國立交通大學

資訊科學與工程研究所

碩 士 論 文

利用資料涵蓋觀念解決無線感測網路中之近似查詢處理

Exploiting Data Coverage for Approximate Query Processing in Wireless Sensor Networks

研 究 生：李志劭

指導教授：彭文志　教授

中 華 民 國 九 十 五 年 六 月

利用資料涵蓋觀念解決無線感測網路中之近似查詢處理
Exploiting Data Coverage for Approximate Query Processing in Wireless Sensor Networks

研 究 生：李志劭	Student：Chih-Shao Lee
指導教授：彭文志	Advisor：Wen-Chih Peng

國 立 交 通 大 學
資 訊 科 學 與 工 程 研 究 所
碩 士 論 文

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer Science

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

摘　　要

在這篇論文當中，我們探討的議題是無線感測網路中的近似查詢處理。由於鄰近感測節點的讀數通常具有空間相關性，將這些讀數相近的感測節點聚集成多個叢集，並替每一個叢集選出一個 r-node(即叢集代表點)來回答查詢能有效地節省網路的電力消耗。為了解決這個議題，我們提出資料涵蓋的觀念。我們首先定義資料涵蓋問題是選擇最少數目的 r-node 來完整涵蓋整個網路。我們證明資料涵蓋問題是一個 NP-complete 的問題。因此，我們提出了兩個誘導式演算法 DCglobal 和 DClocal 來解決此問題。演算法 DCglobal 是一個集中式的演算法。我們證明經由 DCglobal 得到的解和最佳解之間的誤差是有限的。演算法 DCglobal 消耗的電力相當可觀，因此，我們提出演算法 DClocal。DClocal 為一分散式演算法，分別由網路中的各個感測節點執行以選出 r-node。同樣的，我們也證明了經由 DClocal 得到的解和最佳解之間的誤差是有限的。接著我們討論如何決定演算法 DClocal 中所使用的參數的最適值。經由效能評估，我們發現演算法 DClocal 比起 DCglobal 的確消耗了較少的電力，並且經由和其他的演算法進行比較，演算法 DClocal 在網路時間及電力消耗方面的效能都較為優異。因此我們提出的演算法能夠有效地節省電力，並且針對使用者的查詢提供符合使用者所需之精確度的答案。

關鍵字：查詢處理，無線感測網路，資料涵蓋，集合涵蓋。

Abstract

In this paper, we focus on the issue of approximate query processing in wireless sensor networks. Since there are usually spatial correlations between the readings of sensors in vicinity, it is energy-efficient to group these sensors into clusters and select one *r-node*, i.e. cluster-head, for each cluster to answer queries. We propose an innovative concept called *data-coverage* to address the problem. We prove that the data-covering problem which selects minimal number of r-nodes to fully data-cover the whole network is an NP-complete problem by reducing the set-covering problem to our data-covering problem. In order to solve the data-covering problem, we devise two heuristic algorithms *DCglobal* and *DClocal*. The first algorithm, DCglobal, is a centralized algorithm executed at the sink. The ratio between the number of r-nodes selected by DCglobal and that of the optimal solution is bounded. Since the energy consumption of DCglobal is extremely high, we devise another algorithm called DClocal. DClocal is a distributed algorithm executed by each sensor to locally select r-nodes. We prove that the ratio between the number of r-nodes selected by DClocal and that of the optimal solution is bounded. We then discuss the optimal value of the parameter used in DClocal. Through the experimental study, it can be seen that the energy consumption of DClocal is less than that of DCglobal. In addition, through comparing with other algorithms, the performance of DClocal is much better in terms of network lifetime and energy consumption. Thus, we conclude that our algorithms are energy-efficient and can provide the users the answers that satisfying their requirements for precision.

Keywords —Query processing, sensor network, data-coverage, set-covering.

致　　謝

歷經兩年的努力，終於完成了這篇論文，算是為自己的碩士生涯做了總結，將來回首這段日子的時候，也證明我並沒有空手而回。

能夠完成這篇論文，首先要感謝我的指導教授彭文志教授，富有耐心地帶領我這個非本科系出身的學生踏進學術殿堂的大門，在這兩年當中我從老師身上獲得了許多珍貴的研究經驗。同時也感謝兩位口試委員金仲達教授和曾煜棋教授在口試時提供了許多寶貴的意見，讓我能夠將這篇論文作得更完整。

其次要感謝的是博士班的洪智傑學長，在我遇到問題時總是熱心地和我一同討論，在我口試後修改論文這段期間更是提供我許多珍貴的意見，若是沒有他的協助，我無法完成修改論文的工作。

在碩士班的兩年中，實驗室生活可以幾乎和我的生活劃上等號，我很榮幸能和張民憲、楊慧友、以及蕭向彥三位非常優秀的同學在研究的路途結伴同行，我很感激他們從不吝於分享自己的經驗和知識給我，我從他們身上得到的實在太多了。大家一起在實驗室做研究到三更半夜的日子，雖然當時我們都疲憊不堪，現在想來反而是相當有趣的經驗。

另外還要感謝林裕欽、胡星垣、柯郁任、張修維、以及蕭志鵬學長等人，在碩一時帶領我熟習實驗室的環境，讓我能夠很快融入實驗室的生活。在碩二時認識的李柏逸、游敦皓、周佳欣、黃正和等學弟妹，讓我在忙碌的碩二生活可以偶爾忘卻論文的壓力一同歡笑。和大家在實驗室一起打球、吃飯的生活絕對是我碩士生涯最好的回憶。

要特別感謝的是我大學就讀的清華大學計量財務金融學系的黃裕烈教授，承蒙老師在許多時候的照顧，雖然彼此的研究領域並不相同，老師還是在研究的心態上給了我許多相當寶貴的意見，我非常珍惜也相當感激和老師這段亦師亦友的關係。

當然，我最感謝的還是我的父母，無論是精神上或是物質上，他們都是我最大的支柱，很感謝他們能夠讓我在人生至今的求學生涯當中都能無後顧之憂的學習。還有我即將就讀碩士班的弟弟，祝福他在這兩年的碩士生涯能夠一切順利。

最後，感謝我所有的同學和朋友，多虧你們的支持，我才能夠順利的完成這篇論文。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Sensor networks are composed of a large number of sensor nodes, which are deployed randomly or in a planned way. Since the deployed areas are usually unreachable, sensors must self-organize and work cooperatively. Because of the cheapness and self-organizing capability, sensor networks provide a cost-efficient way to monitor large-area phenomena that may be impossible to do before. The monitoring applications include inventory management, habitat monitoring [12], and battlefield awareness. The energy constraint on sensors, however, is an important issue of monitoring applications since the power supply of sensors is limited and irreplaceable. Therefore, the designers should take account of the low power consumption requirement while designing applications.

Since the sensor network can be viewed as a distributed database [3], several previous works proposed declarative SQL-like query which enable users to acquire the information about the network through issuing queries to the sink [10][17]. Table 1.1 shows a query example which gets the average temperature in the monitored region. To compute the answer of a query, the sink floods the query to every sensor in the network, and the sensors which satisfy the criterion transmit their sensing readings to the sink for computing the answer. Obviously, this straight approach gives the user the exact answer of the query. However, one main drawback is that, because of the huge amount of messages transmitted in the flooding, the energy cost

SELECT	AVG(S.Temperature)
FROM	Sensor S
WHERE	S.Location IN region

Table 1.1: A query example.

is considerable.

