Energy-Aware Set-Covering Approaches for Approximate Data Collection in Wireless Sensor Networks

Chih-Chieh Hung, Wen-Chih Peng, *Member*, *IEEE*, and Wang-Chien Lee, *Member*, *IEEE*

Abstract—To conserve energy, sensor nodes with similar readings can be grouped such that readings from only the representative nodes within the groups need to be reported. However, efficiently identifying sensor groups and their representative nodes is a very challenging task. In this paper, we propose a centralized algorithm to determine a set of representative nodes with high energy levels and wide data coverage ranges. Here, the data coverage range of a sensor node is considered to be the set of sensor nodes that have reading behaviors very close to the particular sensor node. To further reduce the extra cost incurred in messages for selection of representative nodes, a distributed algorithm is developed. Furthermore, maintenance mechanisms are proposed to dynamically select alternative representative nodes when the original representative nodes run low on energy, or cannot capture spatial correlation within their respective data coverage ranges. Using experimental studies on both synthesis and real data sets, our proposed algorithms are shown to effectively and efficiently provide approximate data collection while prolonging the network lifetime.

Index Terms—Approximate data collection, wireless sensor networks, spatial correlation and clustering

1 Introduction

Recent advances in wireless and embedded technologies have increased the deployment of small and inexpensive wireless sensor nodes in various applications, including field data collection, remote monitoring and control, smart homes, factory automation, and security. In a wireless sensor network, sensor nodes that are capable of collecting, processing, and storing environmental information are deployed in a monitored region. An access point (i.e., a sink) serves as a network interface, issuing queries and collecting readings from the nodes. Each node sends its readings to the sink via multihop communication. Since sensor nodes are usually powered by batteries, energy saving is a vital design issue in wireless sensor networks.

To address this issue, a considerable amount of research has proposed energy saving mechanisms. Without loss of generality, these mechanisms are categorized into two types: the network perspective type and the data perspective type. These two types of energy-saving mechanisms are targeted on different layers and aspects. The network (respectively, data) perspective type deals with the network (respectively, data) layer of wireless sensor networks. Since different layers have different characteristics, most of prior works usually target at developing energy-saving mechanisms on either the network or the data layer. In the network perspective type, research efforts are targeted at designing energy efficient network protocols [32], [3], [17], [14] for scheduling the activity and sleep modes of the sensors. With a proper schedule, one can reduce energy consumption by preventing the waste of listening packets and of transmission collision. For the data perspective type, most previous work has focused on designing energy efficient query processing and data collection approaches with the purpose of minimizing the number of messages transmitted. Usually, these works assume that the MAC layer protocol can be adopted by existing energy efficient network protocols (e.g., S-MAC [31], [32]). For example, the authors in [24] focused on the design of an algorithm for adaptive aggregation data collection, and adopted S-MAC as their MAC layer protocol. Previous works have proposed in-network aggregate query processing in which sensor nodes are able to perform aggregation to reduce the number of messages, thereby conserving the energy of the sensor nodes [22], [11], [27], [10]. On the other hand, for some applications, one common query is to collect all sensing readings from sensors. Previous research works have elaborated on *approximate data collection* approaches to reduce the energy consumption. Without loss of generality, given a sensor network with n sensor nodes, approximate data collection is usually performed by selecting m representative nodes to report their readings while the remaining $n - m$ sensor nodes do not need to report their sensor reading to the sink [13], [16], [29]. Using the readings from these m representative nodes (referred to as R-nodes), the readings of all the nodes can be estimated. This approach is promising since sensor nodes in the vicinity are usually expected to have similar readings (i.e., they are spatially correlated) [5], [23], [24]. Since representative nodes are

- *C.-C. Hung is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. E-mail: hungcc@cs.nctu.edu.tw.*
- *W.-C. Peng is with the Department of Computer Science, National Chiao Tung University, EC542, 1001 Ta Hsueh Road, Hsinchu 300, Taiwan, R.O.C. E-mail: wcpeng@cs.nctu.edu.tw.*
- *W.-C. Lee is with the Department of Computer Science and Engineering, The Pennsylvania State University, 360D Information Sciences and Technology Building, University Park, PA 16802. E-mail: wlee@cse.psu.edu.*

Manuscript received 12 Aug. 2010; revised 19 Aug. 2011; accepted 15 Sept. 2011; published online 19 Oct. 2011.
Recommended for acceptance by B.C. Ooi.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2010-08-0445.
Digital Object Identifier no. 10.1109/TKDE.2011.224.

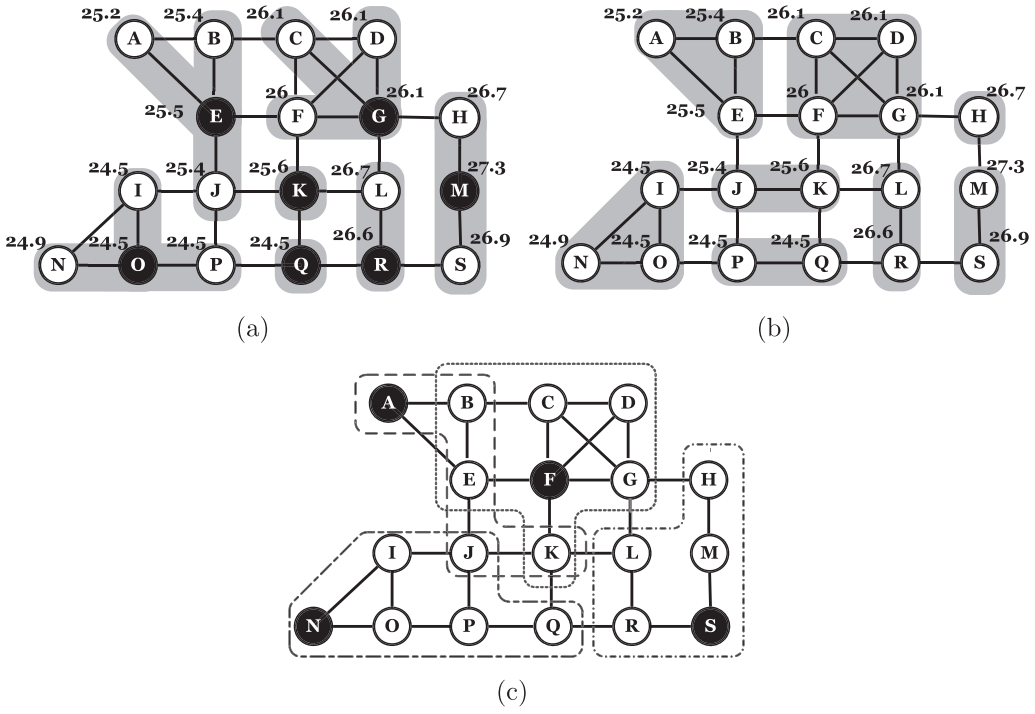


Fig. 1. Illustrative examples of communication connectivity graph: (a) snapshot, (b) clique-covering, and (c) set-covering.

responsible for reporting their readings, the number of representative nodes m and their energy usage has a great impact on the energy consumption of the sensors. Our main goal in this paper is to design energy efficient approaches for approximate data collection, and we adopt existing energy efficient MAC protocols (i.e., S-MAC) for our network protocol in wireless sensor networks.

In general, approximate data collection in wireless sensor networks allows users to specify an error bound on readings of the sensor nodes. Some research efforts have elaborated on exploiting spatial correlation for approximate data collection. Explicitly, clustering techniques can be used to capture similar readings due to spatial correlation between sensor nodes. A distance function in the data domain of sensor readings can be used to model the similarity between the readings of two sensor nodes, whereby the smaller the distance between two readings, the more similar they are. Meanwhile, sensor nodes located spatially close to each other can be identified by their communication connectivity. For our purposes, we denote the distance to refer to the similarity in the data domain of sensor readings, whereas the physical distance between the sensors refers to the communication connectivity. Given a specified distance threshold, nearby sensor nodes with similar readings can be grouped. In [13], Kotidis proposes a snapshot query in which only some selected representative nodes (i.e., R-nodes) would report their readings, and the sensor readings of nearby one-hop nodes are approximated as readings of the corresponding R-node. However, only one-hop neighbors are involved in the similarity calculation, and the spatial correlation between sensor nodes is not fully exploited. In [16], Liu et al. propose clustering nearby sensor nodes that have similar sensing readings, and formulate the problem as a clique-covering problem. In this clique-covering problem, a clique in the communication graph refers to a group of sensor nodes that have strong spatial correlation.[1] Sensor nodes in the same clique can not only communicate directly with each other but also have similar readings. Thus, it is sufficient to select one R-node to represent all sensor nodes in the same clique. For example, Fig. 1 shows a communication connectivity graph, where each vertex is a sensor associated with its reading, and each edge between a pair of vertices represents that these two sensors can directly communicate with each other. Assume that the Manhattan distance (i.e., the absolute difference value in the two sensing readings) is used as the similarity function and that the error threshold is 0.5. As shown in Fig. 1a, the number of R-nodes under snapshot queries is 7. As can be observed in Fig. 1b, there are eight disjoint cliques covering the whole set of nodes in the graph. Note that the clique-covering problem is better than snapshot queries in terms of the number of R-nodes.

Prior works have shown that only R-nodes reporting their sensing readings are able to prolong the network lifetime of wireless sensor networks, where the network lifetime is measured as time duration until the first sensor in the network exhausts its battery [13], [16]. To conserve energy, as few R-nodes as possible should be selected such that the union of their particular data coverage ranges covers all the sensor nodes. From this point of view, we first propose the concept of data coverage ranges to fully capture spatial correlations among sensors. The data coverage range of a sensor node s_i is the set of sensor nodes that are connected via a sequence of one-hop neighbors with readings similar to that of s_i (governed by a threshold value ϵ). Each sensor node within the data coverage range of a sensor node s_i is said to be *data-covered* by s_i.[2] Consider the example in Fig. 1, where the

1. Given a graph $G = (V, E)$, find the minimum number of disjoint cliques $\{C_1, C_2, \ldots, C_k\}$ such that $\bigcup_{i=1}^{k} C_i = V$.

2. We use the term "cover" in short to represent the term "data-cover" in the rest of this paper.

Manhattan distance is used as the similarity function and the error threshold is set to 0.5. Since the readings of sensor nodes A, B, E, J, and K are 25.2, 25.4, 25.5, 25.4, and 25.6, respectively, we can derive the data coverage range of A as {A,B,E,J,K}. Clearly, the data coverage ranges of A, F, N, and S are {A,B,E,J,K}, {B,C,D,E,F,G,K}, {I,J,N,O,P,Q}, and {H,L,M,R,S}, respectively. Given a set of sensors with their data coverage ranges, one can select fewer R-nodes for the purposes of approximate data collection. For example, in Fig. 1c, A, F, N, and S are selected as R-nodes and their data coverage ranges cover all sensor nodes. However, selecting the minimal number of R-nodes may not extend the network lifetime of wireless sensor networks since R-nodes may quickly deplete their energy.

