Maximizing Lifetime of Sensor Surveillance Systems

Hai Liu, Xiaohua Jia, Peng-Jun Wan, Chih-Wei Yi, S. Kami Makki, and Niki Pissinou

***Abstract*—This paper addresses the maximal lifetime scheduling problem in sensor surveillance systems. Given a set of sensors and targets in an area, a sensor can watch only one target at a time, our task is to schedule sensors to watch targets and forward the sensed data to the base station, such that the lifetime of the surveillance system is maximized, where the lifetime is the duration that all targets are watched and all active sensors are connected to the base station. We propose an optimal solution to find the target-watching schedule for sensors that achieves the maximal lifetime. Our solution consists of three steps: 1) computing the maximal lifetime of the surveillance system and a workload matrix by using the linear programming technique; 2) decomposing the workload matrix into a sequence of schedule matrices that can achieve the maximal lifetime; and 3) determining the sensor surveillance trees based on the above obtained schedule matrices, which specify the active sensors and the routes to pass sensed data to the base station. This is the first time in the literature that the problem of maximizing lifetime of sensor surveillance systems has been formulated and the optimal solution has been found.**

***Index Terms*—Energy efficiency, lifetime, scheduling, sensor network, surveillance system.**

I. Introductions

A SENSOR SURVEILLANCE system consists of a set of wireless sensor nodes (sensors for short) and a set of targets to be monitored. The sensors collaborate with each other to watch or monitor the targets and pass the sensed data to the base station. The sensors are powered by batteries and have a stringent power budget [1], [2]. The nature of the sensor surveillance system requires a long lifetime. In this paper, we discuss a maximal lifetime problem in sensor surveillance systems. Given a set of targets, a set of sensors, and a base station (BS) in an area, the sensors are used to watch the targets and collect sensed data to the BS. Each sensor has an initial energy reserve, a fixed surveillance range, and an adjustable transmission range. A sensor can watch at most one target at a time and a target should be watched by a sensor at any time. The problem is to schedule a subset of sensors to be active at a time to watch the targets and find the routes for the active sensors to send data back to the BS, such that the lifetime of the entire sensor network is maximized. The lifetime is the duration up to the time when there exists one target that can no longer be watched by any sensors or data cannot be forwarded to the BS any longer due to the depletion of energy of the sensor nodes. We assume the positions of targets, sensors, and the BS are given in prior and static. The location information of both sensors and targets can be obtained via a distributed monitoring mechanism [3] or the scanning method [4], [5] by the BS.

The solution to this problem includes two parts: scheduling the sensors to watch targets and routing the sensed data to the BS. The schedule and the routes are pre-computed at the BS, and they are disseminated to sensors by the BS at the system initialization. When the system starts operation, all sensors work according to the schedule, such as when and for what duration to sleep, watch targets, or relay messages.

There are many applications of this type of surveillance systems. For example, sensors equipped with camera are used to guard cargo containers to prevent them from being tampered during the long journey of shipment or during the storage at a port. Another example is the use of sensors to monitor some hot spots in a region or in a building. In these examples, sensors and targets are static, and each sensor can only focus on watching one target at a time. For the applications in which one sensor can watch multiple targets simultaneously, some work on sensor scheduling has been done in [2] and [6], where sensors are scheduled to work in turn such that a given area can be covered fully [2] or partially [6] and the system lifetime is maximized.

The rest of this paper is organized as follows. Section II is related work and Section III is the problem definition. Section IV presents our solution which consists of three parts. Section IV-A gives a linear programming formulation that is used to compute the maximal lifetime of the surveillance system. In Section IV-B, we show that the maximal lifetime is achievable, and give the algorithms for scheduling the sensors to watch targets. Section IV-C discusses surveillance trees for routing sensed data to BS. Section V gives a numeric example solved by using our method and simulation results. We conclude our work in Section VI.

II. Related Work

There are two major techniques for maximizing the sensor network lifetime: the use of energy efficient routing and the introduction of sleep/active modes for sensors.

Extensive research has been done on energy efficient data gathering and information dissemination in sensor networks. Some well-known energy efficient protocols were developed,

Manuscript received April 28, 2005; revised April 14, 2006; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor N. Shroff. This work was supported in part by Hong Kong Research Grant Council under Grant No. CityU 114505, NSF China Grant No. 60633020, and NSF CCR-0311174.

H. Liu and X. Jia are with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: liuhai@cs.cityu.edu.hk; jia@cs.cityu.edu.hk).

P.-J. Wan is with the Department of Electrical Engineering and Computer Science, Illinois Institute of Technology, Chicago, IL 60616 USA (e-mail: wan@cs.iit.edu).

C.-W. Yi is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan 300, R.O.C. (e-mail: yi@cs.nctu.edu.tw).

S. K. Makki is with the Department of Electrical Engineering and Computer Science, University of Toledo, Toledo, OH 43606-3390 USA (e-mail: kmakki@eng.utoledo.edu).

N. Pissinou is with the Telecommunications and Information Technology Institute, Florida International University, Miami, FL 33174 USA (e-mail: pissinou@fiu.edu).

Digital Object Identifier 10.1109/TNET.2007.892883

such as Directed Diffusion [7], LEACH [8], PEGASIS [9], and ACQUIRE [10]. Directed Diffusion is regarded as an improvement over the SPIN [11] protocol that used a proactive approach for information dissemination. LEACH organizes sensor nodes into clusters to fuse data before transmitting to the BS. PEGASIS improved the LEACH by considering both metrics of energy consumption and data-gathering delay. In [12], an analytical model was proposed to find the upper bound of the lifetime of a sensor network, given the surveillance region and a BS, the number of sensor nodes deployed and initial energy of each node. Some routing schemes for maximizing network lifetime were presented in [13]. In [14], an analytic model was proposed to analyze the tradeoff between the energy cost for each node to probe its neighbors and the routing accuracy in geographic routing, and a localized method was proposed. In [15] and [16], linear programming (LP) formulation was used to find energy-efficient routes from sensor nodes to the BS, and approximation algorithms were proposed to solve the LP formulation.

Another important technique used to prolong the lifetime of sensor networks is the introduction of switch on/off modes for sensor nodes. J. Carle *et al.* did a good survey in [17] on energy-efficient area monitoring for sensor networks. They pointed out that the best method for conserving energy is to turn off as many sensors as possible, while still keeping the system functioning. An analytical model was proposed in [18] to analyze the system performance, such as network capacity and data delivery delay, against the sensor dynamics in on/off modes. A node scheduling scheme was developed in [19]. This scheme schedules the nodes to turn on or off without affecting the overall service provided. A node decides to turn off when it discovers that its neighbors can help it to monitor its monitoring area. The scheduling scheme works in a localized fashion where nodes make decisions based on its local information. Similar to [19], the work in [20] defined a criterion for sensor nodes to turn themselves off in surveillance systems. A node can turn itself off if its monitoring area is the smallest among all its neighbors and its neighbors will become responsible for that area. This process continues until the surveillance area of a node is smaller than a given threshold. A deployment of a wireless sensor network in the real world for habitat monitoring was discussed in [21]. A network consisting of 32 nodes was deployed on a small island to monitor the habitat environment. Several energy conservation methods were adopted, including the use of sleep mode, energy-efficient communication protocols, and heterogeneous transmission power for different types of nodes.

We use both of the above-mentioned techniques to maximize the network lifetime in our solution. We find the optimal schedule to switch on/off sensors to watch targets in turn, and we find the optimal routes to forward data from sensor nodes to the BS.

III. System Model and Problem Statement

We consider a set of targets and a set of sensors that are used to watch targets and collect information. We first introduce the following notations:

B	base station whose energy is unbounded.
S	set of sensors, and $n = \|S\|$.
T	set of targets, and $m = \|T\|$.
$S(j)$	set of sensors that are able to watch target $j, j = 1, 2, \ldots, m$.
$T(i)$	set of targets that are within the surveillance range of sensor $i, i = 1, 2, \ldots, n$.
$N(i)$	set of neighbors of sensor $i, i = 1, 2, \ldots, n$.
E_i	initial energy reserve of sensor $i, i = 1, 2, \ldots, n$.
d_{ij}	distance between sensor i and $j, i, j = 1, 2, \ldots, n, B$.
e^T, e^R	energy required for transmitting and receiving one unit data, respectively.
e^S	energy required for watching a target per unit time.
R	data rate generated from sensors while watching targets.