In fact, it is impossible to acquire all the relevant data of the sensor network environment. The physically observable world consists of a set of continuous phenomena in both time and space, so the set of relevant data is in principle infinite. However, sensors can only collect data by sampling the environment at discrete points in time and space. Therefore, once we issue a query about the environment, we can get only approximate answers about the real phenomena from the network. There are many previous works dealing with the approximate query processing problem of sensor networks [6] [9].

In this paper, we deal with the approximate query processing in wireless sensor networks. We propose to cluster sensors with similar readings and select a *representative node* (referred to as *r-node*) for each cluster. Since the sensors are usually densely deployed in the environment, the readings of sensors in vicinity are highly correlated. Transmitting similar readings of sensors in vicinity to the sink incurs redundant energy consumption. To figure out this issue, we propose a new concept, called *data-coverage*. A sensor is said to be able to *cover* another sensor if their readings are similar. Each sensor in the networks has its own data-coverage containing neighbors with similar readings. The r-node will *represent* neighbors in its data-coverage to answer queries. Unlike traditional coverage problem in sensor networks which focuses on full coverage on physical area, the objective of data-coverage is to select a set of r-nodes to fully cover the network in terms of data similarity, i.e., every sensor in the network is in at least one data-coverage of the r-nodes.

The concept of data-coverage can be best understood by the illustrative example in Figure 1.1, where the data-coverages of the 12 sensors is shown in Table 1.2. Note that there are

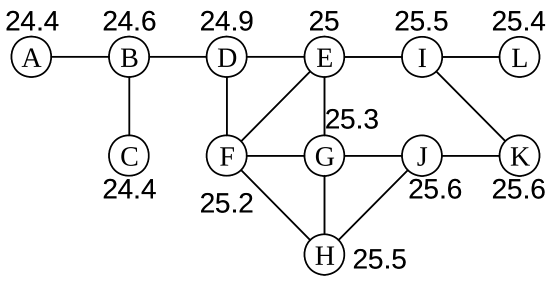


Figure 1.1: An illustrative example of representative selection.

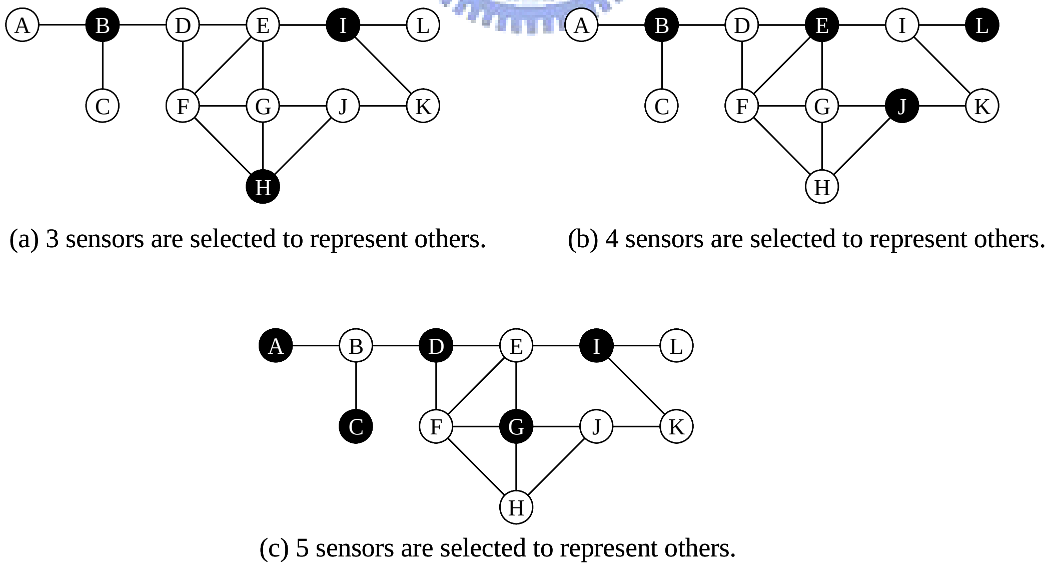

A
B
D
E
I
L
C
F
G
J
K
H
(a) 3 sensors are selected to represent others.
(b) 4 sensors are selected to represent others.
(c) 5 sensors are selected to represent others.

Figure 1.2: There are several combinations of representative sensors.

ID	Coverage Range
A	{A,B,C,D}
B	{A,B,C,D,E}
C	{A,B,C,D}
D	{A,B,C,D,E,F,G}
E	{B,D,E,F,G,H,I,L}
F	{D,E,F,G,H,I,J,K,L}
G	{D,E,F,G,H,I,J,K,L}
H	{E,F,G,H,I,J,K,L}
I	{E,F,G,H,I,J,K,L}
J	{F,G,H,I,J,K,L}
K	{F,G,H,I,J,K,L}
L	{E,F,G,H,I,J,K,L}

Table 1.2: The data-coverages of the sensors in the example.

several combinations to fully cover the whole sensor network. For example, 3 possible selections of r-nodes are shown in Figure 1.2(a), 1.2(b), and 1.2(c). As can be seen in this example, the number of r-nodes differs with different selection, and thus the energy consumed in answering queries also differs. In order to conserve energy consumption, we want to select as less number of r-nodes as possible so as to cover the whole sensor network. The minimal number of r-nodes of the above example is 3, as can be seen in 1.2(a).

Consequently, we explore in this paper the issue of selecting minimal number of r-nodes to approximately answer user queries while guaranteeing that the approximation error is within the user-tolerable error threshold. Specifically, we define this as a *data-covering problem*. Given the pre-specified error threshold and a set of sensor nodes, we are going to select minimal number of sensors as r-nodes such that the union of their data-coverages fully data-covers the network. Moreover, the difference between the readings of a r-node and the sensors in its data-coverage will not exceed the error threshold. We prove that the data-covering problem is NP-Compelete in later section by reducing the set-covering problem to the data-covering problem. Then, we propose two approximate algorithms to solve the data-covering problem. The first one, *DCglobal* (standing for algorithm of *D*ata-*C*overage problem with *global* information), which is a centralized algorithm executed at the sink, has bounded approximation ratio but higher energy consumption; whereas the second one, *DClocal* (standing for algorithm of *D*ata-*C*overage problem with *local* information), which is a distributed algorithm executed at each sensor, sacrifices the theoretical bound but is more energy-efficient. The sensors select r-nodes cooperatively according to the selection criteria in DClocal.

Performance of these algorithms is comparatively analyzed and sensitivity analysis on several parameters, including the number of hops for sensors to exchange readings in DClocal, the error threshold, and the degree of spatial correlation is conducted. Simulation results show that by exploiting the feature of data-cover, the lifetime of wireless sensor networks is extended. Furthermore, due to the distributed nature of DClocal, DClocal is able to further reduce the number of messages.