Since the goal of our paper is to extend the network lifetime, besides reducing the number of R-nodes, we further claim that the selection of R-nodes should take not only data coverage ranges but also the remaining energy of the sensors into consideration. If R-nodes with lower energy are selected, the network lifetime will still be shortened. However, there is a possibility that selecting only sensor nodes with higher energy levels may increase the number of R-nodes because R-nodes with more energy may have smaller data coverage ranges. Consequently, we formulate the problem of selecting R-nodes as an *energy-aware set-covering problem*, which is an extension of the well-known set-covering problem. In the energy-aware set-covering problem, the R-nodes that are selected should have high energy and wider data coverage ranges, and the union of their data coverage ranges should be the set of all sensor nodes. Clearly, three challenging issues need to be addressed: the collection of data coverage ranges and energy of sensors, the selection of R-nodes, and the maintenance mechanism. In this paper, a centralized algorithm *DCglobal* (standing for Data Coverage with global information) is proposed. By collecting the sensing readings and the energy of the sensors, DCglobal is able to determine a set of R-nodes with high availability of energy and wide data coverage ranges to cover the whole network. Moreover, for a large-scale network, since collecting readings and energy information from all sensor networks incurs a considerable number of message transmissions for the selection of the R-nodes, we therefore develop a distributed algorithm Data Coverage with local information *(DClocal)*. Due to its distributed nature, the procedure of selecting R-nodes is viewed as a state transition diagram in each sensor node. By exchanging information with neighboring nodes, each sensor node can decide whether it should be an R-node or not. Furthermore, when the selected R-nodes run low on energy or can no longer capture the correlations within their data coverage ranges, we further propose maintenance mechanisms for DCglobal and DClocal to dynamically select new R-nodes. Experiments based on both synthesis and real data sets show that the proposed algorithms, DCglobal and DClocal, significantly outperform previous works in terms of prolonging the network lifetime.

The rest of this paper is organized as follows. Section 2 presents some related works. Section 3 gives preliminary information. Section 4 develops the centralized algorithm, DCglobal, to select R-nodes for wireless sensor networks. Section 5 develops the distributed algorithm, DClocal. Section 6 presents performance results. Finally, Section 7 draws conclusions.

2 Related Works

Energy saving in wireless sensor networks has attracted a significant amount of research. The works presented in this section discuss data-perspective energy saving mechanisms that can be further classified into two categories. In the first category, in-network data aggregation is used for data collection [19], [30]. This uses a routing tree to partially aggregate measurements (e.g., MAX) on the way to their destination such that the amount of data transmitted is reduced. In [23], Tang and Xu employ a data collection method in which some aggregate functions are performed, while the accuracy of the data is guaranteed. In [24], Tang and Xu propose an algorithm for adaptive aggregation data collection to allocate the number of updates sent by the sensor nodes in an aggregation tree under energy constraints. These aforementioned works focus on in-network data aggregation techniques to conserve the energy of wireless sensor networks.

The second category is approximate data collection. Approximate data collection can be further divided into two subcategories. The first subcategory is approximate data collection which includes building probabilistic models of sensing readings collected from wireless sensor networks [8], [5]. In [8], Deshpande et al. explore a model-driven architecture, and a centralized probabilistic model is used to estimate the readings of the sensor nodes. Furthermore, Chu et al. [5] employ spatial correlation for approximate data collection, where a replicated dynamic probabilistic model is built to predict sensor readings. If the readings are accurately predicted, sensor nodes will not send their readings to the sink, thereby reducing communication costs. In these works, probabilistic models need to be built and carefully maintained. It is not easy to build appropriate probability models that fully capture sensed readings because the reading distribution may vary. The second subcategory is approximate data collection without building probabilistic models. Kotidis [13] derives an extension of a declarative query, termed a snapshot query, for wireless sensor networks. Snapshot queries can be answered via a data-driven approach using a linear regression model to predict readings of one-hop neighbors. Gupta et al. [9] formulate data gathering into a *connected correlation-dominating set problem* to select R-nodes. These R-nodes should form a connected subgraph in order to relay sensed data. Thus, the number of selected nodes should be sufficiently large to form a connected correlation-dominating set. Liu et al. [16] propose a centralized algorithm, named EEDC, which partitions sensor nodes based on spatial correlation into disjoint cliques such that the sensor nodes in the same clique have similar readings. Furthermore, a round-robin schedule is employed to share the workload of the data collection in each clique.

In this paper, we exploit spatial correlation for approximate data collection without any aggregation operators or probabilistic models. In particular, we formulate an energy-aware set-covering problem in which R-nodes with

comparatively higher energy and wider data coverage ranges are selected for approximate data collection. To the best of our knowledge, this is the first work to model the approximate data collection problem as an energy-aware set-covering problem. Moreover, two kinds of algorithms are proposed: a centralized algorithm DCglobal, and a distributed algorithm, DClocal. To prolong the network lifetime of wireless sensor nodes, if the current R-nodes run low their energy, R-nodes should be adjusted. With this in mind, we propose maintenance mechanisms for DCglobal and DClocal. These key features distinguish our work from existing works in this area.

3 Preliminaries

Sensor node readings are usually represented as a time series, which can be formally defined as follows:

Definition 1 (Reading Vectors). *Given a sliding window size ξ, the reading vector of a sensor node s_i is defined as $v_i(t) = <x_i(t-\xi+1), x_i(t-\xi+2), \ldots, x_i(t)>$, where $x_i(t)$ is the reading sensed by s_i at the time t.*

Considering reading vectors of sensor nodes, we can thus define the similarity (i.e., the distance function in the data domain) between sensors. The similarity between two sensor nodes (e.g., s_i and s_j) is represented as a distance function $d(v_i(t), v_j(t))$, where $v_i(t)$ and $v_j(t)$ are the reading vectors of sensors s_i and s_j at time t. It should be noted that existing distance functions are developed, and selecting an appropriate distance function usually depends on applications and tasks [12]. Thus, the distance functions used in our examples and experiments are common distance functions used in most research works. To facilitate the presentation of our paper, we employ the *Manhattan distance* (i.e., $d(s_i, s_j) = |v_i(t) - v_j(t)|$) in our examples.

Given a distance function between two sensor nodes, the data coverage range of a sensor is defined as follows.

Definition 2 (Data Coverage Range). *Given an error threshold ϵ, the data coverage range of sensor node s_i, denoted as C_i, is the set of sensor nodes such that sensor node s_j is in C_i if and only if there exists a sequence of sensor nodes $<s_i = s_0, s_1, \ldots, s_k = s_j>$ for $k \geq 0$ such that s_{t-1} directly communicates with s_t and $d(s_i, s_t) \leq \epsilon$ for $1 \leq t \leq k$.*

Consider an example in Fig. 1b, where ϵ is set to 0.5. Note that E is in sensor node A's data coverage range because there exists a sequence of sensor nodes $<A, B, E>$ such that A can communicate with B and B can communicate with E. In addition, $d(A, B) = 0.3 \leq 0.5$, and $d(A, E) = 0.3 \leq 0.5$. Consequently, the data coverage range of sensor node A is $\{A, B, E, J, K\}$.

Generally speaking, approximate data collection is performed periodically every f_c time units. Such approximate data collection can be viewed as a query "$SELECT\ *\ WITHIN \pm \epsilon\ EPOCH\ f_c$ submitted to the sink. The value of f_c is called the data collection interval. This parameter is user-specified and application-dependent. The effect of parameter f_c will be evaluated in our experiments. As mentioned earlier, since wireless sensor networks are energy constrained, our objective should be to prolong the network lifetime. As in other works [16], [23], [24], the network lifetime of wireless sensor networks is the length of time until the first sensor's battery is depleted. To prolong the network lifetime, some R-nodes are selected to report their readings, and these readings are approximated as other sensing readings. Since only a portion of sensor nodes report their readings, the network lifetime can therefore be extended. Approximate data collection is usually a long-running process, and R-nodes should be dynamically adjusted. In this paper, our proposed algorithms are executed every f_p time units and thus, for every f_p time units, a new set of R-nodes is determined. Additionally, between two consecutive executions of our proposed algorithms, we propose maintenance mechanisms to fine-tune the R-nodes. These maintenance operations will be performed every f_m time units. The execution schedule can be best understood in Fig. 2, where **C** represents the time that the R-nodes should report their readings, **P** is the time that our proposed algorithms are performed for R-node selection, and **m** represents the period for performing maintenance mechanisms. In the later sections, both the R-node selection algorithms and the corresponding maintenance mechanisms are presented.

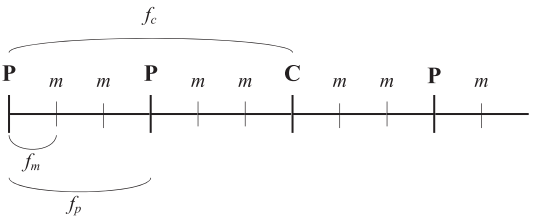


Fig. 2. An execution schedule of our proposed algorithms.

The main goal of this paper is to propose algorithms to select R-nodes from sensor nodes. As noted earlier, the R-node selection should take into consideration both energy levels and the size of the coverage ranges of all sensor nodes. We could formulate an energy-aware set-covering problem for R-node selection as follows:

Energy-aware set-covering problem. Given a set of sensor nodes, denoted as $S = \{s_1, s_2, \ldots, s_n\}$ with their data coverage ranges, expressed by $C = \{C_1(t), C_2(t), \ldots, C_n(t)\}$ and energy levels $E = \{E_1(t), E_2(t) \ldots, E_n(t)\}$[3] at time unit t, find the sets of R-nodes $R(t)$ at each time t to maximize the network lifetime (i.e., $\max\{t|\text{there exists the first sensor node } s_i \text{ with } E_i(t) = 0\}$), where $\bigcup_{s_i \in R(t)} C_i(t) = S$.

The set-covering problem is known as an NP-hard problem [7]. By mapping the set-covering problem to the energy-set covering problem, the energy-set covering problem is proven as NP-hard problem. Suppose that the instance of the set-covering problem is (S, C), where S is a set of sensor nodes and C is the set of data coverage ranges of sensor nodes. Explicitly, given the instance of the set-covering problem (S, C), this instance of the set-cover problem is reduced to the one of the energy-aware set-covering problem, where sensor nodes are initially deployed (i.e., $t = 1$) with the same energy level. Due to NP-hard of the energy-set covering problem, we propose two heuristic algorithms DCglobal and DClocal to solve this problem.

3. In this paper, we discretize the remaining energy into levels.

4 DCglobal: Determining R-Nodes with Global Information

In Section 4.1, we propose a centralized algorithm *DCglobal* performed at the sink. The maintenance mechanism for this algorithm is developed in Section 4.2.

