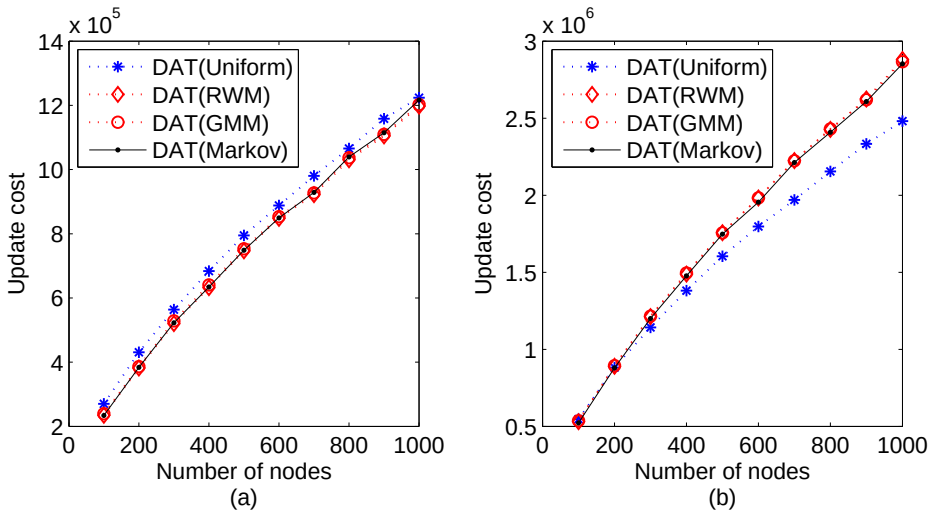

Figure 13: Update cost comparisons among various MRTs generated by DAT, where (a) RWM or (b) GMM was used in cost measurements.

4.4 Query and Overall Costs

To test the quality of the proposed profiling when working with QCR, we applied QCR to every MRT and measured query and total costs of the resultant tree. As QCR needs both query

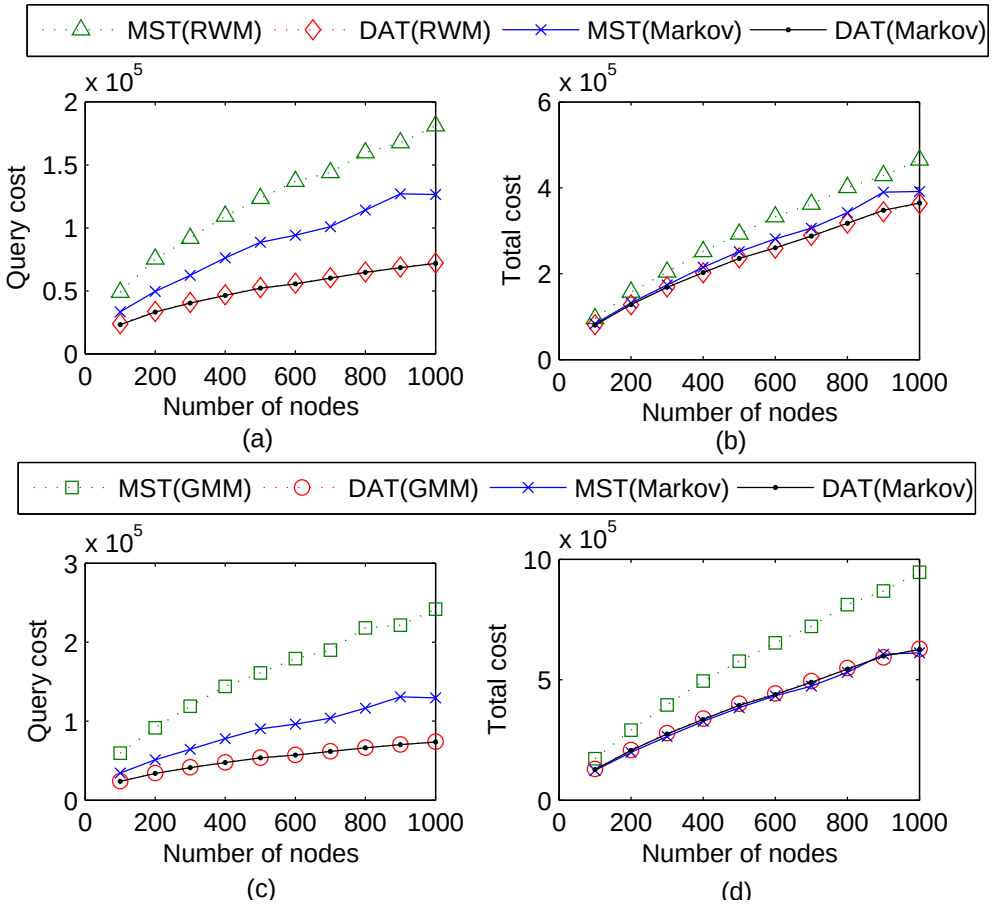

Figure 14: Query and total costs after applying QCR to all MRTs. The mobility model used in (a) and (b) is RWM while that in (c) and (d) is GMM. Mean event arrival rate is 0.2 event per unit of time.