A significant amount of researches of query processing in sensor network has been done. We mention in passing that the authors in [6] explored a model-driven architecture in which a centralized probabilistic model is used to estimate the readings of sensors by generating an observation plan to collect appropriate readings of sensors. In [9], the author proposed an extension of declarative query in sensor networks, called snapshot queries. The snapshot queries can be answered through a data-driven approach by using a linear regression model to predict the readings of 1-hop neighbors. Despite of applying prediction techniques for approximate query processing, another class of query processing uses in-network data aggregation approaches to reduce message transmissions [18]. In [10], a declarative SQL-like query is supported and the data is aggregated in a collection tree. Similarly, the authors in [17] proposed an architecture to support declarative queries by generating a query plan which will be spread to the network for in-network query processing.

Since the spatial correlation between sensors is not restricted in 1-hop neighborhood, we propose to selected sensors to answer queries for neighbors with similar readings multi-hops away, rather than only 1-hop neighbors in [9]. Although the range of spatial correlation is usually large, dissimilarity between two distant sensors still exists. Therefore, instead of using only one distribution to capture the readings in the network in [6], we propose to select several sensors to capture the different distributions of readings in the network environment. The contributions of this paper are organized as follows.

- In order to take advantage of the spatial correlation in sensing readings, we propose an innovative concept, called *data-coverage*, to address the problem of approximate query processing. We define that the problem of selecting minimal number of r-nodes to fully data-cover the network is a data-covering problem.

- We prove that the data-covering problem is NP-complete. We prove it by reducing the set-covering problem to the data-covering problem.

- In light of this concept, we develop two approximate algorithms DCglobal and DClocal to solve the data-covering problem. DCglobal is a centralized algorithm executed at the sink while DClocal is a distributed algorithm. By using only r-nodes to answer queries, the number of sensors participated in queries is greatly reduced. Thus the network lifetime is prolonged. If there are sensors which are temporarily inaccessible, the r-nodes can answer queries for them.

- We conduct a theoretic analysis for users to decide the optimal value of the parameter used in DClocal.

- We devise an energy efficient maintenance mechanism which rotates the responsibility of r-node to every sensor. Through this mechanism, the energy consumption of each sensor is balanced.

The rest of this paper is organized as follows. Preliminaries are given in Section 2. In Section 3, we develop two heuristic algorithms DCglobal and DClocal to solve the data-covering problem. Performance studies are conducted in Section 4. This paper concludes with Section 5.

Chapter 2

Preliminaries

In this section, we first introduce the terminologies of algorithm DCglobal and DClocal in Section 2.1. We also give a formal definition of the data-covering problem and prove that it is NP-Complete by reducing from the set-covering problem in this subsection.

2.1 Problem Definition

In this study, we focus on the issue of approximate query processing in sensor networks. We observed that the readings of sensors in vicinity are highly correlated. To take advantage of the spatial correlation, we propose to cluster sensors with similar readings together and select a sensor as an r-node for each cluster.

Table 2.1 shows the description of symbols used in our algorithms. The set of sensors is denoted as S, and R denotes the set of r-nodes. Usually, the number of r-nodes is less than that of sensors. Therefore, we have $|R| \leq |S|$. The data-coverage (or cluster) of a sensor i is denoted as C_i. For a sensor $j \in C_i$, $R(j) = i$ if sensor i is selected as an r-node. Since the selected r-nodes must fully data-cover the whole network, we have $\bigcup_{r \in R} C_r = S$. Let v_{it} denote the reading of a sensor i sampled at time t. For each sensor, the current reading is correlated with the successive one. In order to utilize such temporal correlation, we compare

Description	**Symbol**
The set of sensors	S
The set of r-nodes	R
The data-coverage (or cluster) of r-node r	C_r
R-node of sensor i	$R(i)$
The reading of sensor i sampled at time t	v_{it}
The reading vector of sensor i	$\overrightarrow{v_i}$
The pre-specified error threshold	ϵ
Distance between sensor i and j	$d\left(i,j\right)$

Table 2.1: Discription of symbols.

the similarity between two sensors based on their readings sampled in a time period. The readings of sensor i are stored in a reading vector $\overrightarrow{v_i}$. The dissimilarity between two sensors is denoted as $d(i,j)$. We give a mathematical definition of $d(i,j)$ in the following.

Definition 1: Suppose that the length of reading vectors is l. The distance between sensor i and j is the Euclidean distance between their reading vectors and can be defined as:

$$d\left(i,j\right) = \sqrt{\sum_{k=1}^{l} |v_{ik} - v_{jk}|^2}.$$

From the above definition, sensor i can compute its data-coverage C_i if it has the readings of its neighbors. The data-coverage of sensor i consists the neighbors which are *data-covered* by i. We give a formal definition of data-cover in the following.

Definition 2: Sensor j is data-covered by sensor i if (1) i and j are connected by at least a path, (2) supposed that the path connecting i and j is $< s_0 = i, s_1, ..., s_k = j >$, sensor i can data-cover s_t for $1 \leq t \leq k-1$, and (3) $d\left(i,j\right) \leq \epsilon$, the dissimilarity between i and j does not exceed the error threshold.

Thus, the problem that we study in this paper can be formally defined as follows.

Data-covering problem: Given a set of sensors, the collection of the data-coverages of all the sensors, and a pre-specified error threshold ϵ, the data-covering problem is to select a set of r-nodes R, such that for each sensor i, there is at least an r-node $r \in R$ data-covering

i, and the number of r-nodes is minimized.

By reducing the *set-covering problem* to the data-covering problem, we prove that the data-covering problem is NP-complete. The set-covering problem is briefly described as follows:

Set-covering problem: Given the set of elements X, a family F of subsets of X, and every element of X belongs to at least one subset in F, i.e. $\bigcup_{S \in F} S = X$. The problem of selecting a minimize subset $C \subseteq F$ whose members cover all of X, i.e. $\bigcup_{S \in C} S = X$, is NP-complete [5].

Theorem 1: The data-covering problem is NP-complete.

Proof: Given an instance $I(X, F)$ of set-covering problem, we can map it into an instance I' of data-covering problem. Define a sensor network with the members in X as sensors. For each subset S in F, define S as the data-coverage of any one sensor corresponding to the member covered by S. Noted that no two subsets in F will be assign to the same sensor. For sensors without any data-coverage assigned, define that its data-coverage covers only itself. From the above mapping, we can clearly see that if I' has a solution, I will have a solution. Further, if a solution for data-covering problem exists, this can be verified in polynomial time. $\square$

Since the data-covering problem is NP-complete, we therefore propose two heuristic algorithms so as to select the minimal number of r-nodes that are able to data-cover the whole sensor networks.

Chapter 3

Discovering Representative Nodes

In this section, we propose two heuristic algorithms for the data-covering problem. Specifically, a centralized algorithm DCglobal is developed in Section 3.1. In Section 3.2, we devise a distributed algorithm DClocal. Section 4.3 is devoted to the comparison of DCglobal and DClocal.

3.1 Algorithm DCglobal

Usually, there are a large number of sensors in the network. It is undesirable to enumerate all the possible combinations to find the optimal set of r-nodes to fully data-cover the whole network. Fortunately, since the data-covering problem is equivalent to the set-covering problem, several heuristic solutions are proposed for the set-covering problem and can be applied to solve the data-covering problem. The most well-known one is the greedy algorithm which iteratively finds the current largest set [5]. The algorithm executed at the sink (referred to as *DCglobal*) for the data-covering problem is shown below.

Algorithm DCglobal first will collect the coverage information (i.e. collect the readings of each sensor and compute the data-coverage of each sensor), and then according to the coverage information determine the set of r-nodes.