4.1 Algorithm for Selecting R-Nodes in DCglobal

In DCglobal, the sensor nodes should first report their readings and energy levels to the sink. Once collecting global information at the sink, DCglobal is able to determine the set of R-nodes. According to Definition 2, since the sink has the network topology and the readings of the sensor nodes, the data coverage ranges of the sensor nodes are easily generated. Since our goal is to extend the network lifetime of wireless sensor networks, R-nodes should have a sufficient amount of remaining energies. On the other hand, the number of R-nodes should be minimized since a larger number of R-nodes will quickly shorten the network lifetime. To reduce the number of R-nodes, one should select R-nodes with larger data coverage range. Note that to extend the network lifetime, sensor nodes with the maximal remaining energy should be first considered as candidate R-nodes. Then, if sensor nodes have the same maximal remaining energy, the sensor node with the maximal data coverage range becomes as one R-node, which will reduce the number of R-nodes selected. To allow more candidate R-nodes, the remaining energy of the sensor nodes is discretized. In other words, the remaining energy of the sensor nodes is represented as one energy level. Clearly, the granularity of energy levels has an impact on the network lifetime of wireless sensor networks, which is evaluated in our experiments. With the above design concept, a partial order relation $\preceq_{EnC}$ (standing for Energy and Coverage) between two sensor nodes is defined as follows.

Definition 3 (Partial Order Relation $\preceq_{EnC}$). *Let $S = \{s_1, \ldots, s_n\}$ be the set of sensor nodes, the data coverage range of s_i be C_i and the energy level of s_i be E_i. A partial order relation $\preceq_{EnC}$ is a binary relation over $S \times S$. For all $s_i, s_j \in S$, $s_i \preceq_{EnC} s_j$ if and only if $E_i < E_j$ or ($E_i = E_j$ and $C_i \subset C_j$).*

The philosophy of the partial order relation $\preceq_{EnC}$ is to assign priorities to sensor nodes. By employing $\preceq_{EnC}$, sensor nodes are first sorted based on their energy, and their data coverage ranges are used as a tie-breaker. A sensor node s_i is defined as the maximal sensor node if there is no s_j such that $s_i \preceq_{EnC} s_j$. When the sensor nodes are sorted by $\preceq_{EnC}$, the maximal sensor nodes have abundant energy and a wider data coverage range. Algorithm DCglobal iteratively selects the maximal sensor node as R-nodes in that a partial order list is iteratively updated in a round-by-round manner. Once the maximal sensor is selected as an R-node, those sensor nodes within its data coverage range are removed from the partial order list. Thus, two maximal sensor nodes cannot data-cover each other.

To realize the concept above, DCglobal first constructs a list of sensor nodes (referred to as a Plist), where sensor nodes are sorted in descending order by the partial order relation $\preceq_{EnC}$. In light of the Plist, DCglobal can select the maximal sensor node into the set of R-nodes, and the Plist is then updated. As such, algorithm DCglobal iteratively selects the maximal sensor node from the Plist until the union of the data coverage ranges of R-nodes includes all sensor nodes. The selection of R-nodes in algorithm DCglobal is in a round-by-round manner and in each round, one maximal sensor node is included in the set of R-nodes. It is possible that the maximal sensor nodes may have smaller data coverage ranges than other sensor nodes, and thus the number of R-nodes may increase. With the partial relation, both the data coverage ranges and the energy levels of the sensor nodes are considered in the algorithm DCglobal.

Algorithm 1. DCglobal
Input: S, the set of sensor nodes with their data coverage range and energy levels
Output: R, a set of R-nodes

```
1: Plist ← partial ordered set built by ⪯EnC ;
2: S' ← S
3: R ← φ
4: while S' ≠ φ do
5:    begin
6:    s_i ← pick one maximal sensor from Plist;
7:    for each sensor node s_j ∈ C_i do
8:       begin
9:       Remove s_j from S';
10:      Remove all elements related to s_j in Plist ;
11:      end
12:   Add s_i into R;
13:   Remove all elements related to s_i in Plist if it exists;
14:   end
15: Return R
```

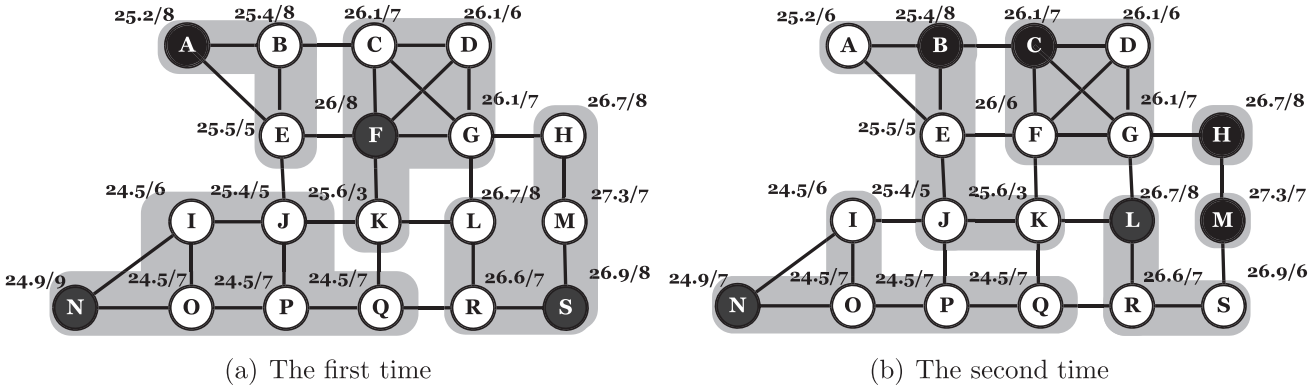
Consider an illustrative example in Fig. 1, where the data coverage range and energy level of each sensor node are shown in Table 1. Assume that the length of a reading vector is 1, the error threshold ϵ is set to 0.5, and $m = 10$. The running example of algorithm DCglobal is shown in Table 2, where R is the set of selected R-nodes and the sequences in the $Plist$ are sensor nodes listed in decreasing order in terms of $\preceq_{EnC}$. In the first round, the maximal sensor node is N, which has the highest energy level of all the sensor nodes. Moreover, it can be seen that in the *Plist* during the first round, sensor nodes A and S are in the same bucket because they have the same energy level but their data coverage ranges cannot contain each other (i.e., $\{\mathrm{A,B,E,J,K}\} \not\subseteq \{\mathrm{H,L,M,R,S}\}$ and $\{\mathrm{H,L,M,R,S}\} \not\subseteq \{\mathrm{A,B,E,J,K}\}$). On the other hand, although sensor node S and sensor node L have the same energy level, sensor node S is inferior to sensor node L because the data coverage range of S contains that of L (i.e., $\{\mathrm{L,R,S}\} \subset \{\mathrm{H,L,M,R,S}\}$). In the first round, sensor node N is selected as an R-node since N is the maximal sensor node in the $Plist$. Then, the $Plist$ is updated in that N and the sensor nodes covered by N are removed from the list. In the second round, sensor node A is selected as an R-node because A is the maximal sensor node in the $Plist$. In a similar way, sensor nodes F and S are selected to be R-nodes. When selected R-nodes are able to cover the whole set of sensor nodes in the network, DCglobal terminates. As a result, Fig. 3a shows that the set of R-nodes R is $\{N, A, F, S\}$. After selecting R-nodes as shown in Fig. 3a, we further investigate the next round in which DCglobal selects R-nodes. Suppose that the readings and data coverage ranges of sensor nodes do not change, and

TABLE 1
Data Coverage Range and Energy Level of Each Sensor Node in DCglobal

ID	Data Coverage Range	Energy	ID	Data Coverage Range	Energy
A	A,B,E,J,K	8	K	B,C,D,E,F,G,J,K	3
B	A,B,E,J,K	8	L	L,R,S	8
C	C,D,F,G,K	7	M	M,S	7
D	C,D,F,G,K	6	N	I,J,N,O,P, Q	9
E	A,B,E,F,J,K	5	O	I,N,O,P,Q	7
F	C,D,E,F,G,K	8	P	I,N,O,P,Q	7
G	C,D,F,G,K	7	Q	I,N,O,P,Q	7
H	H	8	R	G,H,L,R,S	7
I	I,N,O,P,Q	6	S	H,L,M,R,S	8
J	A,B,E,J,K	5			

TABLE 2
A Running Example of DCglobal

Rounds	Plist	R
1	(N) (A, B, F, S) (L, H) (C, G, M, O, P, Q, R) (D, I) (E) (J) (K)	N
2	(A, B, F, S) (L, H) (C, G, M, R) (D) (E) (K)	N, A
3	(F, S) (L, H) (C, G, M, R) (D) (K)	N, A, F
4	(S) (L, H) (M, R)	N, A, F, S



(a) The first time (b) The second time

Fig. 3. A final result of DCglobal, where numbers associated with a sensor node are *reading/energy level*.

the energy levels of previous R-nodes drain out 2 levels. The energy levels of sensor nodes are shown in Fig. 3b. In a similar fashion, the new set of R-nodes can be selected to be $\{B, C, N, L, H, M\}$. Note that R-nodes with large data coverage ranges may spend more energy than other sensors. However, such R-nodes may be replaced in the next round by those sensors with more energy. For example, S is selected as an R-node in the first round. However, in the second round, due to the energy decrease of S, the data coverage range of S is covered by new R-nodes H, M, and L whose energy levels are higher than S. By this feature, DCglobal can balance the loading of R-nodes such that the network lifetime can be effectively extended.

4.2 Maintenance Mechanism in DCglobal

This section presents a maintenance mechanism in which there are two scenarios to be considered. Explicitly, one scenario is that the set of R-nodes should be reselected if the R-nodes run low on energy, and the other scenario is that the R-nodes cannot cover those sensor nodes in their data coverage ranges. For the first scenario, since R-nodes need to report their readings, they may consume more energy than sensor nodes that are not R-nodes, and therefore, to extend the network lifetime, R-nodes should be reselected if they have less energy. For the second scenario, it is possible that an R-node cannot cover all sensor nodes in its data coverage range due to environmental changes. An R-node is called an invalid R-node if it can no longer cover sensor nodes in its data coverage range. The maintenance mechanism is derived for the above two scenarios.

During a data collection interval, it is possible that an R-node can become invalid because the environment may change. Thus, all sensor nodes, including R-nodes, should keep monitoring the variation of their current readings. Due to environmental changes, R-nodes may not cover sensor nodes within their data coverage ranges. For each maintenance interval, the proposed maintenance mechanism will be performed. For each sensor node, two reading vectors should be maintained to detect possible environmental changes: the current reading vector $v_i(t)$ and the last reading vector that performed the maintenance of DCglobal $v_i(t_p)$, where t_p is the last time DCglobal performed. At time

t_p, each sensor node (e.g., s_i) has its reading vector and the reading vector of the corresponding R-node (e.g., s_j). To detect environmental changes, each sensor node compares its reading vector at the current time t with that at time t_p. Suppose that the reading vector of the R-node s_j does not change after DCglobal is performed for the last time. Obviously, if the distance between the current reading of s_i and s_j is larger than ϵ, the sensor node s_i should ask its R-node s_j to further confirm whether s_i is covered by s_j. To guarantee $d(v_i(t), v_j(t_p)) < \epsilon$, $d(v_i(t), v_i(t_p))$ should be smaller than ϵ-$d(v_i(t_p), v_j(t_p))$. On the other hand, if $d(v_i(t), v_i(t_p))$ exceeds ϵ-$d(v_i(t_p), v_j(t_p))$, it is possible that R-node s_j cannot cover sensor node s_i due to the impact of environmental changes on the readings of sensor node s_i. At this time, sensor node s_i will ask its corresponding R-node s_j to send the current reading $v_j(t)$ for further comparison. Once $d(v_i(t), v_j(t))$ is larger than the error threshold ϵ, sensor node s_i will inform sensor node s_j of the invalidity. Then, when all sensor nodes in the data coverage range of s_j send their current readings and energy levels to the sink, the sink will perform DCglobal by giving a set of sensor nodes in the data coverage range of s_j.