and mobility profiles for tree adjustments, we modified simulators to produce a query profile as well as a statistical mobility profile in a single pre-run. Query event generations, which are independent of the object's mobility pattern, form a Poisson process with mean arrival rate $\lambda = 0.2$, 1.0 or 5.0. All experiments are regular type.

Figure 14 shows the result with $\lambda = 0.2$ while Fig. 15 shows that with $\lambda = 1.0$. The result with $\lambda = 5.0$ exhibits similar behavior as those shown here but is not included due to space limitation. From these results the following conclusions can be drawn.

1. Trees generated by DAT had both lower query costs and lower total costs than those generated by MST.

2. The costs of DAT(Markov) are hardly distinguishable from those of DAT(RWM) and DAT(GMM). We once again confirm that, in case of using DAT, the proposed analytical profiling can replace statistical profiling without any performance degradation.

3. For MST, the cost with MP_Markov is lower than that with MP_RWM or MP_GMM. Therefore, in case of using MST, the proposed analytical profiling outperforms statistical profilings.

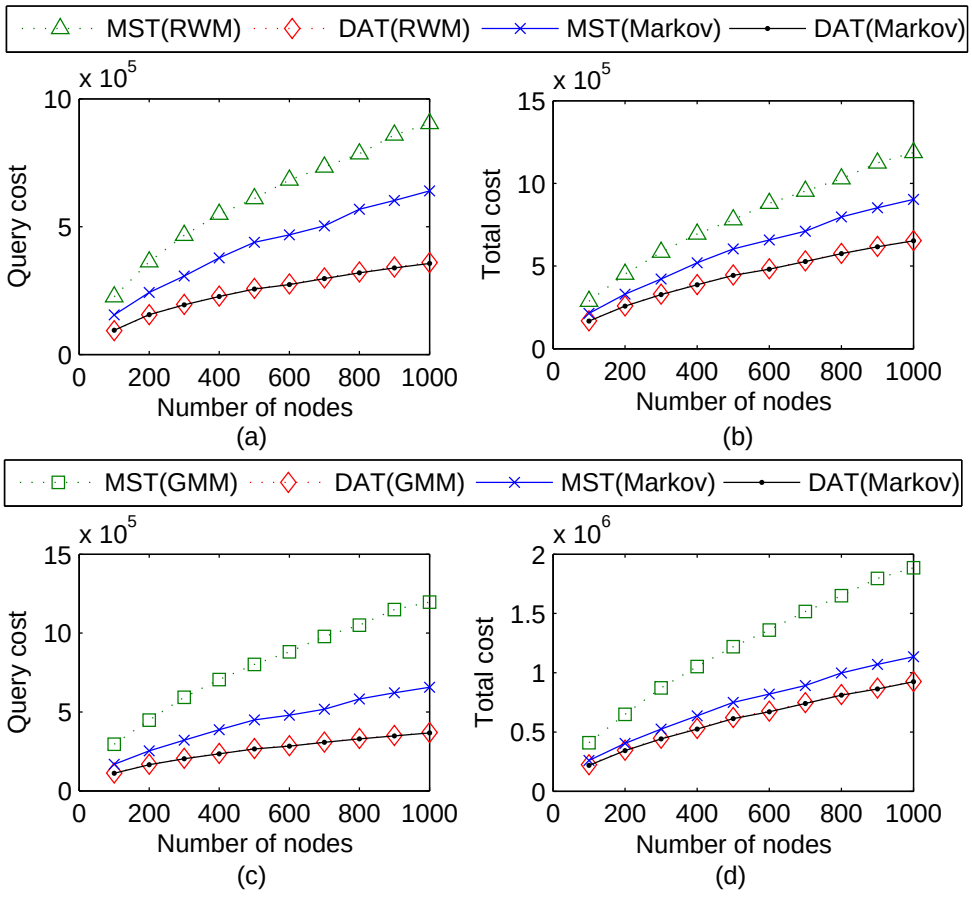

Figure 15: Query and total costs after applying QCR to all MRTs. The mobility model used in (a) and (b) is RWM while that in (c) and (d) is GMM. Mean event arrival rate is 1.0 event per unit of time.