Algorithm 1 DCglobal: Select Representative Nodes by Greedy Set-Covering

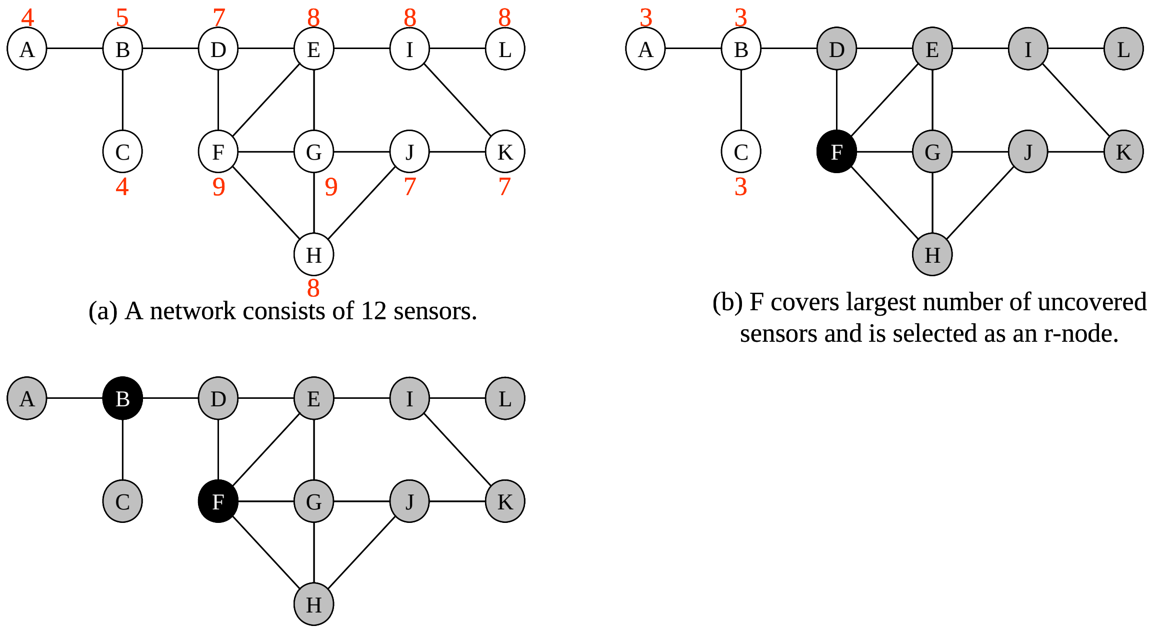
```
Input: $S$, set of all the sensors with their readings; $\epsilon$, the error threshold.
Output: $R$, set of r-nodes where $R \subseteq S$.
 1: for each sensor $i \in S$ do
 2:   for each neighbor $j$ do
 3:     if $d(s_i, s_j) \leq \epsilon$ then
 4:       $i$ adds $j$ to its data-coverage $c_i$.
 5: while $S \neq \phi$ do
 6:   Find a sensor $i$ whose data-coverage $c_i$ maximizes $|c_i|$.
 7:   for each sensor $j \in c_i$ do
 8:     Remove $j$ from $S$ and the remaining data-coverages.
 9:   Add $i$ into $R$.
10: Return $R$
```

From line 1 to line 4 of algorithm DCglobal, the sink collects readings from all the sensors and computes the data-coverage of each sensor according to the error threshold ϵ. When all data-coverage information of sensors are calculated, DCglobal will begin to select the r-nodes in accordance with the coverage information. From line 5 to line 10, algorithm DCglobal selects a sensor whose data-coverage covers the largest number of uncovered sensors as an r-node. As can be seen in line 7 and 8, the sensors covered by the newly selected r-node are removed from the data-coverages of remaining sensors. In line 9, the sink adds the selected r-node into the set of r-nodes R. After the execution of DCglobal, the sink broadcasts the information of R to each sensor in the network. Later on, the sensors belonging to R will answer queries for sensors in their data-coverages. The other sensors only sense the environment and report their readings to their r-nodes periodically. Basically, we borrow the concept of the existing algorithm in [5] for algorithm DCglobal. We can proved that the ratio between the number of r-nodes selected by DCglobal and that of the optimal solution is bounded.

Theorem 2: Given a collection of data-coverages of sensors C, let the set of r-nodes selected by DCglobal be C_g, the set of optimal (i.e., minimal number) r-nodes be C^*, then $|C_g| \leq |C^*| \times H\left(\max_{S \in C} |S|\right)$, where $H(n) = \sum_{k=1}^{n} \frac{1}{k}$.

Proof: See [5].

$\square$



(a) A network consists of 12 sensors.

(b) F covers largest number of uncovered sensors and is selected as an r-node.

(c) B covers largest number of uncovered sensors and is selected as an r-node.

Figure 3.1: An illustrative example for DCglobal algorithm.

An illustrative example for algorithm DCglobal is shown in Figure 3.1, where the number on the top of a sensor is its data-coverage size. The reading and data-coverage content of each sensor is shown in Table 3.1. In the first iteration, sensor F is selected as an r-node since it has the largest data-coverage size. The sensors covered by F are removed from the data-coverages of sensor A, B, and D. In the second iteration, sensor B is selected as an r-node since it has the largest data-coverage size among all the remaining uncovered sensors. Since all the sensors are covered after the second iteration, the execution of algorithm DCglobal terminates. The greedy solution of this example is {F, B}.

The computation complexity for computing the data-coverages of sensors is $O(n^2)$ where n denotes the number of sensors. The centralized method requires sensors periodically transmitting their readings to the sink. The DCglobal works well only when the number of sensors n is small. However, since the number of sensors in a sensor network is usually very large, the method does not scale well. Additionally, the centralized method can not make quick response when the distribution of readings changes. Therefore, it is necessary to develop an

ID	Reading	Coverage Range
A	24.4	{A,B,C,D}
B	24.6	{A,B,C,D,E}
C	24.4	{A,B,C,D}
D	24.9	{A,B,C,D,E,F,G}
E	25	{B,D,E,F,G,H,I,L}
F	25.2	{D,E,F,G,H,I,J,K,L}
G	25.3	{D,E,F,G,H,I,J,K,L}
H	25.5	{E,F,G,H,I,J,K,L}
I	25.5	{E,F,G,H,I,J,K,L}
J	25.6	{F,G,H,I,J,K,L}
K	25.6	{F,G,H,I,J,K,L}
L	25.4	{E,F,G,H,I,J,K,L}

Table 3.1: The readings and data-coverage of sensors in the example.

alternative method which selects r-nodes while still preserve the approximation quality.

3.2 Algorithm DClocal

In this subsection we develop a distributed algorithm DClocal for the data-covering problem. In Section 3.2.1, algorithm DClocal is presented. In Section 3.2.2, we further analyze parameters used in algorithm DClocal.