4.3 Message Complexity

This section provides the complexity of messages involved in a data collection interval. Note that the messages used to determine R-nodes and adjust R-nodes in the maintenance mechanisms are referred to as control messages. In a data collection interval, the total number of messages are the sum of messages involved in determining R-nodes, re-executing DCglobal, and collecting data from R-nodes. Let the total number of sensor nodes be n. Supposing the sensors are randomly deployed in the network, the diameter of a sensor network is $O(\sqrt{n})$ [21]. Therefore, DCglobal needs $O(n\sqrt{n})$ to collect all readings from sensors and to inject the determination of R-nodes to sensors. Therefore, the total number of messages involved for re-execution is $O(\frac{f_c}{f_p} n\sqrt{n})$. In each maintenance, an invalid R-node should ask its members to send readings to the sink for reselecting R-nodes. The number of messages from a sensor to the sink is expected as $O(\sqrt{n})$. Thus, it needs $O(|C_j| \times \sqrt{n})$ to select new R-nodes for sensor nodes in the data coverage range s_j. Therefore, supposing there are on average ρ invalid R-nodes, the number of messages for maintenance should be at most $O(\rho \times C \times \sqrt{n})$, where C is the maximum data coverage range of the R-nodes. Let the proportion of selected R-nodes be δ. The data collection costs $\delta \times n \times \sqrt{n}$ messages. To sum them up, the total number of messages is at most $O((\delta + \frac{f_c}{f_p}) \times n\sqrt{n} + \frac{f_c}{f_m} \times \rho \times C \times \sqrt{n})$.

5 DClocal: Determining R-Nodes with Local Information

In Section 5.1, a distributed algorithm *DClocal* is proposed. In Section 5.2, the corresponding maintenance mechanism is developed.

5.1 Algorithm for Selecting R-Nodes in DClocal

Since algorithm DCglobal is a centralized algorithm, all computation and maintenance should be done in the sink. In large-scale sensor networks, DCglobal leads to a considerable number of message transmissions since all information should be collected at the sink. To reduce the number of message transmissions, DClocal, a distributed algorithm, is run at each sensor node for the selection of R-nodes. DClocal consists of two phases: the data collection phase and the R-node determination phase. In the data collection phase, each sensor node collects information from the sensor nodes within k-hops, where k is an adjustable parameter. Then, in the R-node determination phase, each sensor node could decide whether or not it should be an R-node. The details of each phase are described in the following sections.

5.1.1 Phase 1: Data Collection Phase

In order to reduce the number of message transmissions, each sensor node only disseminates its reading vectors within k-hop neighboring sensor nodes. Hence, each sensor node is able to collect the reading vectors and the energy levels of k-hop neighboring sensor nodes. According to the k-hop neighborhood information collected, a sensor node s_i can determine which sensor nodes would cover it, and these sensor nodes are included in a covering list, denoted as CL_i.[4] On the other hand, those sensor nodes that are covered by sensor node s_i are put in a covered list, expressed by covered list CL_i. Considering the covering list and the covered list, a sensor node s_i could then determine whether it should be an R-node or not. The value of k should be judiciously determined since with a larger k, the number of message transmissions is increased, whereas with a smaller value k, limited information is collected, thereby having an impact on the determination of R-nodes.

5.1.2 Phase 2: R-Node Determination Phase

Since DClocal is a distributed algorithm, each sensor node will self-determine whether it becomes an R-node or not. The R-node determination phase can be performed at each sensor node as the state transition diagram shown in Fig. 4, where there are five states. In the beginning, each sensor node starts in the *count* state, where according to the energy level E_i and covering list CL_i, the sensor node s_i will determine whether it should be an R-node or not. If a sensor node decides to become an R-node, it transits to the *covering-wait* state and invites sensor nodes in its covering list to join its own cluster.[5] After waiting for a certain period of time, sensor nodes in the covering-wait state will transit to the covering state and become R-nodes. At the same time, R-nodes will inform the sink of the R-node identifications and the corresponding covering lists. On the other hand, sensor nodes that are not candidate R-nodes are in the *covered-wait* state, in which sensor nodes join some R-nodes' clusters according to messages received from candidate R-nodes, after which the sensor nodes will be in the *covered* state. The messages used in DClocal are summarized in Table 3.

The transitions between states are controlled by counters, which is a common approach in distributed algorithms. The setting for the counters will be described later. The time duration for each counter to decrease by 1 is measured as

4. Here, we use the term *covering list* instead of *coverage range*. This is due to the fact that some sensor nodes may be covered by a sensor s_i but these sensors may be far than k-hops from s_i.

5. For ease of presentation, a cluster here represents the sensor nodes that an R-nodes can cover.

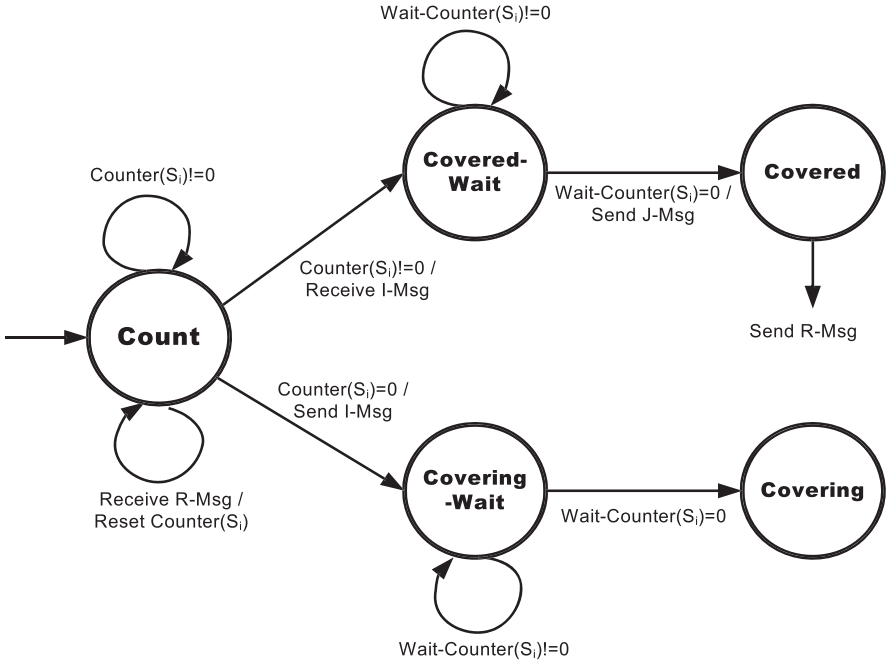


Fig. 4. A state transition diagram for algorithm DClocal.

the time it takes for the CPU to tick λ times. The value of λ can be determined by the computation ability of the CPU and the application requirements. In DClocal, once the counter of a sensor starts to count down, this sensor will not send messages to the other nodes. After finishing the counting-down procedure, if the sensor node needs to listen the messages from other sensor nodes, one should take this overhead incurred by MAC protocols into account in the setting of λ. For example, assume that the clock rate of MicaZ CPU is 16 MHz. Clearly, the tick time of MicaZ CPU is 62.5 ns. Suppose that λ is set to 1,000. Then, the counter will take $62.5\ \mu$s to decrease by 1. A detailed description of each state is presented as follows:

- **Count state.** In the count state, each sensor node competes with other sensor nodes to become an R-node. As pointed out earlier, to minimize the energy index for extending the network lifetime, sensor nodes with larger covering lists and higher energy levels should have a higher probability of becoming an R-node. To realize the above concept, for each sensor node, a counter is set and the value of the counter is determined by its covering list and energy level. Explicitly, the counter in sensor node s_i is set to $Counter(s_i) = \frac{n \times m}{|CL_i| \times E_i}$, where n is the total number of sensor nodes, m is the energy level when the sensor node's battery is full, $|CL_i|$ denotes the number of sensor nodes in the covering list of sensor node s_i, and E_i is its current energy level. Once the counter of a sensor node is set, it decreases as time passes. When the counter of sensor node s_i counts down to zero, this sensor node will become an R-node and the state of sensor node s_i is transited to the covering-wait state. At the same time, sensor node s_i sends invitation messages (referred to as *I-Msg*s) to those sensor nodes within its covering list CL_i.

  In the count state, if s_i does not receive any messages, s_i is still in the count state and $Counter(s_i)$ keeps decreasing. Otherwise, if s_i receives an *I-Msg*, s_i changes itself to the covered-wait state because it is covered by another sensor node. On the other hand, if s_i receives an *R-Msg*, the counter value is modified because some sensor nodes in CL_i are covered by other R-nodes. Formally, let X be the sensor nodes that send *R-Msg*s to s_i. The counter value *Counter(s_i)* is modified as $\max(0, \frac{n \times m}{|CL_i^{'}| \times E_i} - \tau)$, where $CL_i^{'} = \bigcup_{s_j \in X} s_j$ and τ is the time passed since the beginning. When a sensor node receives an *R-Msg*, the size of CL_i may decrease. Based on the design of DClocal, if a sensor node only covers a smaller number of sensors, the probability that this sensor node will become an R-node is thus smaller. Therefore, the counter value $Counter(s_i)$ is adjusted according to the number of uncovered sensor nodes in CL_i.
- **Covering-wait state.** If a sensor node is in the covering-wait state, it will become an R-node. To determine the set of sensor nodes covered, sensor nodes send I-Msgs with their CL_i to invite other nodes to join their clusters. Hence, a waiting time should be set and, for that purpose, a waiting counter is devised. Assume that sensor node s_i is in the covering-wait state and the waiting counter in sensor

TABLE 3
Messages Used in Algorithm DClocal

Message Type	Descriptions
I-Msg	Invite sensor nodes to join one R-node's cluster
J-Msg	Join R-node's cluster
R-Msg	Notify a sensor node to resize its covering list

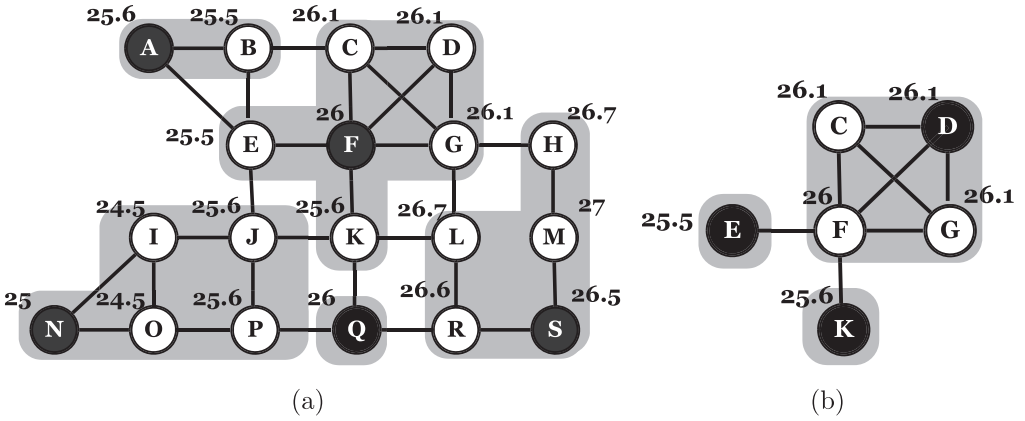


Fig. 5. An illustrative example: (a) the maintenance mechanism of DClocal (the shaded areas are the previous covering lists) and (b) selecting R-nodes from one covering list.

node s_i is formulated as *Wait-Counter*$(s_i)= \alpha \times \frac{|CL_i|}{n}$, where α is a constant, $|CL_i|$ is the number of sensor nodes in its coverage list, and n is the total number of sensor nodes. If a sensor node can cover more sensor nodes, it should wait for a longer period due to the time it takes for message propagation among the sensor nodes. Similarly, the counter counts down as time goes by. During the waiting time, those sensor nodes in the covering-wait state receive join messages (abbreviated as J-Msgs) from sensor nodes in the covered-wait state. Note that when the *Wait-Counter*(s_i) becomes zero, the state of sensor node s_i is transited to the covering state.