5 Conclusions

We have proven that the OMRT problem, which seeks an optimal message report tree from a given mobility profile, has no polynomial-time solution. Two heuristic designs for the OMRT problem, DAT and MST, have been discussed. A quantitative analysis on the performance gain due to tree adjustments has also been given. We have analyzed object crossing rates between any two sensors deployed in a closed region. The analytical results serve as a mobility profile that is essential to tree-construction algorithms. Extended simulations have been conducted to examine the performance of the proposed analytic profiling. The results show that in case of MST, the proposed analytic profiling outperforms statistical profiling. In case of DAT, the proposed analytic profiling performs the same as the statistical profiling. In short, the proposed analytic profiling can replace an otherwise statistical profiling without noticeable performance degradation, which is valuable when historical running traces are unavailable.

References

[1] 3GPP. Selection procedures for the choice of radio transmission technologies of the UMTS (TS30.03 v3.2.0), April 1998.

[2] Christian Bettstetter. Smooth is better than sharp: A random mobility model for simulation of wireless networks. In *Proc. 4th ACM Int'l Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, pages 19–27, Rome, Italy, July 2001.

[3] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34, October 2000.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[5] T.C. Hu. Optimum communication spanning trees. *SIAM Journal on Computing*, 3(3):188–195, 1974.

[6] David B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In Tomasz Imieliński and Henry F. Korth, editors, *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[7] D.S. Johnson, J.K. Lenstra, and A.H.G. Rinnooy Kan. The complexity of the network design problem. *Networks*, pages 279–285, 1978.

[8] H. T. Kung and D. Vlah. Efficient location tracking using sensor networks. In *Proc. IEEE WCNC*, pages 1954–1961, March 2003.

[9] Yu Li and Youcef Bouchebaba. A new genetic algorithm for the optimal communication spanning tree problem. In *Lecture Notes in Computer Science 1829*, pages 162–173. Springer-Verlag, 1999.

[10] B. Liang and Zygmunt J. Haas. Predictive distance-based mobility management for multidimensional PCS networks. *IEEE/ACM Trans. on Networking*, 11:718–732, October 2003.

[11] Chih-Yu Lin, Wen-Chih Peng, and Yu-Chee Tseng. Efficient in-network moving object tracking in wireless sensor networks. *IEEE Trans. on Mobile Computing*, 5(8):1044–1056, August 2006.

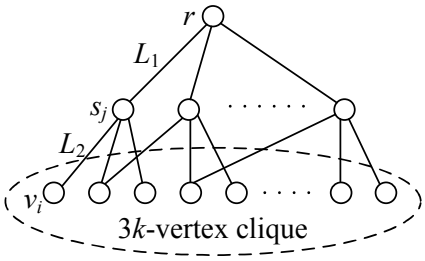

Figure 16: The mobility profile corresponding to an instance of the X3C problem. Edges in the clique are not shown.

[12] Bing-Hong Liu, Wei-Chieh Ke, Chin-Hsien Tsai, and Ming-Jer Tsai. Constructing a message-pruning tree with minimum cost for tracking moving objects in wireless sensor networks is NP-complete and an enhanced data aggregation structure. *IEEE Trans. Comput.*, pages 849–863, January 2008.

[13] Guoqiang Mao, Baris Fidan, and Brian D.O. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10):2529–2553, July 2007.

[14] David Peleg and Eilon Reshef. Deterministic polylog approximation for minimum communication spanning trees. In *Proc. 25th Int'l Colloquium on Automata, Languages and Programming*, pages 670–682, 1998.

[15] S. M. Ross. *Stochastic Processes.* John Wiley & Sons, New York, 2nd edition, 1996.

[16] Y. Xu and W.-C. Lee. On localized prediction for power efficient object tracking in sensor networks. In *Proc. Intl Workshop Mobile Distributed Computing*, May 2003.

[17] Wensheng Zhang and Guohong Cao. DCTC: Dynamic convoy tree-based collaboration for target tracking in sensor networks. *IEEE Trans. on Wireless Communications*, 3(5):1689–1701, September 2004.

Appendix

We prove Theorem 1 by reducing the Exact Cover by 3-sets (X3C) problem [4] to the BCMRT problem. Given a set $Q = \{q_1, q_2, \ldots, q_{3k}\}$ and a collection $\chi = \{X_1, X_2, \ldots, X_p\}$ of 3-element subsets of Q, the X3C problem asks whether there exists a subcollection $\chi' \subseteq \chi$ such that every element of Q occurs in exactly one member of χ'.