3.2.1 Design of Algorithm DClocal

Collecting the complete data-coverage information is necessary for DCglobal, since at each step DCglobal needs to find a sensor with globally maximal coverage size. In a distributed environment, however, collecting all data-coverage information is costly. Moreover, due to the spatial correlation property in sensor networks, the readings of a sensor are more similar to the readings of nearby sensors rather than the readings of further sensors. Thus, the requirement for global coverage size comparison has to be relaxed. Our objective is to select r-nodes based on local information of sensors, without knowing the global data-coverage information. To avoid globally comparing coverage size, in algorithm DClocal, sensors compare coverage size locally. The concept of DClocal is that a sensor whose data-coverage is larger than its neighbors' data-coverages has to become an r-node. Through this relaxation, we expected that multiple sensors may be selected as r-nodes in a single round. DClocal is executed iteratively until every sensor is data-covered. In order to balance energy consumption, sensors with energy capacity below a threshold value will not participate in the execution of algorithm DClocal.

Before executing algorithm DClocal, each sensor has to collect the readings of sensors in vicinity to compute its data-coverage. However, the message cost is considerable if each sensor collects the readings of all the other sensors in the network. Thus, in algorithm DClocal we restrict that each sensor exchanges readings only with neighbors within k-hops distance since the readings are usually locally correlated. The value of k actually depends on the type of sensing data and application. We will discuss the issue of selecting optimal value of k in next subsection.

At the beginning of each round, every uncovered sensor exchanges its data-coverage size with neighbors for comparison. Sensors with largest data-coverage among their neighbors within k-hops distance become r-nodes. These r-nodes inform sensors in their data-coverages to join their clusters. This process iterates until every sensor is covered by at least one r-node.

Since the readings of nearby sensors are highly correlated, the data-coverages of nearby sensors usually are overlapped. When a sensor i is selected as an r-node, the neighbors of i whose data-coverages overlap i's data-coverage have to update. When a sensor i becomes an r-node, i broadcasts join-messages containing the IDs of uncovered sensors in its data-coverage. The sensor whose ID appearing in the join-message joins i's cluster and informs its neighbors to adjust their data-coverages by sending inform-messages containing the information of the join-messages. The sensor which receives an inform-message adjusts its data-coverage by eliminating the sensors whose IDs appearing in the inform-message from its data-coverage. After all uncovered sensors adjusting their data-coverages, another round of selection starts.

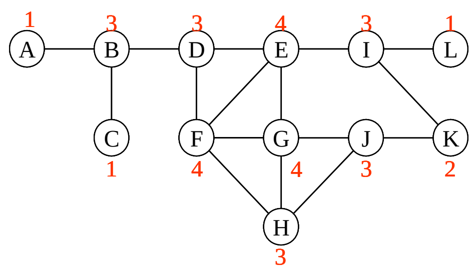
Since DClocal is executed in a distributed way, each sensor has to wait until it has received all the inform-messages. The length of the waiting time interval is set to be equal to the time needed for an inform-message to traverse k hops. Algorithm DClocal is shown below.

Algorithm 2 DClocal: Select Representative Nodes by Local Set-covering

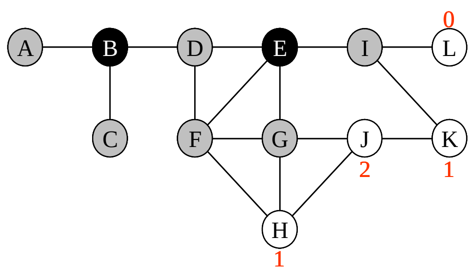
```
Input: $\epsilon$, the error threshold.
Output: $R$, the set of r-nodes.
 1: repeat
 2:   for each unclustered sensor $i$ do
 3:     Exchanges coverage size and content with unclustered neighbor $j$ in $k$-hops neighbor-
        hood.
 4:     if $i$ has the largest data-coverage size among its neighbors then
 5:       $i$ becomes r-node and joins $R$
 6:       $i$ broadcasts join messages to sensors within its data-coverage.
 7:       for each sensor $j$ which is unclustered and is in the data-coverage of $i$ do
 8:         $j$ joins the cluster of $i$.
 9:         $j$ forwards the information in the join message to its neighbors.
10:     else
11:       while $i$ is unclustered and receives inform messages from neighbors do
12:         $i$ deletes the IDs in the inform message from its data-coverage $c_i$.
13: until all the sensors are clustered.
```

Algorithm DClocal runs iteratively until every sensor in the network is clustered. Line 3 requires every sensor to exchange its data-coverage size with neighbors in k-hops neighborhood. After comparing the data-coverage size, if a sensor i whose data-coverage size is the largest in its k-hop neighborhood, i decides to become an r-node, as shown in line 5. In addition, in line 6, i has to broadcast join-messages to sensors in its data-coverage. The join-message contains the IDs of uncovered sensors in i's data-coverage. For a sensor which has not become an r-node, if it receives an inform-message, it deletes the sensors whose IDs appearing in the inform-message from its data-coverage. If there are still some sensors uncovered in the end this round, the remaining uncovered sensors again exchange their updated data-coverage size and select new r-nodes.

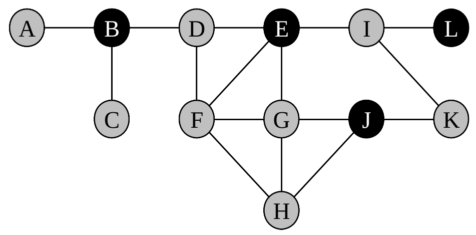
Consider the illustrative example in Figure 3.2(a), where $k = 1$. After each step, the updated coverage size of each sensor is shown in red number. In the first round, both sensor B and E have the largest coverage size among their 1-hop neighbors. Thus, B and E decide



(a) A network consists of 12 sensors.



(b) At first round, both sensor B and E are selected as r-nodes.



(c) At second round, both sensor J and L are selected as r-nodes. Since every sensor is clustered, the selection process stops.

Figure 3.2: An illustrative example for algorithm DClocal where $k = 1$.

to become r-nodes. Sensor A and C join the cluster of B while sensor D, F, G, and I join the cluster of E, as be seen in Figure 3.2(b). Since sensor E has transmitted its data-coverage content to sensors in its cluster, sensor G and I can recognize that there are neighbors may not be covered by any sensor yet. For example, the data-coverages of sensor E and G are {D, E, F, G, I} and {E, F, G, H, J}. By computing the set difference {E, F, G, H, J}$\setminus${D, E, F, G, I}={H, J}, sensor G knows that its neighbor H and J are still uncovered. Thus, sensor G informs H and J by transmitting inform-messages containing the data-coverage content of E to them. Sensor H and J update their data-coverage according to the inform-message, respectively. Similarly, L and K update their data-coverage after receiving the inform-message of I. In the second round, after exchanging data-coverage size, sensor J and L become r-nodes. The execution of algorithm DClocal terminates at this round since all the sensors are covered. The result is shown in Figure 3.2(c).

After our approximate algorithm DClocal, we are going to analyze the approximation ratio of DClocal. The following theorem shows a bound of the local method.

Theorem 3: Given a collection of data-coverages of sensors C, let the set of r-nodes selected by DClocal be C_l, the set of optimal (i.e., minimal number) r-nodes be C^*, then $|C_l| \leq |C^*| \times \left(\frac{2^k-1}{2^{k+1}} \times \max_{S \in C^*} |S|\right)$, where k is the number of hops for sensors exchanging readings in DClocal.