- **Covering state.** When sensor node s_i is in the covering state, it determines the set of sensor nodes covered as its covering list according to the J-Msgs received. Then, sensor node s_i informs the sink that it has become an R-node.
- **Covered-wait state.** A sensor node in the covered-wait state should decide to join one cluster of R-nodes. It is possible that sensor node s_j receives I-Msgs from other sensor nodes in the covering-wait state. To reduce the number of R-nodes, sensor nodes in the covering-wait state should select sensor nodes with larger covering lists. Note that only the covering list is considered since sensor nodes in the covering-wait state should have more energy compared with other sensor nodes in other states. To decide which cluster to join, sensor node s_j should wait for more I-Msgs. Thus, the wait time is also formulated as *Wait-Counter*$(s_j)= \beta \times \frac{|L_j|}{n}$, where β is a constant, n is the total number of sensors and $|L_j|$ is the number of sensor nodes that could cover s_j. For a larger scale wireless sensor network and larger k, β should be increased to wait for more I-Msgs, providing more opportunities to join an R-node with a larger set of sensor nodes covered. When *Wait-Counter*(s_j) counts down to zero, sensor node s_j selects the R-node s_i with the maximal CL_i. Note that it is possible that s_j receives two or more R-nodes with the same size covering lists. In this case, s_j randomly joins one of these R-nodes as a tie-breaker. Once a sensor node s_j decides that it will be covered by s_i, s_j will send a *J-Msg* to s_i and the state of s_i becomes the covered state.
- **Covered state.** When sensor node s_j joins the cluster of an R-node, sensor node s_j should send an R-Msg to notify the sensor nodes that could cover s_j. This is because the sensor nodes covering s_j can no longer cover it, and thus, if the sensor nodes in L_j are in the count state, they should adjust their covering lists and counters.

5.2 Maintenance Mechanism of DClocal

As in DCglobal, the maintenance mechanism will be executed every f_m time units. As pointed out, the maintenance mechanism should deal with two scenarios: one is that R-nodes run low on energy and the other is that R-nodes can no longer fully cover sensor nodes due to environmental changes. For the above two scenarios, our maintenance will perform the following operations:

5.2.1 Scenario: R-Nodes Run Low on Energy

These valid R-nodes should be replaced by other sensor nodes for energy balance. Notice that valid R-nodes have reading vectors of sensor nodes within their covering lists. Intuitively, one valid R-node with low energy could perform algorithm DCglobal to reselect R-nodes among sensor nodes in its covering list. It is possible that only a limited number of sensor nodes (i.e., sensor nodes in one covering list) is considered, thereby increasing the number of R-nodes. For example in Fig. 5a, assume that node F is a valid R-node with lower energy. Nodes D, E and K are selected as new R-nodes that form the three clusters shown in Fig. 5b. As can be seen in Fig. 5b, the covering lists of newly selected R-nodes are smaller and the number of R-nodes is larger, which reduces the network lifetime. Thus, more sensor nodes should be included when valid R-nodes execute the algorithm DCglobal.

To include more sensor nodes, we propose a distributed method to explore the possibility of the merging covering lists of nearby valid R-nodes. Due to the nature of distributed algorithms, the whole procedure can be represented as a state transition diagram. According to the messages received and the counter values, each R-node will be in a different state. Fig. 6 shows the transition diagram executed in each valid R-node, where there are four states: the explore state, the

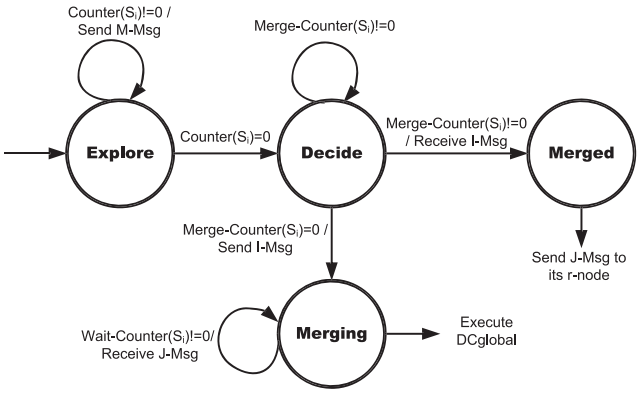


Fig. 6. A state transition diagram for valid R-nodes.

decide state, the merging state and the merged state. Specifically, in the explore state, a valid R-node s_i sets $counter(s_i)$ to be a constant MAX and broadcasts merge-messages (abbreviated as *M-Msg*) to discover k-hop valid R-nodes. The *M-Msg* of s_i includes its covering list CL_i, its own reading vector $x_i(t)$ and one reading vector $y_i(t)$, where $d(x_i(t), y_i(t))$ is the maximal among all distance values for sensor nodes in its covering list. For example, in Fig. 5a, the *M-Msg* of F contains its covering list: $\{C, D, E, F, G, K\}$, 26 (its reading vector), and 25.5 (the reading vector of E). The counter value $counter(s_i)$, which will decrease as time goes by, is the waiting time for receiving M-Msg messages from other valid R-nodes. After receiving M-Msgs from other valid sensor nodes (e.g., s_j), if $d(x_i(t), x_j(t)) \leq \epsilon$ and $d(x_i(t), y_j(t)) \leq \epsilon$, valid R-node s_i will include s_j in the set Ψ_i. Consider the example in Fig. 5a, where node F is in the set Ψ_Q because $d(F, Q) \leq 0.5$ and $d(E, Q) \leq 0.5$. When $counter(s_i)$ becomes zero, s_i transits itself to the decide state.

In the decide state, a valid R-node s_i with a larger set Ψ_i should have a higher probability of merging the covering lists of the valid R-nodes in Ψ_i. Thus, we set a merge-counter(s_i) to be $\frac{MAX}{2^{|\Psi_i|}}$, and the merge-counter will count down. Once the merge-count is zero, valid R-node s_i will send an *I-Msg* to include R-nodes in Ψ_i. Moreover, the state of s_i is set to the merging state. However, if s_i receives any *I-Msg*s while the merge-counter is counting down, s_i goes to the merged state. In the merging state, s_i waits to receive a *J-Msg* from other valid R-nodes by setting Wait-Counter$(s_i) = MAX$ time slots. Then, s_i performs localized DCglobal with the union of covering lists of R-nodes in Ψ_i. On the contrary, if the state of s_i is in the merged state, s_i will send a J-Msg to the valid R-node whose I-Msg is received and the corresponding covering list is maximal, which could be determined from the M-Msg messages received.

5.2.2 Scenario: R-Node Cannot Fully Cover Sensor Nodes in its Range

Due to environmental changes, some sensor nodes may not be covered by their R-nodes. Each R-node s_j maintains the reading vectors of sensor nodes in its covering list at time t_p. For maintenance, each R-node s_j compares its reading vector at the current time $v_j(t)$ with the reading vector of each sensor node s_i in CL_j at the previous time $v_i(t_p)$. Therefore, an R-node s_j can partition the sensor nodes in CL_j into two groups: 1) $\Delta = \{s_i | d(v_j(t), v_i(t_p)) < \epsilon\}$, and 2) $\overline{\Delta} = \{s_i | d(v_j(t), v_i(t_p)) > \epsilon\}$. Each R-node s_j considers that all sensor nodes in Δ can be covered and all sensor nodes in $\overline{\Delta}$ are no longer covered by s_j. Therefore, an R-node s_j sets the covering list CL_j as Δ, which represents that s_j covers only sensor nodes in Δ now. On the other hand, it is necessary to select new R-nodes for sensor nodes in $\overline{\Delta}$. A naive method is that each sensor node in $\overline{\Delta}$ is viewed as an R-node and then executes the merge operation mentioned in the previous section. However, it can be observed that the readings of sensor nodes usually have spatial locality. That is, it is highly possible that readings of nearby sensor nodes may change in a similar way. Therefore, an R-node s_j can further partition these sensors into groups such that sensors v_i and v_ℓ in the same group satisfy $d(v_i(t_p), v_\ell(t_p)) < \epsilon$. For each group, the R-node s_j will select the sensor node with the largest energy levels as a new R-node. Finally, s_j will send messages to all the sensor nodes in $\overline{\Delta}$ to notify them of the information regarding the groups and new R-nodes. For example, in Fig. 5a, note that N can no longer cover sensor nodes J and P. The R-node N will divide the sensors in its covering list into $\Delta = \{N, I, O\}$ and $\overline{\Delta} = \{J, P\}$. Thus, sensor node N will shrink its covering list to $\{N, I, O\}$. Moreover, since the readings of J and P are 25.6, they are clustered into the same group and P could be selected as the new R-node for this group. Once the new R-nodes received the messages from the original R-node, they will execute the merge operation proposed in the previous section to further discover the R-nodes that can be merged. Following the example above, P is a new R-node selected by N. It can be observed that there is an R-node Q, where the difference of reading vectors between Q and P is smaller than 0.5. Moreover, the size of the covering list of P is larger than that of Q. When P executes the merge operation, Q can be included in the covering list of N. Thus, the number of new R-nodes can be further decreased such that the energy cost for data collection can be reduced.