Notice that $S(i)$ may overlap with $S(j)$ for $i \neq j$, and $T(i)$ may overlap with $T(j)$ for $i \neq j$. There are two requirements for sensors watching targets:

1) Each sensor can watch at most one target at a time.
2) Each target should be watched by one sensor at anytime.

The problem of our concern is, for given S, T and B, to find a schedule that meets the above two requirements for sensors watching targets, such that the lifetime of network is maximized. The lifetime of network is the length of time until there exists a target j such that all sensors in $S(j)$ run out their energy or the sensed data cannot be forwarded back to the BS due to the disconnection of the network.

IV. Our Solutions

We solve the problem in three steps. First, we compute the upper bound on the maximal lifetime of the system and find a workload matrix and data flows of sensors. Second, we completely decompose the workload matrix into a sequence of schedule matrices without compromising the obtained maximal lifetime. Finally, we determine a sensor surveillance tree for each schedule matrix that specifies the active sensors and the routes to forward sensed data to the BS.

A. Find Maximal Lifetime

We use linear programming (LP) technique to find the maximal lifetime of the system. Let L denote the lifetime of the surveillance system. We introduce two variables:

x_{ij}	total time sensor i watching target j.
f_{ij}	amount of data transmitted from sensor i to sensor j (the receiver can be the BS).

The problem of finding the maximal lifetime for sensors watching targets can be formulated as the following:

$$\text{Objective}: \text{Max } L$$

$$\text{s.t.} \quad \sum_{i \in S(j)} x_{ij} = L \quad \forall\, j \in T; \tag{1}$$

$$\sum_{j \in T(i)} x_{ij} \leq L \quad \forall\, i \in S; \tag{2}$$

$$e^s \sum_{j \in T(i)} x_{ij} + e^T \sum_{j \in N(i) \cup \{B\}} (d_{ij})^\alpha f_{ij} + e^R \sum_{j \in N(i)} f_{ji} \leq E_i \quad \forall\, i \in S; \tag{3}$$

$$R\sum_{j\in T(i)} x_{ij} + \sum_{j\in N(i)} f_{ji} = \sum_{j\in N(i)\cup\{B\}} f_{ij} \quad \forall\, i \in S; \tag{4}$$

$$x_{ij} \geq 0, \quad f_{ij} \geq 0. \tag{5}$$

Equation (1) specifies that for each target j in T, the total time that sensors watch it is equal to the lifetime of the system. That is, each target should be watched throughout the lifetime.

Inequality (2) implies that for each sensor i in S, the total watching time should not exceed the lifetime of the system.

Inequality (3) implies that the total energy cost of a sensor node shall not exceed its initial energy reserve. There are three components of energy cost, which are the cost for sensing data (i.e., watching targets), the cost for transmitting data (which is dependent on the transmission distance), and the cost for receiving data.

Equation (4) is for flow conservation. It implies that for each sensor i in S, the total amount of data sensed and data received should be equal to the amount of data transmitted.

The above formulation is a typical LP formulation, where x_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq m$, and $f_{ij}, i, j = 1, 2, \ldots, n, B$, are real number variables and the objective is to maximize L. The optimal results of x_{ij}, f_{ij}, and L can be computed in polynomial time.

Notice that L, obtained from computing the above LP formulation, is the upper bound on the lifetime, and each x_{ij} specifies only the total time that sensor i should watch target j in order to achieve this upper bound L. Each f_{ij} specifies only the total amount of data transmitted from sensor i to sensor j. We need to find a schedule that specifies from what time up to what time which sensor watches which target and through which route to pass the sensed data to the BS. In the next two steps we will find the schedule and routes that will finally achieve the optimal lifetime L.

The values of x_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq m$, obtained from the LP, can be represented as a matrix:

$$X_{n\times m} = \begin{bmatrix} x_{11}x_{12}\ldots x_{1m} \\ x_{21}x_{22}\ldots x_{2m} \\ \ldots \\ x_{n1}x_{n2}\ldots x_{nm} \end{bmatrix}_{n\times m}.$$

We call matrix $X_{n\times m}$ the ***workload*** matrix, for it specifies the total length of time that a sensor should watch a target. This workload matrix has two important properties:

1) The sum of all elements in each column is equal to L (from (1) in the LP formulation).
2) The sum of all elements in each row is less than or equal to L (from in (2) in the LP formulation).

In the next step, we will find the schedule for sensors to watch targets based on the workload matrix.

B. Decompose Workload Matrix

The lifetime of the surveillance system can be divided into a sequence of sessions. In each session, a set of sensors are scheduled to watch their corresponding targets; and in the next session, another set of sensors are scheduled to work (some sensors may work continuously for multiple sessions). Suppose a sensor will not switch to watch another target within a session. Thus, the schedule of sensors during a session can be represented as a matrix. In this matrix, there is only one positive number in each column, meaning each target should be watched by one sensor at a time; and at most one positive number in each row, meaning each sensor can watch at most one target at a time and there is no switching to watch other targets in a session. Furthermore, all the non-zero elements in this matrix have the same value, which is the time duration of this session. Now, our task becomes to decompose the workload matrix into a sequence of schedule matrices of sessions, represented as

$$\begin{aligned} \begin{bmatrix} x_{11}x_{12}\ldots x_{1m} \\ x_{21}x_{22}\ldots x_{2m} \\ \ldots \\ x_{n1}x_{n2}\ldots x_{nm} \end{bmatrix}_{n\times m} &= \begin{bmatrix} 0c_10\ldots 0 \\ c_100\ldots 0 \\ \ldots \\ 00c_1\ldots 0 \end{bmatrix} + \begin{bmatrix} c_200\ldots 0 \\ 000\ldots c_2 \\ \ldots \\ 0c_20\ldots 0 \end{bmatrix} \\ &\quad + \cdots + \begin{bmatrix} 000\ldots c_t \\ 000\ldots 0 \\ \ldots \\ 0c_t0\ldots 0 \end{bmatrix} \\ &= P_1 + P_2 + \cdots + P_t \end{aligned} \tag{6}$$

where $c_i, i = 1, 2, \ldots, t$, is the length of time of session i, and t the total number of sessions. We call this sequence of matrices P_i, $i = 1, 2, \ldots, t$, the ***schedule*** matrices. Each schedule matrix, say P_i for session i, has three properties: 1) all elements in it are either "0" or c_i; 2) each column has exactly one non-zero element; and 3) each row has at most one non-zero element (it could be all "0", indicating the sensor has no watching duty in this session).

Next, we discuss how to decompose the workload matrix into a sequence of schedule matrices. In general sensor surveillance systems, the number of sensors is greater than the number of targets, i.e., $n \geq m$. We first consider a special case of $n = m$. Then, we extend the result to the general case of $n > m$.

1) A Special Case $n = m$: We consider the case $n = m$. Let R_i and C_j denote the sum of row i and the sum of column j in the workload matrix, respectively. According to (1) and (2) of the LP formulation, we have

$$C_j = L, \quad j = 1, 2, \ldots, m \tag{7}$$

$$R_i \leq L, \quad i = 1, 2, \ldots, n. \tag{8}$$

Furthermore, since $\sum_{i=1}^{n} R_i = \sum_{j=1}^{m} C_j = m \times L$ and $n = m$, we have

$$\sum_{i=1}^{n} R_i = n \times L. \tag{9}$$

Combining (8) and (9), we have

$$R_i = L, \quad i = 1, 2, \ldots, n. \tag{10}$$

From (7) and (10), we have

$$R_i = C_j, \quad 1 \leq i, j \leq n. \tag{11}$$

Equation (11) gives an important feature of the workload matrix when $n = m$ that the sum of each column is equal to the sum of each row. This feature will guarantee the possibility of decomposing the workload matrix into schedule matrices in Theorem 1.