Proof: For the simplicity of presentation, we prove the theorem in the set-covering framework. Similar to the approach in [5], we assign a cost of 1 to each data-coverage of the r-nodes selected by DClocal, and distribute the cost evenly over the sensors covered for the first time. Recall that R represents the collection of r-nodes. Let r_{ij} denote the jth r-node selected in the ith round of DClocal, and $r_i = (r_{i1}, r_{i2}, ...)$ denote the collection of r-nodes selected by DClocal in the ith round. We use C_{r_i} to represent the set of sensors covered by the r-nodes in r_i.

If sensor s is covered for the first time by an r-node r_{ij}, the cost assigned to s is denoted as c_s. From our definition of c_s, we know that $c_s = \frac{1}{\left|C_{r_{ij}} - \left(C_{r_1} \cup C_{r_2} \cup ... \cup C_{r_{i-1}}\right)\right|}$, and $\sum c_s = |R|$. Let C^* denote the optimal solution, the cost assigned to the C^* is $\sum_{S \in C^*} \sum_{s \in S} c_s$. Moreover, since each sensor is in at least one data-coverage of the r-nodes in the optimal solution, we have

$$|R| = \sum c_s \leq \sum_{S \in C^*} \sum_{s \in S} c_s = \sum_{S \in C^*} K_S.$$

Where $K_S = \sum_{s \in S} c_s = \sum_{r_{ij} \in R} \frac{\left|R\left(C_{r_{ij}}\right) \cap S\right|}{\left|R\left(C_{r_{ij}}\right)\right|}$ and $R\left(C_{r_{ij}}\right) = C_{r_{ij}} - \left(C_{r_1} \cup C_{r_2} \cup ... \cup C_{r_{i-1}}\right)$.

Suppose that the coverage S_i in C^* overlaps the data-coverages of r-nodes $r_1^i, r_2^i, ..., r_p^i$, from the above definition of K_S, we have $K_{S_i} = \sum_{t=1}^{p} \frac{\left|R\left(C_{r_t^i}\right) \cap S_i\right|}{\left|R\left(C_{r_t^i}\right)\right|} \leq p$, since $\left|R\left(C_{r_t^i}\right) \cap S_i\right| \leq \left|R\left(C_{r_t^i}\right)\right|$. Since the number of clustered sensors is at least as much as the number of r-nodes, we have $\left|R\left(C_{r_1^i}\right)\right| + \left|R\left(C_{r_2^i}\right)\right| + ... + \left|R\left(C_{r_p^i}\right)\right| \geq p$. After each round, one of the two similar sensors is clustered or selected as an r-node, we have

$$\begin{aligned}
& \left|R\left(C_{r_1^i}\right)\right| + \left|R\left(C_{r_2^i}\right)\right| + \ldots + \left|R\left(C_{r_k^i}\right)\right| \\
\geq & \frac{|S_i|}{2} + \frac{|S_i|}{4} + \ldots + \frac{|S_i|}{2^k} \\
= & \left(1 - \frac{1}{2^k}\right)\frac{|S_i|}{2} \\
= & \frac{2^k - 1}{2^{k+1}}\left|S_i\right|.
\end{aligned}$$

From the above derivation, we have $K_{S_i} \leq p \leq \frac{2^k-1}{2^{k+1}}|S_i|$, thus the approximation ratio of DClocal is as follows:

$$\begin{aligned}
|C_l| &\leq |C^*| \times \left(\max_{S\in C^*} K_S\right) \\
&\leq |C^*| \times \left(\frac{2^k - 1}{2^{k+1}} \times \max_{S\in C^*} |S|\right).
\end{aligned}$$

□

3.2.2 The Optimal Number of Hops for Collecting Local Information

In algorithm DClocal, each sensor broadcasts its synopsis to all its neighbors within k-hops distance. It consumes more energy in broadcasting synopsis for each sensor when k is larger. However, for larger k, the sensors may be able to cover more neighbors, and the number of r-nodes needed to cover all the sensors is fewer. The energy consumption is saved by requiring fewer r-nodes to answer queries. Therefore, how to decide the optimal value of k is an important issue since the network lifetime is greatly affected by k.

In this subsection, we build a theoretical model for users to decide the optimal value for the parameter k. The value of k decided by this model can be used to analyze the performance of algorithm DCglobal and DClocal. We apply the idea of *marginal benefit* and *marginal cost* in economics to model the extra energy saving and consumption by increasing the value of k. To decide the optimal value of k, our idea is to increase the value of k until the marginal cost exceeds the marginal benefit.

To facilitate the following discussion, we make some assumptions: (1) the number of sensors follows a *spatial Poisson process* with parameter λ, and (2) the difference between the readings of two adjacent sensors is an independent identically-distributed (i.i.d.) Gaussian random variable with mean 0 and variance σ^2. Note that the value of σ differs upon different environments.

When the value of k is increased from t to $t+1$, the additional cost for the network is the additional number of message transmissions for each sensor to collect the synopses of its $(t+1)$-hops neighbors. The message cost for a sensor to collect the synopsis of each $(t+1)$-hops neighbor is $t+1$. We define that the additional message transmission resulted from increasing the value of k is the marginal cost. We give a formal definition of the marginal cost in the following.

Definition 5: Let n_t denote the average number of t-hops neighbors for each sensor, and MC_{t+1} denote the marginal cost of increasing the value of k from t to $t+1$, then

$$MC_{t+1} = n_{t+1} \times (t+1) \tag{3.1}$$

In the definition of spatial Poisson process, the number of sensors in an area A is λA. Let r denote the maximal transmission range of a sensor, therefore, the maximal distance for a sensor and its t-hops neighbor is tr. We derive that the number of neighbors within t-hops distance is $\lambda \times \pi \left(tr\right)^2$. Thus, the average number of $(t+1)$-hops neighbors is

$$\begin{aligned} n_{t+1} &= \lambda \times \pi \left(t+1\right)^2 r^2 - \lambda \times \pi t^2 r^2 \\ &= \left(2t+1\right) \lambda \pi r^2. \end{aligned}$$

From the above derivation, we can rewrite 3.1 as

$$\begin{aligned} MC_{t+1} &= (2t+1)\,\lambda\pi r^2 \times (t+1) \\ &= \left(2t^2+3t+1\right)\lambda\pi r^2. \end{aligned}$$

After defining the marginal cost, we are going to define the marginal benefit of increasing the value of k. When the value of k is increased from t to $t+1$, the additional benefit for the network is the future saving of message transmissions in answering queries. The formal definition of marginal benefit is shown below.

Definition 6: Let q denote the average number of queries, $\widehat{n}_t$ denote the expected number of t-hops neighbors which can be data-covered by each sensor, $dist_{\text{sink}}$ denote the average distance between a sensor and sink, and MB_{t+1} denote the marginal benefit of increasing the value of k from t to $t+1$, then

$$MB_{t+1} = q \times \widehat{n}_{t+1} \times dist_{\text{sink}}.$$

To compute $\widehat{n}_{t+1}$, we must have the probability density function of the dissimilarity between a sensor i and its t-hops neighbor j. Since we have assumed that the difference between the readings of two adjacent sensors is an i.i.d. Gaussian random variable with mean 0 and variance σ^2, we can derive the distribution of the dissimilarity between i and j. Suppose that there is a path $< s_0 = i, s_1, s_2, ..., s_{t-1}, s_t = j >$ connecting i and j, and x_{uv} is the random variable which is the difference between the readings of s_u and s_v. The difference between the readings of i and j can be written as follows:

$$x_{0t} = x_{01} + x_{12} + ... + x_{(t-1)t}.$$

Let $P\left(x_{0t}\right)$ denote the probability density function of the dissimilarity between a sensor i

and its t-hops neighbor j. Recall that ϵ represents the pre-specified error threshold, we have $\widehat{n}_{t+1} = n_{t+1} \times P\left(x_{0t} \leq \epsilon\right)$.