5.3 Message Complexity

This section provides the complexity of messages involved in a data collection interval. In DClocal, there are some control messages, including I-Msg, J-Msg, and R-Msg. These messages are used for R-node selection and maintenance mechanisms. Same as DCglobal, the total number of messages in a data collection interval is the sum of messages involved in determining R-nodes, re-executing DClocal, and collecting data from R-nodes. Let the total number of sensor nodes be n. In periodically executing DClocal to reselect R-nodes, each sensor s_i is required to collect information of sensors which are within k-hops to s_i. Assume that the sensor nodes are deployed randomly and uniformly. The number of sensor nodes within k-hops to s_i is proportional to $\pi \times k^2 = O(k^2)$. Thus, collecting data needs at most $O(n \times k^2)$ messages. After collecting all information from k-hop sensors, each sensor sets its counter and starts to count down. Supposing that the proportion of R-nodes is δ, there are totally $\delta \times n$ R-nodes to send I-Msgs to their members. The sensors receiving I-Msgs (totally $(1 - \delta) \times n$) will send back J-Msgs to confirm. At most $O(k)$ messages are involved in sending J-Msgs to the R-node of a sensor node. Therefore, the total number of messages for R-node determination is at

most $\frac{f_c}{f_p}(\delta \times n \times O(k^2) + (1-\delta) \times n \times O(k))$. In each maintenance operation, once an R-node becomes invalid, those sensor nodes which are no longer covered by this particular R-node become R-nodes themselves. Then, the merging operation is executed to discover R-nodes that can be merged. Supposing that the proportion of such R-nodes is ξ, there are at most $\delta \times n \times \xi$ new R-nodes when the original R-nodes become invalid. These new R-nodes need $O(k^2)$ messages to discover R-nodes that can be possibly merged. Therefore, the total number of messages for maintenance is $O(\frac{f_c}{f_m} \times \delta \times n \times \xi \times k^2)$. Finally, the data collection involves at most $O(\delta \times n \times \sqrt{n})$ messages. To sum them up, the total number of messages of DClocal during a data collection interval is $O(\delta n((\frac{f_c}{f_p} + \frac{f_c}{f_m}\xi)k^2 + \frac{f_c}{f_p}\frac{1-\delta}{\delta}k + \sqrt{n}))$.

6 Performance Evaluation

In this section, we evaluate our proposed algorithms compared with existing works. All experiments are conducted using both a synthesis data set and a real data set. Sensitivity analysis of DCglobal and DClocal is investigated.

6.1 Setting and Data Sets

To simulate the sensor network environment to be as close as possible to reality, we adopt the *ns-2 network simulator* for the following experiments [1]. As aforementioned, the goal of this paper is to select R-nodes to maximize network lifetime. Therefore, the settings of the MAC layer in our simulation are taken from [31]: the radio energy for rxPower (receiving) is 0.014 Watts and that for txPower (transmitting) is 0.025 Watts. In the listening mode, the idle power of the radio is 0.014 Watts, which is equal to the rxPower in the receive mode. The initial energy of a sensor node is 100 J. Similar to [24], we use S-MAC as our MAC layer protocol. The duty cycle is set as 0.3. We deploy 500 sensor nodes in a uniformly-random manner over a $100 \times 100\ \mathrm{m}^2$ area. The sink is located at $(1, 1)$. According to the Mica2 specification [6], the transmission range of one sensor node is 100 m. The total simulation time is set to 4,000 time units. The data collection interval f_c is 200 time units. To maintain the R-nodes, the R-node selection algorithms (DCglobal and DClocal) are executed every $f_p = 50$ time units, and the maintenance mechanism (fine tuning) is executed every $f_m = 25$ time units. All experimental results are obtained by averaging the results over 50 simulation runs. We employ Euclidean distance as our distance function, and the default error threshold is $3\ ^\circ\mathrm{C}$. The window size is set to 10 time units. These parameters are the default settings in the following experiments unless otherwise specified.

The environmental change measured by the sensors can be obtained in two ways. The first approach is to generate synthetic temperature readings on the monitored region, whereas the second method is to obtain readings from a real data set. In the synthetic data set, 20 events $\{e_1, e_2, \ldots, e_{20}\}$ are generated. These events are uniformly distributed over the monitored region. Sensor readings are affected by these events, and the influence of each event on a sensor is inversely proportional to their distance. The initial temperature in the center of an event is randomly selected from $[20, 30]\ ^\circ\mathrm{C}$. This range is determined by the temperature range of the Intel Lab data set [18]. The value of an event e_i at time t is formulated as $e_i(t) = e_i(t-10) + X$, where X is a random variable that follows the standard normal distribution and $e_i(0)$ is the initial temperature of this event. As mentioned previously, the readings of a sensor node at time t can be affected by all events. Formally, the reading of a sensor node s at time t can be set as follows:

$$reading_s(t) = \sum_{i=1}^{20} \left(\frac{\sqrt{dist(s, e_i)}}{\sum_{j=1}^{20} \sqrt{dist(s, e_j)}} \right) * e_i(t),$$

where $dist(s, e_i)$ is the euclidean distance between sensor node s and event e_i. On the other hand, for the real data set, we use the publicly available Intel Lab data set [18] that consists of 54 sensor nodes, and we use the light readings of the sensor nodes from this data set. In addition, due to missing readings and asymmetric communication between two sensor nodes, we fill missing readings with previous readings and consider two sensor nodes to be connected if the probability of packet loss is less than 50 percent. All experimental results are of the average performance from readings taken over ten days. Note that the synthetic data set is used to simulate a large-scale sensor network, whereas the real data set is used to evaluate the performance in a real world environment.

To evaluate the experimental results, three performance metrics, the network lifetime, the number of R-nodes and the clustering cost, are used. Specifically, the *network lifetime* measures the length of time until the first sensor node battery is depleted.[6] Note that the network lifetime is mainly affected by the energy cost involved in selecting R-nodes and reporting the readings of the R-nodes. Therefore, to get further insight into the performance of these algorithms, the *number of R-nodes* shows the average number of selected R-nodes under each algorithm. Furthermore, the *clustering cost* is used to measure the amount of energy involved in R-node selection and the maintenance mechanisms. As pointed out early, the problem dealt with in this paper is to exploit spatial correlations without any aggregation operators and probabilistic models. There are two research works (i.e., *EEDC* [16], *Snapshot* query [13]) that address the same problem. Thus, both Snapshot and EEDC are implemented. Furthermore, we borrow the concept in [29] that explores k-hop dominating sets in mobile ad hoc networks for R-node selections. This approach is called *DomSet* [29]. Specifically, Snapshot builds a linear regression prediction model to estimate the readings of a sensor node's *one-hop* neighbors and to select R-nodes which can represent their one-hop neighbors. EEDC is a centralized algorithm that explores the clique-covering problem for approximate data collection. Explicitly, in EEDC, all of the sensor nodes are partitioned into several cliques, where sensor nodes in each clique have their distance values no larger than the error threshold required. In each clique, the R-node can be assigned by any sensor nodes within the same clique. Rather than the round-robin approach proposed in [15], we implement the randomized intracluster scheduling method proposed in [16] to activate multiple R-nodes at an instant. The readings of each sensor node are represented as line segments by the piecewise linear approximation approach proposed in [16]. DomSet is a distributed algorithm that finds a k-connected dominating set as a virtual backbone for an ad hoc network. The k-connected dominating set refers to a set of nodes in which 1) the nodes are connected, and 2) the farthest distance

6. Since our experiments are executed in ns2, the unit of time is a time slot in this simulator.

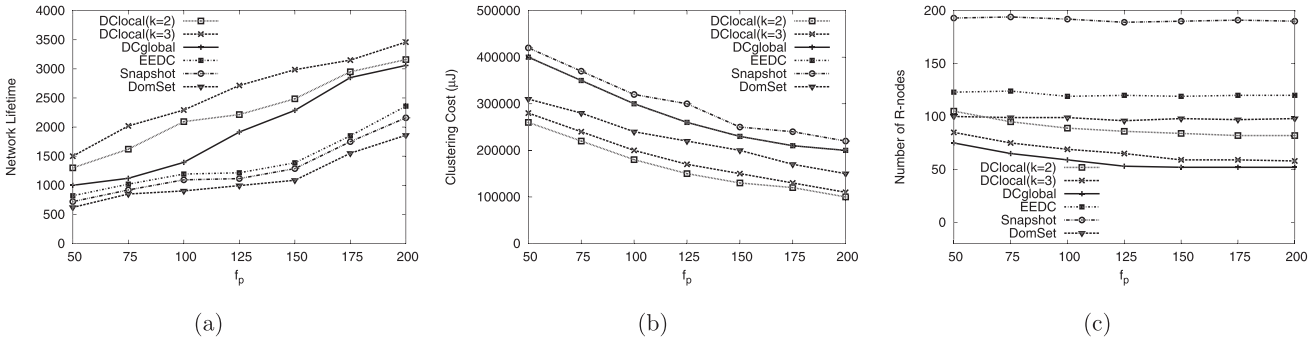


Fig. 7. The impact of re-execution intervals in the synthesis data set: (a) network lifetime, (b) clustering cost, and (c) number of R-nodes.

between each node outside the set to some node in the set does not exceed k-hop. In the following experiments, k is set to be 2. In addition, DomSet proposes a mechanism to maintain the connectivity of dominating sets derived in mobile ad hoc networks. In this paper, the application scenario is a static sensor network. For fair comparison, a communication connectivity graph is given, where two nodes are connected if they can communicate with each other and the distance of their current reading vectors is smaller than ϵ.

6.2 Comparison of R-Node Selection Algorithms

To maintain the validity of R-nodes, the R-node selection algorithm will be executed for every f_p time units, i.e., each algorithm should be executed for R-node selection. In brief, f_p is called the re-execution interval in the following. To investigate the impact of various re-execution intervals f_p, we compare the performance of EEDC, Snapshot, DomSet, DCglobal, and DClocal in terms of the network lifetime, number of R-nodes, and clustering cost. For DCglobal and DClocal, the R-node adjusting maintenance interval f_m is set to $\frac{f_p}{2}$ by default. The optimal setting of f_m will be discussed in the sensitivity analysis later.

Fig. 7a depicts the network lifetime for various algorithms in the synthesis data set. This figure shows that a longer re-execution interval leads to a longer network lifetime for all algorithms. Intuitively, without frequently reselecting R-nodes, the energy consumption may be reduced and the network lifetime can be extended. This phenomenon can be observed in Fig. 7b. With the increased length in the re-execution interval, the energy consumption for selecting R-nodes is reduced significantly. By comparing the network lifetime between algorithms, it can be observed that DCglobal and DClocal lead to longer network lifetimes than EEDC, Snapshot, and DomSet. Moreover, a large value of k may have more benefit for DClocal because the covering list of each sensor can be increased such that an R-node represents more sensor nodes. This can be observed in Fig. 7c, which shows that the number of R-nodes derived by $\mathrm{DClocal(k=3)}$ is much a smaller than that derived by $\mathrm{DClocal(k=2)}$. The number of R-nodes is proportional to the energy consumption for data collection. In addition, the clustering costs of $\mathrm{DClocal(k=2)}$ and $\mathrm{DClocal(k=3)}$ do not differ too much. Therefore, $\mathrm{DClocal(k=3)}$ has much longer network lifetime than $\mathrm{DClocal(k=2)}$. Note that even though DCglobal can derive fewer R-nodes than DClocal as shown in Fig. 7c, DCglobal still has a shorter network lifetime than DClocal since the clustering cost for the former is much higher than for the latter. Therefore, if DCglobal is executed frequently, it may easily drain out the energy of the sensor nodes. On the other hand, if the re-execution interval is increased, the network lifetime for DCglobal can be extended significantly since the dominant energy cost is due to data collection in this case. An interesting observation from this experiment is that DomSet selects fewer R-nodes than EEDC and Snapshot, as shown in Fig. 7c. However, as shown in Fig. 7a, DomSet cannot extend network lifetime to be longer than EEDC and Snapshot. Note that the network lifetime is proportional to the energy consumption of the clustering cost and the reporting cost. From Fig. 7a, DomSet does not consume more energy in clustering cost. Since the R-nodes selected by DomSet should connect to the sink, it is possible that the main energy consumption of DomSet comes from the bottleneck of R-nodes close to the sink.