To decompose the workload matrix $X_{n\times m}$ into a sequence of schedule matrices, our basic idea is to represent $X_{n\times m}$ as a bipartite graph $G(S \cup T, E)$, where one side are sensors $S = (s_1, s_2, \ldots, s_n)$ and the other are targets $T = (t_1, t_2, \ldots, t_m)$. For each non-zero element x_{ij} in the workload matrix, there is an edge between s_i and t_j and the weight of the edge is x_{ij}. Since $n = m$, every sensor has a target to watch in any session. This means n sensors exactly match n targets in each session, which can be represented as a perfect matching in the bipartite graph G. Thus, each schedule matrix is corresponding to a perfect matching. The problem of decomposing the workload matrix is transformed into the problem of finding perfect matchings in G.

The decomposing process is as follows. Each time, we compute a perfect matching M in the bipartite graph. M has exactly n edges, which defines the pairs of sensor-target watching. Let c_i be the smallest weight of the n edges in M. We deduct c_i from the weight of the n edges in M and remove the edges whose weight becomes zero. The schedule matrix P_i, corresponding to this matching M, can be represented as $c_i \times P_i^*$, where P_i^* is the permutation matrix of M. (A permutation matrix is a square matrix that has only "0" and "1" elements, and each row and each column has exactly one "1" element). The schedule matrix $c_i \times P_i^*$ defines the sensor-target watching of a session and c_i is the duration of this session. This decomposing process is repeated until there is no perfect matching can be found any more in the bipartite graph.

For example, suppose we obtained a workload matrix $\begin{bmatrix} 2 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}$ for a system with three sensors and three targets from the LP (1)–(5). The matrix is first represented as a bipartite graph G shown in Fig. 1(a). We compute a perfect matching in G as shown in Fig. 1(b) and the smallest edge weight $c_i = 1$ in the matching. The schedule matrix corresponding to this matching is $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$. Then we deduct c_i from the weight of the three edges in the perfect matching and remove the edges (t_2, s_3) and (t_3, s_2) whose weight become zero. The resulting bipartite graph is shown in Fig. 1(c). We repeat the operation until all edges are removed from G.

The details of the algorithm for decomposing a workload matrix when $n = m$ are given below.

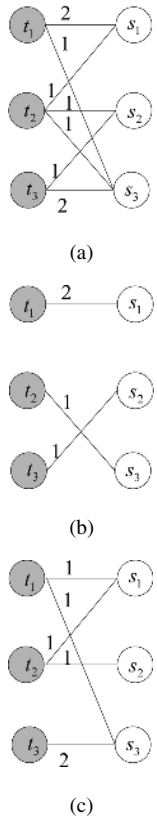


Fig. 1. (a) The bipartite graph. (b) A perfect matching. (c) The graph after deducting c_i.

DecomposeMatrix-nn Algorithm

Input: a workload matrix $X_{n\times n}$.

Output: a sequence of schedule matrices $P_1, P_2, \ldots, P_t$.

Begin

Construct a bipartite graph G from $X_{n\times n}$;

while there exist edges in G **do**

Find a perfect matching M on G;

Represent M as $P_i = c_i \times P_i^*$;

Deduct c_i from the n edges in M and remove edges whose weight is 0;

endwhile

Output $X_{n\times n} = P_1 + P_2 + \cdots + P_t$;

End

The above algorithm tries to decompose the matrix by using the technique of finding perfect matchings. There are two questions about this decomposability:

1) Does it guarantee that there exists a perfect matching in every round of the decomposition?
2) Does it guarantee that the number of decomposition rounds is bounded?

Theorem 1 and Theorem 2 will give answers to these two questions, respectively. To prove Theorem 1, we need the following lemma.

Lemma 1: For any square matrix $X_{n\times n}$ of nonnegative real numbers, if $R_i = C_j$ for $1 \le i, j \le n$, there exists a perfect matching in the corresponding bipartite graph, where R_i and C_j are the sum of row i and the sum of column j of $X_{n\times n}$, respectively.

Proof: Let L be the sum of all elements in a row in $X_{n\times n}$, and $A_{n\times n}$ denotes matrix $A_{n\times n} = \frac{1}{L} \times X_{n\times n}$. It is obvious that $A_{n\times n}$ is a doubly stochastic matrix [22], [23], where the sum of all elements in any row or column is equal to 1.

Now we prove the lemma by contradictory. Assuming there does not exist a perfect matching in the corresponding bipartite graph of $A_{n\times n}$, there does not exist n positive entries in $A_{n\times n}$ that no two entries in the same column or row. According to the König theorem [24], [25], we can cover all of the positive entries in the matrix with e rows and f columns, such that $e+f<n$. However, since the sum of all lines of $A_{n\times n}$ is equal to 1, it follows $n \le e+f < n$. This contradicts to the assumption. Lemma 1 is proved. □

Theorem 1: The *DecomposeMatrix-nn* algorithm can always find a perfect matching so long as there are edges in G.

Proof: For any workload matrix $X_{n\times n}$, according to (11), we have $R_i = C_j$ for $1 \le i, j \le n$. According to Lemma 1, there exists a perfect matching on the corresponding bipartite graph G.

Since in each round i, the *DecomposeMatrix-nn* algorithm computes a perfect matching and deducts c_i from the weight of the n edges in the perfect matching. It is equivalent to deducting a schedule matrix P_i from the workload matrix $X_{n\times n}$. Thus, the resulting matrix $X_{n\times n}$ (after deducting P_i) still holds the condition $R_i = C_j$ for $1 \le i, j \le n$. According to Lemma 1, there exists a perfect matching on G in every round of the decomposition process. This process stops at the last round where all the remaining edges in G make up an exact perfect matching, and they are all removed at this last round. Theorem 1 is proved. □

The following theorem states that the number of decomposition rounds can be bounded by using the *DecomposeMatrix-nn* algorithm.

Theorem 2: The workload matrix can be exactly decomposed into a sequence of schedule matrices by using the *DecomposeMatrix-nn* algorithm and the time complexity is $O(|E| \times n^3)$, where $|E|$ is the number of non-zero elements in $X_{n\times n}$.

Proof: According to theorem 1, a perfect matching can be found in G at each round of decomposition until there is no edge left in G.

Since at each time of finding a perfect matching, at least one edge in G is removed. Therefore, it takes at most $|E|$ number of rounds to remove all edges in G, where $|E|$ is the number of edges in G, which is the number of non-zero elements in $X_{n\times n}$. Furthermore, it takes $O(n^3)$ to find a perfect matching if we use depth-first search [26]. So the total time complexity is $O(|E| \times n^3)$. Theorem 2 is proved. □

Thus, the workload matrix can be successfully decomposed into a sequence of schedule matrices when $n = m$, where n, m are the number of sensors and the number of targets, respectively. In Section IV-B5, we will discuss the general cases of $n > m$ and propose a complete decomposition algorithm.

2) General Case $n > m$: When $n > m$, our idea is to transform the case to $n = m$ by introducing some dummy targets into the system. That is to "fill" the matrix $X_{n\times m}$ with some dummy columns to make it a square matrix of order n, such that the sum of all elements in each row is equal to the sum of all elements in each column.