According to the property of Gaussian distribution, $P\left(x_{0t}\right)$ also follows a Gaussian distribution with mean 0 and variance $t\sigma^2$. Thus, we have the distribution of the difference between the readings of a sensor i and its t-hops neighbor j. Recall that we have defined the sufficient condition for sensor i to data-cover j by the inequality shown below:

$$\sqrt{\left(v_{i1} - v_{j1}\right)^2 + \left(v_{i2} - v_{j2}\right)^2 + ... + \left(v_{il} - v_{jl}\right)^2} \leq \epsilon. \tag{3.2}$$

If we can derive the probability density function of the dissimilarity between sensor i and j, we can get the probability that sensor i can data-cover j.

Let $x_{ij_m} = v_{im} - v_{jm}$, i.e., the difference between the mth readings of sensor i and j for $1 \leq m \leq l$. Therefore, (3.2) becomes:

$$x_{ij_1}^2 + x_{ij_2}^2 + ... + x_{ij_l}^2 \leq \epsilon^2. \tag{3.3}$$

By normalizing the random variable x_{ij_m}, we define a new standard Gaussian random variable $y_{ij_m} = \frac{x_{ijm}}{\sqrt{t}\sigma}$ with mean 0 and variance 1. Thus, we can rewrite (3.3) as follows:

$$y_{ij_1}^2 + y_{ij_2}^2 + ... + y_{ij_l}^2 \leq \frac{\epsilon^2}{t\sigma^2}.$$

To facilitate our discussion, we define a new random variable $Y = y_{ij_1}^2 + y_{ij_2}^2 + ... + y_{ij_l}^2$. According to the property of standard Gaussian distribution, the distribution of Y is a chi-square distribution with degree of freedom l. Thus, $P\left(x_{0t} \leq \epsilon\right)$ is equal to $P\left(Y \leq \frac{\epsilon^2}{t\sigma^2}\right)$ whose definition is shown below:

$$P\left(Y \leq \frac{\epsilon^2}{t\sigma^2}\right) = F\left(\frac{\epsilon^2}{t\sigma^2}, l\right) = \frac{\gamma\left(\frac{l}{2}, \frac{\epsilon^2}{2t\sigma^2}\right)}{\Gamma\left(\frac{l}{2}\right)}, \tag{3.4}$$

where Γ denotes the Gamma function and γ is the incomplete Gamma function.

Thus, the probability that a sensor can data-cover its t-hops neighbors can be evaluated through (3.4). We get the expected number of t-hops neighbors that a sensor can data-cover by multiplying the probability and the average number of t-hops neighbors. Moreover, we can computed the marginal benefit of increasing k from $t-1$ to t. We can then compute the marginal benefit according to the definition described above.

From the above discussion, the theoretically optimal value of k for their applications can be decided by the marginal benefit and marginal cost.

3.3 Comparison of DCglobal and DClocal

In this subsection, we first analyze the energy consumption of DCglobal and DClocal, respectively. The energy efficiency is important since the energy of sensors is limited. We compare the average energy consumption of each sensor during the execution of DCglobal and DClocal in terms of number of message transmission because the energy consumption is dominated by message transmissions.

Suppose that there are n sensors in the network. Recall that in DCglobal, every sensor has to transmit their readings to the sink. Let $dist\,(i, sink)$ denote the number of hops between sensor i and the sink. Since the cost for a sensor to send its information to external storage is $O\left(\sqrt{n}\right)$ [14], the energy consumption for a sensor to send its readings to the sink is $O\left(\sqrt{n}\right)$.

In DClocal, each sensor has to exchange its readings and coverage size with neighbors within k-hops distance. We also assume that the number of sensors follows spatial Poisson distribution with parameter λ. To broadcast a message to all 1-hop neighbors, a sensor i has to broadcast once. To broadcast a message to all 2-hops neighbors, all i's 1-hop neighbors have to broadcast once, and the energy consumption is n_1 where n_t is the number of t-hops neighbors of sensor i. Similarly, the energy consumption for broadcasting messages to i's 2-hops neighbors is n_2. Thus, we conclude that the energy consumption for a sensor i to broadcast messages to all its neighbors within k-hops distance is $1 + n_1 + n_2 + ... + n_{k-1}$. Since in spatial Poisson distribution, the number of neighbors within k-hops distance of a sensor is $\lambda k^2 \times \pi r^2$, where r denotes the maximal transmission range of a sensor, $n_1 + n_2 + ... + n_{k-1} = \lambda (k-1)^2 \pi r^2$. Since every uncovered sensor has to compare the coverage size with at least one neighbor, at least half number of uncovered sensors are clustered after each round of DClocal. The number of rounds executed is bounded by $\log_2 n$ and the average number of message transmissions can be written as

$$\lambda (k-1)^2 \pi r^2 \times \sum_{i=0}^{\log_2 n} \frac{1}{2^i} \leq \lambda (k-1)^2 \pi r^2 \times \sum_{i=0}^{\infty} \frac{1}{2^i} = 2\lambda (k-1)^2 \pi r^2.$$

Thus, the average number of message transmissions for each sensor is actually the number of neighbors within $(k-1)$-hops distance. Since k is usually small, the number of neighbors within $(k-1)$-hops distance is smaller than $\sqrt{n}$. Therefore, we can conclude that algorithm DClocal is more energy-efficient than DCglobal.

Recall that we proved in previous sections that the ratio between the number of r-nodes selected by DCglobal or DClocal and that of the optimal solution is bounded in. Let the set of optimal r-nodes be C^*, the set of r-nodes selected by DCglobal is C_g, and the set of r-nodes selected by DClocal is C_l. Theorem 2 shows that $|C_g| \leq |C^*| \times H(\max_{S \in C} |S|)$, where $H(n) = \sum_{k=1}^{n} \frac{1}{k}$. Theorem 3 shows that $|C_l| \leq |C^*| \times \left(\frac{2^k - 1}{2^{k+1}} \times \max_{S \in C^*} |S|\right)$. By the definition of $H(n)$, we have $n \geq H(n)$. Therefore, we can conclude that the number of r-nodes selected by DClocal and that of DCglobal is also bounded, and DClocal will select more r-nodes than DCglobal do. From the comparison of DCglobal and DClocal in terms of energy consumption and the number of selected r-nodes, we can conclude that although DClocal selects more r-nodes than DCglobal does, the energy consumption of DClocal is less than that of DCglobal.

Chapter 4

Performance Analysis

In this section, we evaluate the performance of our algorithms DCglobal and DClocal on synthetic data. We develop a network simulator that allows us to vary several characteristics of the network like the network size, the number of sensors, the error threshold, and the distribution of readings, etc. We first introduce the design and settings of the network simulator in Section 4.1. In Section 4.2, we studied the comparison of the energy consumption in algorithm DCglobal, DClocal, and without representative sensors. Finally, we present a sensitivity analysis of our algorithms, varying various parameters in Section 4.3.