Since the network size in the real data set is smaller than that in the synthesis data set, the energy of a sensor node is adjusted to 5 J for studies of the network lifetime. The network lifetime of all algorithms is shown in Fig. 8a, where DCglobal has the longest network lifetime compared with the other algorithms. As shown in Fig. 8b, a smaller network can afford the energy cost to send readings from sensor nodes to the sink. Compared with the centralized algorithms (i.e., DCglobal, EEDC), the distributed algorithms (i.e., DClocal, Snapshot, DomSet) cannot have the advantage in terms of clustering cost. Moreover, Fig. 8c shows that the number of R-nodes derived by DCglobal and DClocal are almost the same. Therefore, by selecting fewer R-nodes, DCglobal outperforms the other algorithms. As for the other algorithms, compared with the previous experimental results, DomSet leads to longer network lifetime than Snapshot in a small-scale network. In a small-scale network, there are few sensors so that each sensor is easy to relay the data from other sensor nodes, i.e., become an intermediate node of the routing tree. Although R-nodes selected by DomSet are required to be connected, these R-nodes are almost the intermediate sensors of the routing tree. Thus, the connected R-nodes will not reduce the network lifetime significantly in a large-scale network. Since the clustering cost of all algorithms is almost the same, the number of R-nodes is a dominant factor that affects the network lifetime. Selecting fewer R-nodes, DomSet thus has longer network lifetime than Snapshot.

From the above experiments, it can be seen that the data collection interval has a great impact on the performance of all of the algorithms. This can be summarized by stating that DCglobal is able to select the smallest number of R-nodes because, with the global information it collects, DCglobal is able to derive wider data coverage ranges, and

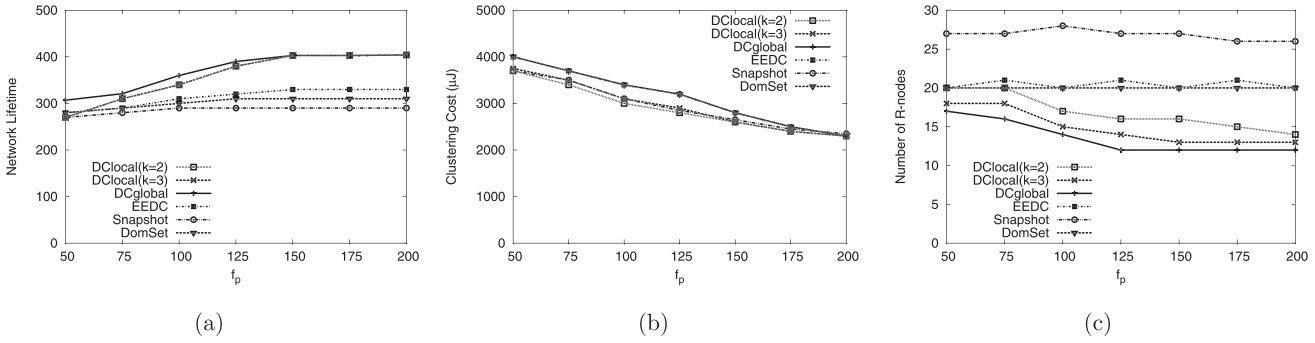


Fig. 8. The impact of re-execution intervals in the real data set: (a) network lifetime, (b) clustering cost, and (c) number of R-nodes.

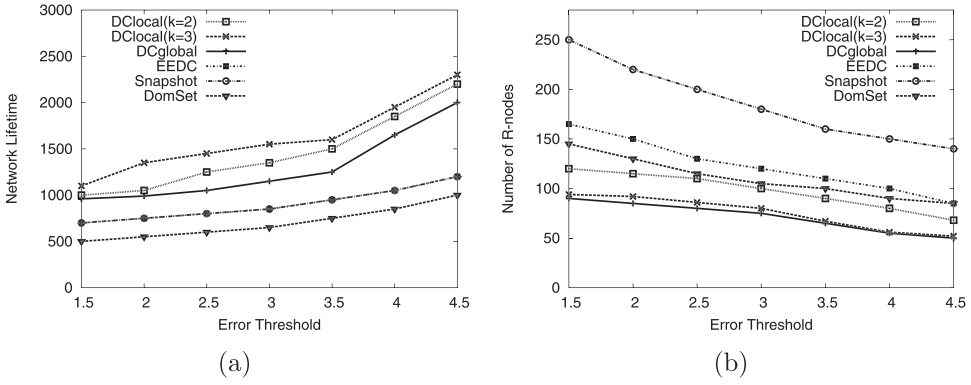


Fig. 9. The impact of error thresholds in the synthesis data set: (a) network lifetime and (b) the number of R-nodes.

thereby generate a smaller number of R-nodes. With a larger data collection interval, the network lifetime is mainly affected by the clustering cost. For a large-scale wireless sensor network, DClocal is likely to have a smaller clustering cost. However, for a small-scale wireless sensor network, DCglobal has the lowest clustering cost. Thus, depending on the data collection interval settings and the scale of the wireless sensor network, DCglobal and DClocal should be utilized appropriately for extending the network lifetime of wireless sensor networks.

6.3 Effect of Error Thresholds

The effect of the error threshold ϵ is investigated next. This threshold is to set a tolerable range between two reading vectors. Once the distance between two reading vectors is smaller than ϵ, they can be used to represent each other. For a larger threshold, on sensor may potentially cover more sensors. Therefore, this threshold mainly affects the number of R-nodes. For the synthesis data set, Fig. 9a shows the network lifetime of wireless sensor networks for various algorithms. Network lifetime is extended as the error threshold increases. Clearly, with larger tolerable error thresholds, all algorithms are able to select fewer R-nodes as shown in Fig. 9b, where it can be seen that with a larger error threshold, the number of R-nodes for DCglobal and DClocal is significantly less than that for EEDC, DomSet, and Snapshot. For the real data set, Fig. 10a demonstrates similar results as those for the synthesis data set. As can be seen in Fig. 10b, the number of R-nodes decreases as the error threshold increases. The above experiments indicate that with a larger error threshold, the network lifetime for both DCglobal and DClocal is longer than for other algorithms, thereby highlighting the advantage of using the concept of data coverage for selecting R-nodes.

6.4 The Impact of k on DClocal

In DClocal, the value of k plays an important role that determines how far a sensor node distributes its own readings and gathers information on other nearby sensor nodes. Note that with a larger k, each sensor node may discover a larger data coverage range. Clearly, a larger k incurs a larger number of messages, thereby reducing the network lifetime. However, a smaller k restricts the possibility of discovering a larger data coverage range, thus increasing the number of R-nodes. Hence, it is an important issue to set an appropriate value of k for DClocal. We then investigate the comparison of DClocal with the value of k varied, where DCglobal is the base line for comparison.

Fig. 11a shows the network lifetime of DClocal with k varied. This figure shows that setting $k = 3$ for DClocal can lead to the longest network lifetimes in this case. It can be observed that the network lifetime of DClocal with $k = 1$, $k = 6$, and $k = 7$ are shorter than those of DCglobal. As mentioned before, the network lifetime is affected by the clustering cost and the number of R-nodes. In the case of $k = 1$, although the clustering cost is lowest among all cases, the number of R-nodes selected is much more than for the other cases. Therefore, the network lifetime is shortened because a greater proportion of energy will be consumed for data collection. On the other hand, even though the numbers of R-nodes selected by DClocal with $k = 6$ and $k = 7$ are very close to those selected by DCglobal, the clustering costs of these two cases are higher than the centralized algorithm DCglobal. In this case, the network lifetime cannot be longer than DCglobal because DClocal cannot take any advantage by executing R-node selection in a distributed manner. These experiments show that the setting of k should strike a compromise between the clustering cost and the number of

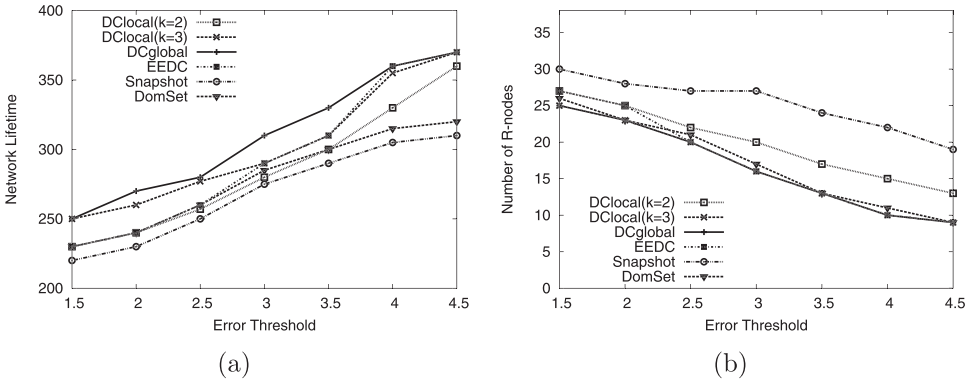


Fig. 10. The impact of error thresholds in the real data set: (a) network lifetime and (b) the number of R-nodes.

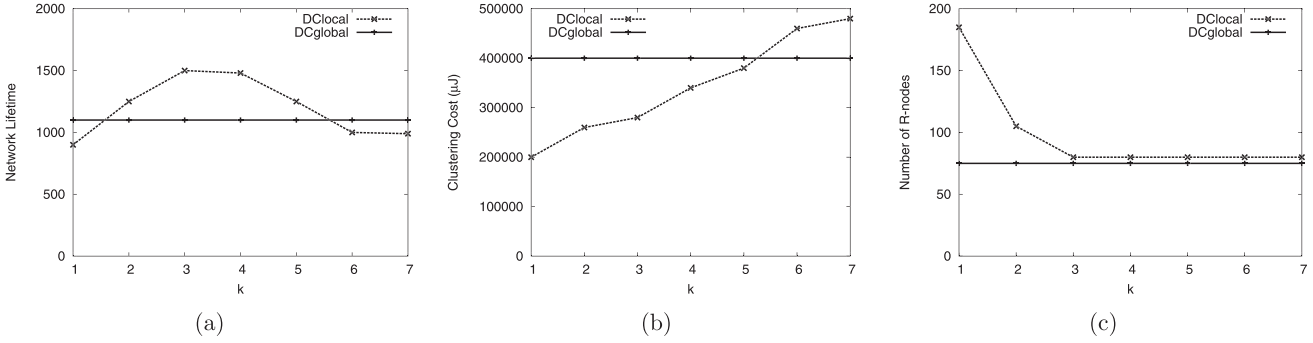


Fig. 11. The impact of k in the synthesis data set: (a) network lifetime, (b) clustering cost, and (c) the number of R-nodes.