Let $Z_{n\times}(n-m)$ be the dummy matrix, which has $(n-m)$ columns. By appending the columns of the dummy matrix to the right hand side of $X_{n\times m}$, the resulting matrix, denoted by $W_{n\times n}$, is in the form

$$W_{n\times n} = \begin{bmatrix} x_{11}x_{12}\ldots x_{1m} & z_{11}z_{12}\ldots z_{1n-m} \\ x_{21}x_{22}\ldots x_{2m} & z_{21}z_{22}\ldots z_{2n-m} \\ \ldots & \ldots \\ x_{n1}x_{n2}\ldots x_{nm} & z_{n1}z_{n2}\ldots z_{nn-m} \end{bmatrix}_{n\times n}.$$

To make matrix $W_{n\times n}$ having the features of (7) and (10), i.e., the sum of each column is equal to the sum of each row and equal to L, the dummy matrix $Z_{n\times}(n-m)$ should satisfy the following conditions:

$$1) R_i' = \sum_{j=1}^{n-m} z_{ij} = L - R_i \quad \forall\, i = 1,2,\ldots,n. \tag{12}$$

$$2)\ C_j' = \sum_{i=1}^{n} z_{ij} = L \quad \forall\, j = 1,2,\ldots,n-m. \tag{13}$$

We propose a simple algorithm to compute the dummy matrix $Z_{n\times}(n-m)$. The algorithm starts to assign values to the elements of $Z_{n\times}(n-m)$ from its top-left corner. Let R_i^- and C_j^- record the sum of the remaining undetermined elements of row i and column j, respectively, for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,n-m$. Initially, $R_i^- \leftarrow (L - R_i)$ and $C_j^- \leftarrow L$, where R_i and L are computed from matrix $X_{n\times m}$. The strategy of the algorithm is to assign the remaining sum of the row (or column), as much as possible, to an element without violating conditions (12) and (13), and assign the rest elements of the row (or column) to 0. Then, we move down to the next undetermined element from the top-left of the matrix. For example, we start with z_{11}. Now R_1^- is $(L - R_1)$ and C_1^- is L, i.e., $R_1^- < C_1^-$. Thus, we can assign R_1^- to z_{11}, and assign 0 to the rest of elements of row 1 (so condition (12) is met). Then, C_1^- should be updated to $(C_1^- - z_{11})$, because the remaining sum of column 1 now becomes $(C_1^- - z_{11})$ and this value is used to ensure that condition (13) will be met during the process. Suppose we now come to element z_{ij}, (i.e., elements of z_{kl}, for $k = 1,2,\ldots,i-1$ and $l = 1,2,\ldots,j-1$, are already determined so far). We compare R_i^- with C_j^-. There are three cases:

1) $C_j^- > R_i^-$: it means z_{ij} can use up the remaining value the sum of row i, i.e., R_i^-. Thus, $z_{ij} \leftarrow R_i^-$ and the rest elements of this row should be assigned to 0. So, all elements of row i have been assigned and condition (12) is met for row i.
2) $R_i^- > C_j^-$: it means z_{ij} can use up the remaining value the sum of column j, i.e., C_j^-. Thus, $z_{ij} \leftarrow C_j^-$ and the rest elements of this column should be assigned to 0, i.e., $z_{kj} = 0$, $k = 2,3,\ldots,n$. By doing so, all elements of column j have been assigned and condition (13) is met for column j.
3) $R_i^- = C_j^-$: we can determine elements in both row i and column j by $z_{ij} \leftarrow R_i^-$ and setting the rest elements in row i and in column j to 0. It is easy to see that condition (12) is met for row i and condition (13) is met for column j.

After determining each row (or column), we need to update C_j^- (or R_i^-), before moving to the next row (or column). Each step, we can determine the elements in one row (or column).

This process is repeated until all elements in $Z_{n\times(n-m)}$ are determined. The details of the algorithm are given below.

FillMatrix Algorithm

Input: a workload matrix $X_{n\times m}$.

Output: a filled matrix $W_{n\times n}$.

Begin

$R_i^- = L - R_i$, for $i = 1$ to n;

$C_j^- = L$, for $j = 1$ to $n - m$;

$i = 1; j = 1;$

while $(i \le n)$ && $(j \le n - m)$ **do**

if $C_j^- > R_i^-$ **then**

// determine elements in row i.

$z_{ij} = R_i^-;$

$z_{ik} = 0$, for $k = j + 1$ to $n - m;$

// set the rest of row i to 0.

$C_j^- = C_j^- = -z_{ij};$

$i = i + 1;$

else if $R_i^- > C_j^-$

//determine elements in column j.

$z_{ij} = C_j^-;$

$z_{kj} = 0$, for $k = i + 1$ to $n;$

// set the rest of column j to 0.

$R_i^- = R_i^- - z_{ij};$

$j = j + 1;$

else

//determine elements in both row i and column j.

$z_{ij} = R_i^-;$

$z_{ik} = 0$, for $k = j + 1$ to $n - m;$

$z_{kj} = 0$, for $k = i + 1$ to $n;$

$i = i + 1; j = j + 1;$

endwhile

Output $W_{n\times n} = [X_{n\times m} Z_{n\times}(n - m)];$

End

The following theorem claims the correctness of the *FillMatrix* Algorithm.

Theorem 3: For a given workload matrix $X_{n\times m}$, the *FillMatrix* Algorithm can compute the filled matrix $W_{n\times n}$, such that the sum of each column and the sum of each row have the properties defined in (7) and (10).

Proof: At the beginning of the *FillMatrix* Algorithm, row sums and column sums of the dummy matrix are initialized, and then the dummy matrix is worked out step by step to satisfy conditions (12) and (13). So we can prove a general case: given row sums R'_i and column sums C'_j of a matrix $Z_{n\times m}$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$, the proposed algorithm can compute all elements z_{ij} that satisfy conditions (12) and (13). We use the induction method to prove the theorem.

1) When $n = 1$, $m = 1$, according to the *FillMatrix* algorithm, since $C_1^- = R_1^-$, we have $z_{11} = R_1^- = C_1^- = R'_1 = C'_1$. The conditions (12) and (13) are both met.
2) We assume when $n \le p - 1$, $m \le q - 1$, the proposed algorithm can compute $Z_{n\times m}$, such that the conditions (12) and (13) are both met.
3) When $n = p$, $m = q$, according the algorithm, we first compare C_1^- with R_1^-, there are three cases.
   a) If $C_1^- = R_1^-$, then set $z_{11} = R_1^-$, $z_{1k} = 0$, $k = 2, 3, \ldots, m$ and $z_{k1} = 0$, $k = 2, 3, \ldots, n$. For the row 1 and column 1 where z_{ij} have been determined, we have $\sum_{j=1}^{m} z_{1j} = z_{11} = R_1^- = R'_1$ and $\sum_{i=1}^{n} z_{i1} = z_{11} = C_1^- = C'_1$. So the conditions (12) and (13) are both met in row 1 and column 1. The remaining undetermined elements z_{ij}, $i = 2, 3, \ldots, n$, $j = 2, 3, \ldots, m$, are in the matrix $Z_{(p-1)\times(q-1)}$. According to assumption 2), the remaining matrix $Z_{(p-1)\times(q-1)}$ can be correctly worked out.
   b) If $C_1^- > R_1^-$, then set $z_{11} = R_1^-$, $z_{1k} = 0$, $k = 2, 3, \ldots, m$ and $C_1^- = C_1^- - R_1^-$. For the row 1 where z_{ij} have been determined, we have $\sum_{j=1}^{m} z_{1j} = z_{11} = R_1^- = R'_1$, condition (12) is met. For the column 1 which is updated, we have $C_1^- + z_{11} = C'_1$, it does not violate condition (13). The remaining undetermined elements z_{ij}, $i = 2, 3, \ldots, n$, $j = 1, 2, \ldots, m$, are in the matrix $Z_{(p-1)\times q}$. We continue run the algorithm to compute the remaining elements in $Z_{(p-1)\times q}$ that satisfies the conditions (12) and (13). Note that C_1^- monotonously decreases after each round of assignment and $\sum_{i=2}^{n} R_i^- = \sum_{j=1}^{m} C_j^- > C_1^-$. There must exist $R_l^- \ge C_1^-$ in round l, we set $z_{l1} = C_1^-$, $z_{k1} = 0$, $k = l+1, l+2, \ldots, n$ and $R_1^- = R_l^- - C_1^-$. Then the remaining matrix is $Z_{(p-l+1)\times(q-1)}$. According to assumption 2), the remaining matrix $Z_{(p-l+1)\times(q-1)}$ can be correctly worked out.
   c) If $R_1^- > C_1^-$, similar to b), we can prove this case.
4) The proof of cases $n = p$, $m = q - 1$ and $n = p - 1$, $m = q$ are similar to 3).

Combining 1), 2), and 3) with 4), the proposed algorithm can correctly compute all elements in the matrix $Z_{n\times m}$, such that the conditions (12) and (13) are both met. Theorem proved. □

Theorem 4: The time complexity of *FillMatrix* Algorithm is $O(n^2)$.