For an instance of the X3C problem, we construct a mobility profile M as in Fig. 16, in which v_i and s_j are for q_i and X_j, respectively. There is another vertex r connected to every s_j with an edge of weight L_1. There is an edge (v_i, s_j) of weight L_2 whenever $q_i \in X_j$. All v_i's are connected by unit-weight edges to form a clique. L_1 and L_2 are large enough so that these edges are enforced to be in the MRT if it exists. We claim that there exists an exact cover of the X3C problem *if and only if* the BCMRT problem has a solution with cost

$$c^* = pL_1 + 3k(L_2) + 3(3p - 3k)L_2 + 3k(2 \times 2 + (3k - 3) \times 4)/2 = pL_1 + (9p - 6k)L_2 + 3k(6k - 4).$$

For the convenience of discussion, we divide the node set into three subsets: $V = \{v_i | \forall i\}$, $S = \{s_j | \forall j\}$, and $R = \{r\}$. Let $E_1 = \{(r, s_j) | 1 \leq j \leq p\}$, $E_2 = \{(s_j, v_i) | q_i \in X_j\}$ and

$E_3 = \{(v_i, v_j) | 1 \leq i < j \leq 3k\}$ be three types of node pairs that are connected by edges in M. Since an exact cover does not exist if $\exists v_i \in V, \forall s_j \in S : (s_j, v_i) \notin E_2$, we preclude such trivial cases in the rest of the proof.
(the *only-if* part) Suppose that there exists an exact cover χ'. We show that there exists a spanning tree T with cost c^*. The edge set of T consists of all edges in E_1 and edges (s_j, v_i) for all $X_j \in \chi'$ and $q_i \in X_j$. Since χ' is an exact cover, T is a spanning tree. The cost of T consists of three distinct parts.

- Cost associated with E_1: This part of cost is pL_1.
- Cost associated with E_2: There are $3p$ node pairs in E_2, $3k$ of which are included as edges in T and each of the others has a distance of 3 (with common ancestor r). The total cost is therefore $3kL_2 + 3(3p - 3k)L_2$.
- Cost associated with E_3: Each $v_i \in V$ shares with other two nodes in V a common parent $s_j \in S$ and with each of the remaining $3k - 3$ nodes in V the common parent r. Each of the former node pairs has a distance of 2 while the distance of the latter is 4. The cost is therefore $3k(2 \times 2 + (3k - 3) \times 4)/2 = 3k(6k - 4)$.

Clearly, the total cost of T is c^*.
(the *if* part) M has $p + 3k + 1$ nodes, and therefore any spanning tree has exactly $p + 3k$ edges. The weights L_1 and L_2 are set large enough such that if the OMRT has cost c^* it must contain all the p edges in E_1 and $3k$ edges in E_2, and therefore no edge in E_3 is in the OMRT. Suppose that T^* is the OMRT of cost c^*. To our aim, we assign $L_1 = (9p - 6k)L_2 + 3k(6k - 4)$. If any edge (r, s_j) is not in T^*, the cost is larger than $(p + 2)L_1 > c^*$ since the distance from r to s_j is at least three. Since all edges in E_1 are in T^*, no node in V can be connected to more than one node in S. Otherwise it forms a cycle.

For convenience, we root T^* at r. By the above discussion, all nodes in S are at level one. Next, we set $L_2 = 9k$ and claim that all nodes in V are at level two, which also means that no edge in E_3 can be an edge of T^*. Suppose by contradiction that there exists a node v in level 3. We shall show that there is a way to reduce the cost of ${}^*T$, which contradicts the optimality of T^*. Let v_i be the parent of v and s_j the parent of v_i. Also assume that (v, s) is an edge in E_2. We replace edge (v_i, v) with (s, v) and make all children of v, if any, the children of v_i. The cost between v and all nodes in S is reduced by at least L_2, while the cost between v and all nodes in V is increased by at most $3(3k - 1)$ since the new distance from v to v_i is 4.

We have shown that the entire E_1 is in T^* and the other tree edges are from E_2. The cost in the cuts (R, S) and (S, V) is therefore $pL_1 + (9p - 6k)L_2$ no matter which of the $3k$ edges in E_2 are chosen. For each pair of nodes in V, the update cost is 2 if the node pair is connected to the same parent and 4 otherwise. Since each s_j can have at most three children, the cost is lower bounded by $3k(6k - 4)$, and it happens only when k nodes in S have exactly three children and others have none. In this case there is an exact cover for the X3C problem.