R-nodes such that the network lifetime can then be maximized. Based on these results, the default value of k is set to 3 in our experiment. From the above experiments, one could see that the setting of k has a great impact on the performance of DClocal and can be empirically determined.

7 Conclusions

In this paper, we have addressed the problem of selecting a set of R-nodes for approximate data collection in a wireless sensor network. Specifically, we argued that the number of R-nodes could be reduced by solving an *energy-aware set-covering problem*, an extension of a set-covering problem. In an energy-aware set-covering problem, the R-nodes selected should have higher energy and wider data coverage ranges, while the union of their respective data coverage ranges should be the set of all sensor nodes. Consequently, we proposed an algorithm *DCglobal*, which is able to derive a set of R-nodes with the most available energy and wide data coverage ranges. Moreover, for a large-scale network, since collecting readings and energy information from all sensor networks incurs a considerable number of message transmissions in the selection of R-nodes, we have also developed a distributed algorithm, *DClocal*. By exchanging nearby information, each sensor node can decide whether it should be an R-node or not. In addition, maintenance mechanisms for DCglobal and DClocal were proposed to efficiently select alternative R-nodes for those with low energy, and to reflect changes in the environment. Experimental results on both synthesized and real data sets show that both DCglobal and DClocal are able to significantly extend the network lifetime in comparison to prior works.

Acknowledgments

Wen-Chih Peng was supported in part by the National Science Council, Project No. 100-2218-E-009-016-MY3 and 100-2218-E-009-013-MY3, by Taiwan MoE ATU Program, by the Theme project of Academia Sinica, Project No. AS-102-TP-A06, by D-Link and by Microsoft.

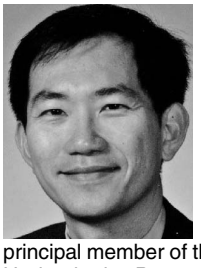
References

[1] *The Network Simulator - ns2,* http://www.isi.edu/nsnam/ns/, 2012.
[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks,* vol. 38, no. 4, pp. 393-422, 2002.
[3] Q. Cao, T.F. Abdelzaher, T. He, and J.A. Stankovic, "Towards Optimal Sleep Scheduling in Sensor Networks for Rare-Event Detection," *Proc. Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 20-27, 2005.
[4] J. Chou, D. Petrovic, and K. Ramchandran, "A Distributed and Adaptive Signal Processing Approach to Reduce Energy Consumption in Sensor Networks," *Proc. INFOCOM,* 2003.
[5] D. Chu, A. Deshpande, J.M. Hellerstein, and W. Hong, "Approximate Data Collection in Sensor Networks Using Probabilistic Models," *Proc. 22nd Int'l Conf. Data Eng. (ICDE),* pp. 48-59, 2006.
[6] Crossbow Technology Co. Mica2 Datasheet, http://www.xbow.com, 2012.
[7] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* second ed. MIT Press, 2001.
[8] A. Deshpande, C. Guestrin, S. Madden, J.M. Hellerstein, and W. Hong, "Model-Driven Data Acquisition in Sensor Networks," *Proc. 13th Int'l Conf. Very Large Data Bases (VLDB),* pp. 588-599, 2004.
[9] H. Gupta, V. Navda, S.R. Das, and V. Chowdhary, "Efficient Gathering of Correlated Data in Sensor Networks," *Proc. MobiHoc,* pp. 402-413, 2005.
[10] S.C.-H. Huang, P.-J. Wan, C.T. Vu, Y. Li, and F.F. Yao, "Nearly Constant Approximation for Data Aggregation Scheduling in Wireless Sensor Networks," *Proc. INFOCOM,* pp. 366-372, 2007.

[11] C.-C. Hung and W.-C. Peng, "Optimizing In-Network Aggregate Queries in Wireless Sensor Networks for Energy Saving," *Data and Knowledge Eng.,* vol. 70, no. 7, pp. 617-641, 2011.
[12] H.V. Jagadish, A.O. Mendelzon, and T. Milo, "Similarity-Based Queries," *Proc. 14th ACM Symp. Principles of Database Systems (PODS),* pp. 36-45, 1995.
[13] Y. Kotidis, "Snapshot Queries: Towards Data-Centric Sensor Networks," *Proc. 21st Int'l Conf. Data Eng. (ICDE),* pp. 131-142, 2005.
[14] H. Lee and A. Keshavarzian, "Towards Energy-Optimal and Reliable Data Collection via Collision-Free Scheduling in Wireless Sensor Networks," *Proc. INFOCOM,* pp. 2029-2037, 2008.
[15] C. Liu, K. Wu, and J. Pei, "A Dynamic Clustering and Scheduling Approach to Energy Saving in Data Collection from Wireless Sensor Networks," *Proc. IEEE Second Ann. Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (SECON),* 2005.
[16] C. Liu, K. Wu, and J. Pei, "An Energy-Efficient Data Collection Framework for Wireless Sensor Networks by Exploiting Spatio-temporal Correlation," *IEEE Trans. Parallel and Distributed System,* vol. 18, no. 7, pp. 1010-1023, July 2007.
[17] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay Efficient Sleep Scheduling in Wireless Sensor Networks," *Proc. INFOCOM,* pp. 2470-2481, 2005.
[18] S. Madden, *Intel Lab Data 2004,* http://db.csail.mit.edu/labdata/labdata.html, 2012.
[19] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *Proc. Fifth Symp. Operating Systems Design and Implementation (OSDI),* 2002.
[20] A. Meka and A.K. Singh, "Distributed Spatial Clustering in Sensor Networks," *Proc. 10th Int'l Conf. Advances in Database Technology (EDBT),* pp. 980-1000, 2006.
[21] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table," *ACM Mobile Networks and Applications (MONET),* vol. 8, no. 4, pp. 427-442, 2003.
[22] X. Tang and J. Xu, "Optimizing Lifetime for Continuous Data Aggregation with Precision Guarantees in Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 14, no. 4, pp. 904-917, Aug. 2008.
[23] X. Tang and J. Xu, "Extending Network Lifetime for Precision-Constrained Data Aggregation in Wireless Sensor Networks," *Proc. INFOCOM,* 2006.
[24] X. Tang and J. Xu, "Adaptive Data Collection Strategies for Lifetime-Constrained Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 19, no. 6, pp. 721-734, June 2008.
[25] V.S. Tseng and K.W. Lin, "Energy Efficient Strategies for Object Tracking in Sensor Networks: A Data Mining Approach," *J. Systems and Software,* vol. 80, no. 10 pp. 1678-1698, 2007.
[26] V.S.-M. Tseng, E.H.-C. Lu, and K.W. Lin, "An Energy-Efficient Approach for Real-Time Tracking of Moving Object in Multi-Level Sensor Networks," *Proc. IEEE 11th Int'l Conf.Embedded and Real-Time Computing Systems and Applications (RTCSA),* pp. 305-310, 2005.
[27] P.-J. Wan, S.C.-H. Huang, L. Wang, Z. Wan, and X. Jia, "Minimum-Latency Aggregation Scheduling in Multihop Wireless Networks," *Proc. MobiHoc,* pp. 185-194, 2009.
[28] S.-H. Wu, K.-T. Chuang, C.-M. Chen, and M.-S. Chen, "Toward the Optimal Itinerary-Based KNN Query Processing in Mobile Sensor Networks," *IEEE Trans. Knowledge and Data Eng.,* vol. 20, no. 12, pp. 1655-1668, Dec. 2008.
[29] H.-Y. Yang, C.-H. Lin, and M.-J. Tsai, "Distributed Algorithm for Efficient Construction and Maintenance of Connected k-Hop Dominating Sets in Mobile Ad Hoc Networks," *IEEE Trans. Mobile Computing,* vol. 7, no. 4, pp. 444-457, Apr. 2008.
[30] Y. Yao and J. Gehrke, "The Cougar Approach to In-Network Query Processing in Sensor Networks," *SIGMOD Record,* vol. 31, no. 3, pp. 9-18, 2002.
[31] W. Ye, J.S. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. INFOCOM,* 2002.
[32] W. Ye, J.S. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 12, no. 3, pp. 493-506, June 2004.

Chih-Chieh Hung received the MS degree from National Chiao Tung University (NCTU), Taiwan, in 2005 and the PhD degree from the Department of Computer Science atthe National Chiao Tung University, Taiwan, in 2011. During the PhD program, he was mainly involved in the projects related to trajectory pattern mining and applications, data management on sensor networks, and sensor-enabled cloud computing. Currently, he is a Technical Yahoo! of Global Media Foundation in Yahoo Inc. His research interests include trajectory pattern mining, location-based social network, sensor data management, and data mining. He has already published nine international conference papers and three international journal papers; especially, he received the Best Paper Award in ACM Workshop on Location-Based Social Network in 2009.

Wen-Chih Peng received the BS and MS degrees from the National Chiao Tung University, Taiwan, in 1995 and 1997, respectively, and the PhD degree in electrical engineering from the National Taiwan University, R.O.C., in 2001. Currently, he is working as an associate professor in the Department of Computer Science, National Chiao-Tung University, Taiwan. Prior to joining the Department of Computer Science, National Chiao-Tung University, he was mainly involved in the projects related to mobile computing, data broadcasting, and network data management. His research interests include mobile data management, sensor data management and data mining. He serves as a program committee member for the IEEE International Conference on Data Engineering (ICDE), the International Conference on Mobile Data Management (MDM), and the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD). He is a member of the IEEE.



Wang-Chien Lee received the BS degree from the Information Science Department, National Chiao Tung University, Taiwan, the MS degree from the Computer Science Department, Indiana University, and the PhD degree from the Computer and Information Science Department, the Ohio State University. He is an associate professor of computer science and engineering at the Pennsylvania State University (Penn State). Prior to joining Penn State, he was a principal member of the technical staff at Verizon/GTE Laboratories, Inc. He leads the Pervasive Data Access (PDA) Research Group at Penn State to pursue cross-area research in database systems, pervasive/mobile computing, and networking. He is particularly interested in developing data management techniques (including accessing, indexing, caching, aggregation, dissemination, and query processing) for supporting complex queries in a wide spectrum of networking and mobile environments such as peer-to-peer networks, mobile ad hoc networks, wireless sensor networks, and wireless broadcast systems. Meanwhile, he has worked on XML, security, information integration/retrieval, and object-oriented databases. He has published more than 160 technical papers on these topics. His research has been supported by multiple US National Science Foundation (NSF) grants. Most of his research results were published in prestigious journals and conferences in the fields of databases, mobile computing, and networking. He was active in various IEEE/ACM conferences and has given tutorials for many major conferences. He was the founding program cochair of the International Conference on Mobile Data Management (MDM). He has also served as a guest editor for several journal special issues (e.g., the *IEEE Transactions on Computers*) on mobile database-related topics. He has served as the TPC chair or general chair for a number of conferences, including the Second International Conference on Scalable Information Systems (Infoscale '07), the Sixth International ACM Workshop on Data Engineering for Wireless and Mobile Access (MobiDE '07), and the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC '08). He was the TPC chair for the 10th International Conference on Mobile Data Management (MDM '09). He is a member of the IEEE and the IEEE Computer Society.