Proof: In *FillMatrix* Algorithm, each time we compare R_i^- with C_j^- and determine the dummy elements in a row (or a column), without backtracking. Plus the initialization of R_i^- and C_j^-, all dummy elements in the matrix can be determined in $O(n^2)$ time. Theorem 4 is proved. □

Thus, the case of $n > m$ can be transformed to the case of $n = m$. We have the complete algorithm of decomposing the workload matrix for general cases of $n \geq m$.

DecomposeMatrix Algorithm

Input: a workload matrix $X_{n\times m}$.

Output: a sequence of schedule matrices P_i.

Begin

if $n > m$ **then**

Run *FillMatrix* on $X_{n\times m}$ to obtain $W_{n\times n}$;

endif

Run *DecomposeMatrix-nn* to obtain schedule matrices $P_1, P_2, \ldots, P_t$;

Output $W_{n\times n} = P_1 + P_2 + \cdots + P_t$;

End

Theorem 5: The total time complexity of the *DecomposeMatrix* algorithm is $O(|E| \times n^3)$.

Proof: According to Theorem 2 and Theorem 4, this theorem is proved. □

Given a workload matrix $X_{n\times m}$, using the proposed algorithm, we can fill the matrix to make it a square matrix $W_{n\times n}$ and decompose $W_{n\times n}$ into a sequence of schedule matrices as follows:

$$W_{n\times n} = P_1 + P_2 + \cdots + P_t. \tag{14}$$

Let P_i' denote the matrix which contains the first m columns in P_i (i.e., the information for the m valid targets by dropping the $n - m$ dummy columns), $i = 1, 2, \ldots, t$. By removing the dummy columns in P_i, we have

$$X_{n\times m} = P_1' + P_2' + \cdots + P_t'. \tag{15}$$

The above discussions conclude that a workload matrix is decomposable to a sequence of schedule matrices such that each value of x_{ij}, $1 \leq i \leq n$, $1 \leq j \leq m$, can be actually met. In Section IV-C, we will determine a sensor surveillance tree for each schedule matrix that specifies the routes for active sensors to pass sensed data to the BS, such that the maximal lifetime L can be finally achieved.

C. Determine Surveillance Tree

We have obtained a sequence of schedule matrices. Each schedule matrix specifies the active sensors watching targets for a period of time (called a session). To allow the active sensors send their sensed data to the BS at each session, we need to construct a sensor surveillance tree whose root is the BS and all leaf nodes are the active sensors. The sensed data flow from active sensors to the BS along the tree. Some active sensors can perform both duties of watching targets and forwarding data for other sensors at the same time.

From computing the LP formulation in Section IV-A, we have obtained a data flow f_{ij} from any sensor node i to sensor node j. To forward data to the BS, each sensor node, say i, needs to follow its outgoing flow f_{ij} in order to achieve the maximal lifetime L. Suppose sensor i has l downstream nodes, denoted by $s_1, s_2, \ldots, s_l$, to forward its data to the BS (i.e., $f_{i1}, f_{i2}, \ldots, f_{il}$ have non-zero values). Since there is no ordering of data flow $f_{i1}, f_{i2}, \ldots, f_{il}$, we simply let sensor i forward its outgoing data first to s_1 until flow f_{i1} is saturated, then switch to s_2 until the value of f_{i2} is met, and finally forward the last flow f_{il} to s_l. The outgoing data of sensor i include its own sensed data and the data it helps others to forward to the BS. By following the data flow obtained from the LP formulation in forwarding data to the BS, the optimal routes, in terms of energy cost, are used and thus the maximal lifetime L is achieved.

In the sensor surveillance system, after computing the schedule matrices and the data flow, the BS will disseminate this schedule and flow to sensor nodes at the system initialization stage. After the system starts operation, each sensor will watch targets, turn off to sleep, receive and forward data according to its own schedule. There is no need to coordinate with others to switch target watching at the end of each session. In fact, a sensor does not see sessions. When a sensor is required to watch the same target for several consecutive sessions, its schedule would specify this sensor to watch the target continuously until it is time to turn itself off or switch to another target. Thus, each sensor works according to its own schedule *independently* from the others.

The sensors work by their own schedule based on their local clocks. The clocks on the sensors will drift away from each other from time to time. To ensure a target will be watched by another sensor continuously before the current one switched off, clocks of the sensors need to be synchronized. There are some clock synchronization protocols [27] for sensor networks, including some localized methods [28], and they have bounded errors for clock synchronization. When scheduling sensors to watch targets, the system can add a small buffer-period (in the order of milliseconds depending on the clock error) in the front and at the end of a working session to ensure that a target will be watched continuously at sensor switching. Notice that compared with the duration of a working session the buffer-period is several orders of magnitude smaller.

V. Experiments and Simulations

A. A Numeric Example

We randomly place a BS, six sensors (uncolored in Fig. 2) and three targets (gray in Fig. 2) in a 10×10 two-dimensional free-space region. For simplicity, the surveillance range of sensors is set to 0.4×10, and the maximal transmission range of all sensors is set to 0.8×10 (our solution can work for systems with non-uniform maximal transmission ranges or surveillance ranges). Fig. 2 shows neighbors of sensors and the surveillance relationship between sensors and targets. An edge between a sensor and a target represents the target is within the surveillance range of the sensor. An arc from sensor s_i to s_j represents s_j is within the maximal transmission range of s_i (in this example, maximal transmission ranges for all sensors are uniform, so arcs are replaced by edges in Fig. 2). The initial energy reserves of sensors are random numbers generated in the range of $[0, 100]$

TABLE I
THE INITIAL ENERGY RESERVES OF SIX SENSORS

Sensors	1	2	3	4	5	6
E_i	52.6522	90.9600	76.7550	92.3830	92.7103	61.5487

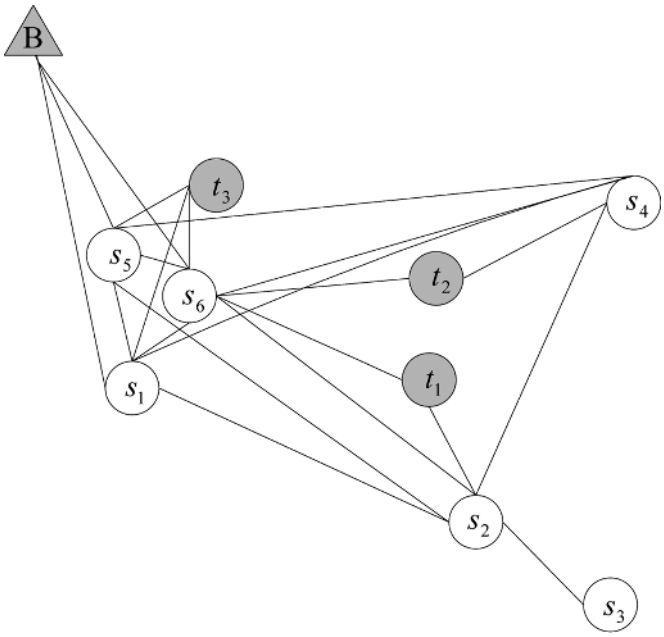


Fig. 2. An example with six sensors and three targets.

with the mean at 50, as shown in Table I. To simulate the energy consumed on different tasks, we set $e^T = 0.12, e^R = 0.1$. These values are in proportional to the actual power consumption for transmitting and receiving data, respectively, as pointed out in [29]. Experiments in [29] further showed that energy cost of sensing data, such as monitoring temperature and humidity, is comparable to the energy cost of receiving data. We set $e^S = 0.1$ and the sensing data rate $R = 1$. The signal decline factor á is set to 2.

We follow the three steps in our method to find sensor surveillance trees.

First, we use the linear programming, described in Section IV-A, to compute the maximal lifetime L, workload matrix $X_{6\times3}$ and data flows (see Table II) that achieve L:

$$L = 28.6972,$$

$$X_{6\times3} = \begin{bmatrix} 0 & 0 & 0 \\ 17.2300 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 13.0999 & 0 \\ 0 & 0 & 28.6972 \\ 11.4672 & 15.5973 & 0 \end{bmatrix}.$$

TABLE II
THE DATA FLOWS AMONG SIX SENSORS AND THE BS

Date flow	The amount of data flow
$s_1 \rightarrow$BS	13.97655
$s_2 \rightarrow s_1$	13.97655
$s_2 \rightarrow s_5$	3.25345
$s_4 \rightarrow s_5$	13.0999
$s_5 \rightarrow$BS	54.674638
$s_6 \rightarrow s_5$	9.624088
$s_6 \rightarrow$BS	17.4404412

In the workload matrix, we can see target 1 is watched by sensors 2 and 6 for 17.2300, 11.4672, respectively. The total time for target 1 to be watched is 28.6972, which is the lifetime of the surveillance system.

Second, we run the *FillMatrix* algorithm to append a dummy matrix to the workload matrix to make it a square matrix $W_{6\times a6}$, where the sum of each column and the sum of each row are all equal to L, as shown in the equation at the bottom of the page.

Third, we run the *DecomposeMatrix-nn* algorithm to decompose $W_{6\times6}$ into three schedule matrices P_1, P_2, and P_3 (i.e., the decomposition terminates at round 3), such that

$$W_{6\times6} = P_1 + P_2 + P_3.$$

By removing the dummy columns of the schedule matrices, we have

$$X_{6\times3} = \begin{bmatrix} 0 & 0 & 0 \\ 1.6327 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1.6327 & 0 \\ 0 & 0 & 1.6327 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 15.5973 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 15.5973 \\ 0 & 15.5973 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 11.4672 & 0 \\ 0 & 0 & 11.4672 \\ 11.4672 & 0 & 0 \end{bmatrix}.$$

$$W_{6\times6} = \begin{bmatrix} 0 & 0 & 0 & 28.6972 & 0 & 0 \\ 17.2300 & 0 & 0 & 0 & 11.4672 & 0 \\ 0 & 0 & 0 & 0 & 17.2300 & 11.4672 \\ 0 & 13.0999 & 0 & 0 & 0 & 15.5973 \\ 0 & 0 & 28.6972 & 0 & 0 & 0 \\ 11.4672 & 15.5973 & 0 & 0 & 0 & 1.6327 \end{bmatrix}.$$

Finally, the surveillance trees based on the above schedule matrices and data flows are determined. The surveillance trees for three sessions are shown in Fig. 3(a)–(c).

It is easy to see that the surveillance trees in Fig. 3(a)–(c) satisfy both the surveillance requirement and data flow constraints. The maximal lifetime L is actually achieved.

B. Simulations

We conduct simulations to study the complexity of our proposed solution and compare its performance with a greedy method.

The simulations are conducted in a 100×100 two-dimensional free-space region. BS, sensors and targets are randomly distributed inside the region. Again, the surveillance range and the maximal transmission range of all sensors are set to 0.4×100 and 0.8×100, respectively (except the simulations for Fig. 5(a) and (b). The signal decline factor $\alpha = 2$ and the initial energy reserves of sensors are the random numbers in the range of $[0, 100]$, with the mean value of 50. The linear programming formulated in Section IV-A is computed by using MatLab 6.5. The results presented in the figures are the means of 100 separate runs.

1) Linear Growth of Decomposition Steps: According to Theorem 2, we know the number of steps for decomposing the workload matrix, t, is bounded by $t \leq |E|$. In the simulations, we found that t is linear to the size of the system.

Fig. 4(a) and (b) shows the increase of t versus the change of N (number of sensors) and M (number of targets), respectively, when one of the two variables is fixed. From the figures, we can see a strong linear relationship between t and N (or M). This result tells us that the actual number of steps for decomposing the matrix is linear to the size of system in real runs.

2) Comparison With a Greedy Method: A greedy algorithm is proposed to compare the performance with our optimal solution. The basic idea of the greedy method is as follows. Each time, we find a sensor to watch each target. We use the maximum matching algorithm in the sensor-target bipartite graph to find the pairs of sensor and target. Then, for each sensor scheduled to work in this session we find the minimal energy cost path from it to the BS in the sensor network graph (the sensor interconnectivity graph). When any node that is either in watching a target or in the path runs out of energy, it is removed from the bipartite graph and the network graph, and another maximum matching and routes are computed. This operation is repeated until no maximum matching can be found to cover all targets or there no longer exists a path from a sensor to BS cannot be found. The system lifetime of the greedy method is the total service time.

We set $N = 100$ and $M = 10$. Fig. 5(a) and (b) shows the lifetime versus the change of surveillance range and the maximal transmission range of sensors, respectively. From the figures, we can see that when the surveillance range (the maximal transmission range) becomes larger, the performance gap becomes more significant. This is because with a small surveillance range (maximal transmission range), sensors have only got a few targets (sensors) within its surveillance range (maximal transmission range). There is little room for optimization.

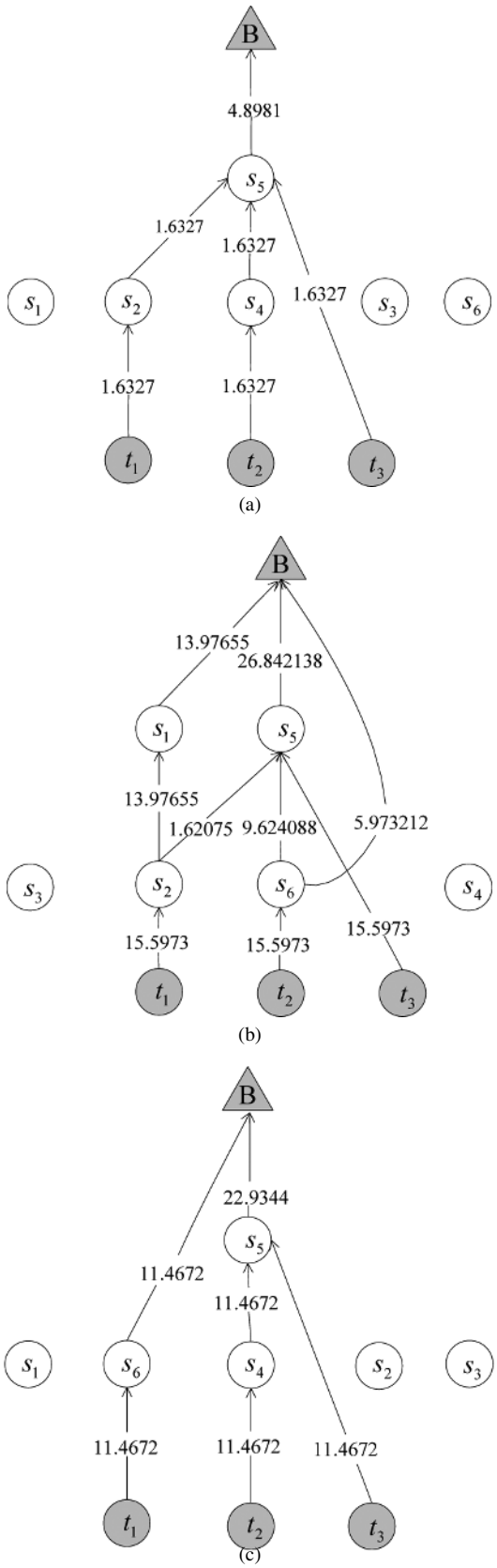
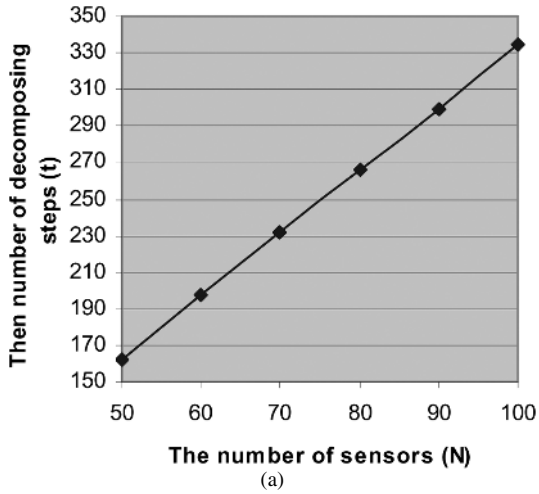
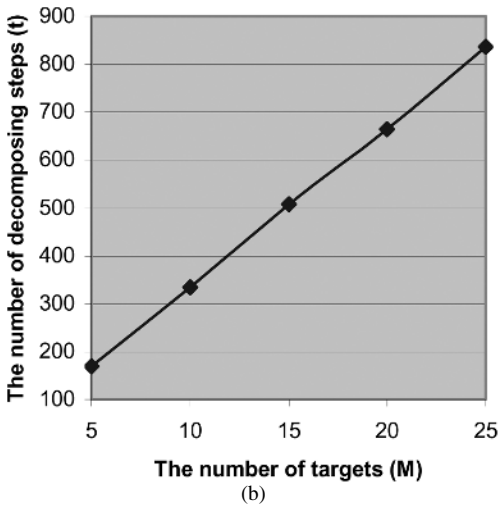


Fig. 3. (a) The surveillance tree of session 1. (b) The surveillance tree of session 2. (c) The surveillance tree of session 3.

As the surveillance range (the maximal transmission range) becomes larger, more sensors are able to cover multiple targets (sensors), which gives our method more room to schedule the



(a)

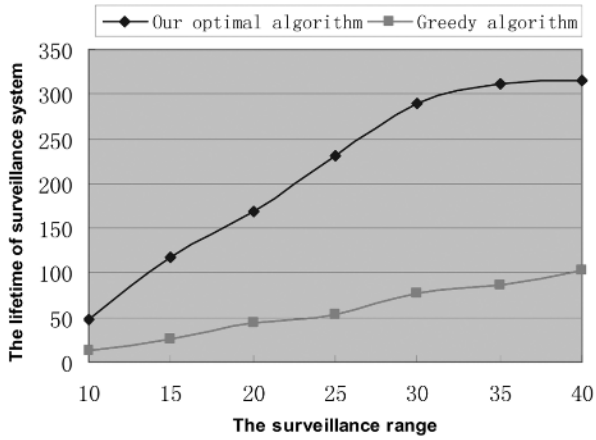


(b)

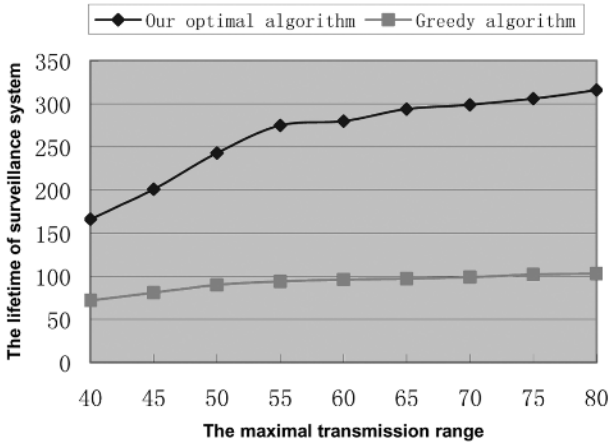
Fig. 4. (a) t versus N when $M = 10$. (b) t versus M when $N = 100$.

sensors properly to achieve the maximal lifetime. That is why the performance gap between the two methods becomes more significant as the increase of the surveillance range (the maximal transmission range). Furthermore, we can see that the increase of surveillance range is more effective to extending the system lifetime than the increase of the maximal transmission range. This is because the surveillance range is usually much smaller than the communication range of sensors. It is always the bottleneck of the maximization of system lifetime, and some targets could not be watched by enough sensors often results in quick die of surveillance systems.

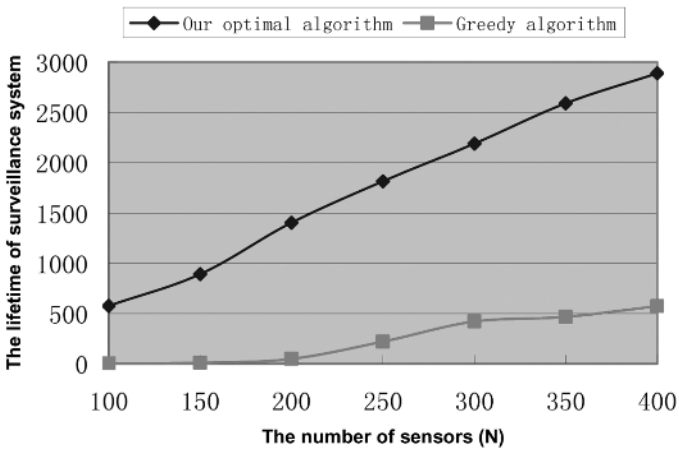
Fig. 5(c) shows the lifetime versus the number of sensors placed in the same region. The number of sensors varies from 100 to 400 and the number of targets is fixed at 50. This simulation shows how the lifetime is affected by the density of sensors. Fig. 5(c) exhibits the similar trend as in Fig. 5(a) and (b). As more sensors deployed in the same region, the density becomes higher. A target can be watched by more sensors and there is a higher chance for a target to be in the watching range of multiple sensors. At the same time, a sensor can reach more neighbors and can choose more energy-efficient routes to forward data. Thus, our optimal algorithm takes more advantages by optimizing the schedule and the performance gain becomes more significant than the greedy method when the density of sensors becomes higher.



(a)



(b)



(c)

Fig. 5. (a) Lifetime versus surveillance range. (b) Lifetime versus the maximal transmission range. (c) Lifetime versus N when $M = 50$.

From Figs. 4(a)–5(c), we can make the following conclusions:

1) The actual number of steps for decomposing the workload matrix is linear to the size of system in real runs.
2) Our optimal algorithm has significantly better performance in the situation where sensors have larger surveillance and communication range, or when sensors are densely deployed.

3) The increase of surveillance range is more effective to extending the system lifetime than the increase of the maximal transmission range of sensors.

VI. Conclusion

We have presented the maximal lifetime scheduling problem in sensor surveillance systems. This is the first time in the literature that the problem of maximizing lifetime of sensor surveillance systems was formulated and the optimal solution was obtained. Simulations have been conducted to show the superior performance of our method in comparison with a greedy scheduling method under various network scenarios.

There is some related work in the literature about scheduling of sensors in surveillance systems. A notable work in [30] discusses the problem of selecting a minimum number of connected sensors to cover a given set of interested points (targets). However, operating with the minimal number of sensors does not simply imply the maximal lifetime of the system. It is because that without global optimization, some nodes that can cover many targets could be scheduled to work heavily and they will quickly run out of energy. Another work is a sensing protocol proposed in [2]. It discusses a sensor coverage problem, which is to schedule a set of sensors to work and sleep in turn, such that a given area can be fully covered by working sensors at any time and the lifetime of the system is maximized. The method used to cover an area is to divide the area into grid, and it is defined that the entire area is covered if all the grid points are covered. A sensor can cover multiple grid points simultaneously if all the grid points are within the sensing range of this sensor. The schedule algorithm is centered with grid points. That is, for each grid point, its watching time is split among the sensors that are able to watch it. This is a localized scheduling method. But, it is not an optimal method and there is no performance guarantee of this method. Moreover, none of the work in [2] and [30] considered the communication cost of sending data from sensors to base stations. According to our experience, a greedy schedule algorithm [31] (similar to the scheduling method in [2]) could perform very badly when taking communication cost into account.

Acknowledgment

The authors would like to thank Prof. Dingzhu Du and Prof. Xiaotie Deng for pointing them towards relevant results on decomposing of doubly stochastic matrices.

References

[1] C.-Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proc. IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.

[2] T. Yan, T. He, and J. A. Stankovic, "Differentiated surveillance for sensor networks," in *Proc. 1st Int. Conf. Embedded Networked Sensor Systems*, Los Angeles, CA, 2003, pp. 51–62.

[3] C.-F. Hsin and M. Liu, "A distributed monitoring mechanism for wireless sensor networks," in *Proc. Int. Conf. Mobile Computing and Networking, ACM Workshop on Wireless Security*, Atlanta, GA, 2002, pp. 57–66.

[4] Y. Zhao, R. Govindan, and D. Estrin, "Residual energy scans for monitoring wireless sensor networks," in *Proc. IEEE Wireless Communications and Networking Conf.*, 2002, pp. 356–362.

[5] S. Mao and Y. T. Hou, "BeamStar: A new low-cost data routing protocol for wireless sensor networks," presented at the IEEE Globecom, Dallas, TX, 2004.

[6] W. Choi and S. K. Das, "Trade-off between coverage and data reporting latency for energy-conserving data gathering in wireless sensor networks," presented at the 1st IEEE Int. Conf. Mobile Ad Hoc and Sensor Systems (MASS 2004), Fort Lauderdale, FL, Oct. 2004.

[7] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," presented at the 6th Annu. ACM/IEEE Int. Conf. Mobile Computing and Networking (MOBICOM), Boston, MA, Aug. 2000.

[8] W. Heinzelman, A. Chandrakasan, and H. Balakrishna, "Energy-efficient communication protocol for wireless microsensor networks," presented at the 33rd Annu. Hawaii Int. Conf. System Sciences (HICSS-33), Maui, HI, Jan. 2000.

[9] S. Lindsey, C. Raghavendra, and K. M. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 9, pp. 924–935, Sep. 2002.

[10] N. Sadagopan and B. Krishnamachari, "ACQUIRE: The acquire mechanism for efficient querying in sensor networks," in *Proc. 1st IEEE Int. Workshop on Sensor Network Protocols and Application (SNPA)*, 2003, pp. 149–155.

[11] W. R. Heinzelman, J. Kulit, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," presented at the 5th ACM/IEEE Annu. Int. Conf. Mobile Computing and Networking (MOBICOM), Seattle, WA, Aug. 1999.

[12] M. Bhardwaj, T. Garnett, and A. Chandrakasan, "Upper bounds on the lifetime of sensor networks," in *IEEE Int. Conf. Communications*, 2001, pp. 785–790.

[13] J. Chang and L. Tassiulas, "Maximum lifetime routing in wireless sensor networks," presented at the Advanced Telecommunications and Information Distribution Research Program (ATIRP'2000), College Park, MD, Mar. 2000.

[14] T. Melodia, D. Pompili, and I. F. Akyildiz, "Optimal local topology knowledge for energy efficient geographical routing in sensor networks," in *Proc. IEEE INFOCOM*, 2004, pp. 1705–1716.

[15] N. Sadagopan and B. Krishnamachari, "Maximizing data extraction in energy-limited sensor networks," in *Proc. IEEE INFOCOM*, 2004, pp. 1717–1727.

[16] G. Zussman and A. Segall, "Energy efficient routing in ad hoc disaster recovery networks," in *Proc. IEEE INFOCOM*, 2003, pp. 682–691.

[17] J. Carle and D. Simplot-Ryl, "Energy-efficient area monitoring for sensor networks," *IEEE Computer*, vol. 37, no. 2, pp. 40–46, Feb. 2004.

[18] C. F. Chiasserini and M. Garetto, "Modeling the performance of wireless sensor networks," in *Proc. IEEE INFOCOM*, 2004, pp. 220–231.

[19] D. Tian and N. D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *Proc. 1st ACM Int. Workshop on Wireless Sensor Networks and Applications*, 2002, pp. 32–41.

[20] L. B. Ruiz *et al.*, "Scheduling nodes in wireless sensor networks: A Voronoi approach," in *Proc. 28th IEEE Conf. Local Computer Networks (LCNS2003)*, Bonn/Konigswinter, Germany, Oct. 2003, pp. 423–429.

[21] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. 1st ACM Int. Workshop on Wireless Sensor Networks and Applications*, Atlanta, Ga, Sep. 2002, pp. 88–97.

[22] H. J. Ryser, *Combinational Mathematics*. Washington, DC: The Mathematical Association of America, 1963, pp. 58–59.

[23] R. A. Brualdi and H. J. Ryser, *Combinatorial Matrix Theory*. Cambridge, U.K.: Cambridge Univ. Press, 1991, pp. 9–10.

[24] D. B. West, *Introduction to Graph Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1996, pp. 109–111.

[25] S. Axler, F. W. Gehring, and K. A. Ribet, *Graph Theory*, 2nd ed. New York: Springer, 2000.

[26] R. Gould, *Graph Theory*. Boston, MA: Benjamin/Cummings, 1988, pp. 198–209.

[27] K. Römer, "Time synchronization in ad hoc networks," presented at the ACM Mobihoc, Long Beach, CA, 2001.

[28] Q. Li and D. Rus, "Global clock synchronization in sensor networks," in *Proc. IEEE INFOCOM*, 2004, pp. 214–226.

[29] A. Savvides, C.-C. Han, and M. Srivastava, "Dynamic fine-grained localization in ad hoc networks of sensors," in *Proc. MobiCom 2001*, Rome, Italy, 2001, pp. 166–179.

[30] Z. Zhou, S. R. Das, and H. Gupta, "Connected K-coverage problem in sensor networks," in *Proc. ICCCN 2004*, pp. 373–378.

[31] H. Liu, P. Wan, C.-W. Yi, X. Jia, S. Makki, and P. Niki, "Maximal lifetime scheduling in sensor surveillance networks," in *Proc. IEEE INFOCOM*, Miami, FL, 2005, pp. 2482–2491.

Hai Liu received the B.Sc. and M.Sc. degrees in applied mathematics from the South China University of Technology in 1999 and 2002, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong in 2006.

He is currently a Research Fellow in the Department of Computer Science, City University of Hong Kong. His research interests include distributed systems, wireless networks, and mobile computing.

Xiaohua Jia received the B.S. and M.Eng. degrees from the University of Science and Technology of China, Hefei, China, in 1984 and 1987, respectively, and the D.Sc. degree in information science from the University of Tokyo, Tokyo, Japan, in 1991.

He is currently a Professor of computer science at the City University of Hong Kong and Cheung Kong Professor with the School of Computing, Wuhan University, China.

Prof. Jia is an editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, *The Journal of Supercomputing*, *Journal of the World Wide Web*, and *Journal of Combinatorial Optimization*, among others. He has been a general chair, PC chair, PC member, and OC member of many international conferences.

Peng-Jun Wan received the B.S. degree from Tsinghua University, China, in 1990, the M.S. degree from the Chinese Academy of Science, Beijing, in 1993, and the Ph.D. degree from the University of Minnesota, Minneapolis, in 1997.

He is currently an Associate Professor of computer science at the Illinois Institute of Technology, Chicago. His research interests include wireless networks and optical networks.

Chih-Wei Yi received the B.S. and M.S. degrees from National Taiwan University, Taipei, Taiwan, R.O.C., and the Ph.D. degree from the Illinois Institute of Technology, Chicago.

He is currently an Assistant Professor of computer science at National Chiao Tung University, Hsinchu, Taiwan, R.O.C. His research focuses on wireless ad hoc and sensor networks.

S. Kami Makki received the Bachelors and Masters degrees in engineering from the University of Tehran, Iran, the M.S. degree in computer science and engineering from the University of New South Wales, Australia, and the Ph.D. degree in computer science from the University of Queensland, Australia.

Before joining the University of Toledo, Toledo, OH, he held a number of academic positions and research appointments, and also worked in public and private industries for a number of years. Currently, he is the Director of the Advanced Systems Lab, and Assistant Professor in the Department of Electrical Engineering and Computer Science at the University of Toledo. His research areas include distributed systems, web and multimedia databases, intelligent web applications, middleware integration and electronic services, and mobile and wireless network and security.

Niki Pissinou received the B.S.I.S.E. degree in industrial and systems engineering from The Ohio State University, Columbus, the M.Sc. degree in computer science from the University of California at Riverside, and the Ph.D. degree in computer science from the University of Southern California, Los Angeles.

She is currently a tenured Professor and the Director of the Telecommunication and Information Technology Institute at Florida International University, Miami. She is active in the fields computer networks, information technology, and distributed systems. She has published over 100 refereed publications and has received best paper awards. She has also co-edited eight volumes and is the author of an upcoming book on wireless internet computing.

Dr. Pissinou has served as a steering committee and general chair and program committee member of over 100 program and organizational committees of IEEE and ACM sponsored technical conferences. She has been the editor of eight journals including IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING and guest editor of seven journals. She has received eight achievements awards and been an invited speaker and keynote speaker of conferences. She has received extensive funding for her research work from such agencies as NSF, NASA, DARPA, and ARO, including two recent NSF awards on wireless networks.