4.1 Simulation Model

We design a network simulator to generate the queries, and the sensors associated with their readings. The simulated network consists of 300 sensors randomly deployed in a $[0, 300] \times [0, 300]$ two-dimensional area. The coordinate of the sink is $(0, 0)$. The transmission range of each sensor is set to 40. In order to simulate the spatial correlation between nearby sensors in real-world data, we devised a mechanism to generate the readings of sensors according to the *events* in the network. An event characterize the readings in a specific area. The readings of sensors which are near an event will be affected by the event. We randomly select 50 points in the network as the locations of events. The readings of events is generated following a random walk pattern. The initialized reading of each event is sampled from a normal distribution with mean 25 °C and standard error 20 °C. The move probability for the random walk is set to 0.7 for the purpose to make the readings more volatile, and the step size for each move is 1 °C. The probability for the next reading to increase or decrease is 0.5. The readings of each sensor is affected by all the events in the network. The reading of a sensor at time t is the weighted average of the readings of events at time t. The weight of an event is the inverse of the square of the distance between the sensor and the event.

4.2 Energy Consumption

In this subsection we first conduct experiments on the energy consumption under different situations with range queries existing over random parts of the network. The query points are randomly generated in the network. The query interval of each query is set to 30 time-units with the start time randomly chosen from the simulation time interval $[0, 999]$. The query range of a query is a 30×30 region. 200 queries are submitted in the network.

In Figure 4.1, we compare the energy consumption of algorithm DCglobal, DClocal, and without r-nodes answering queries (called Naïve scheme). The initial energy capacity of each sensor is set to be equal to the simulated cost of 500 message transmissions. We repeat the experiment 10 times and present the average values. In each run, we let the sensors operate for 100 time-units. Algorithm DCglobal and DClocal are executed every 100 time-units to select new r-nodes, respectively. The error threshold is 0.5 °C for both DCglobal and DClocal. In DClocal, the number of hops for a sensor to discover its data-coverage is set to 3.

In DCglobal or DClocal, for each query, a selected r-node reports its readings to the sink if some sensors in its data-coverage are within the query window. In Naïve scheme, all the sensors within a query window report their readings to the sink. We account total remaining

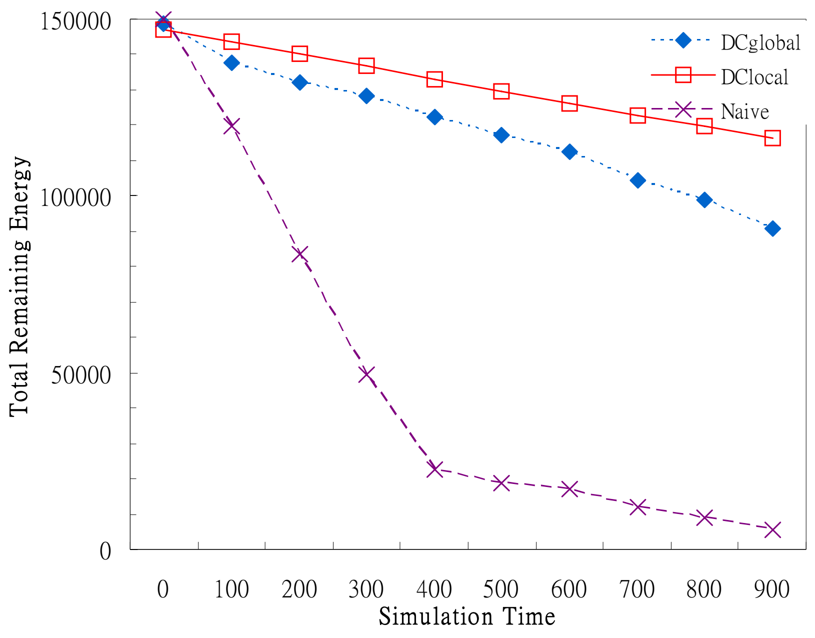


Figure 4.1: The average available energy of different algorithms over time.

energy for DCglobal, DClocal, and Naïve scheme every 100 time-units.

As can be seen in Figure 4.1, the total remaining energy of sensors in Naïve scheme is smaller than that of DCglobal and DClocal. Additionally, it can be seen that the energy consumption speed of DClocal is slower than that of DCglobal. The main reason is that the energy consumed by DCglobal to select r-nodes is much higher since each selection requires all the sensors transmitting their readings to the sink.

In the second experiment, we compare the performance of DCglobal and DClocal with two algorithms. The first one is Snapshot, sensors build a linear regression prediction model to estimate the reading of their 1-hop neighbors [9]. The second one is Max-Min, sensor which has the largest ID among its k-hops neighbors or is the one with largest ID in one of its neighbor's k-hops neighborhood is selected as cluster-head [2].

At first, in Figure 4.2 we compare the number of representatives selected by each algorithm, respectively. Since in Snapshot the selected representatives are 1-hop cluster-head, the number of representatives selected by Snapshot is far more than that of the other algorithms. Because of this property of Snapshot, the number of representatives selected by Snapshot is irrelevant

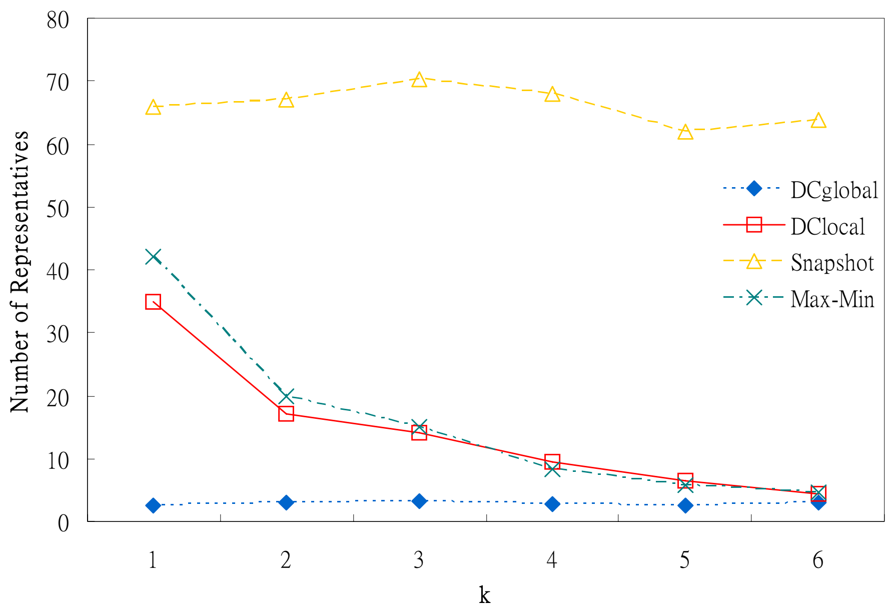


Figure 4.2: The impact of k on different algorithms.

to k, and is almost constant when k increases. Equivalently, the number of r-nodes selected by DCglobal is also constant when k increases. For DClocal and Max-Min, we can observe that for larger value of k, the number of representatives selected by the two algorithms decreases, and the number of representatives of the two algorithms is very close.

We next compare the network lifetime of the 4 algorithms. In this experiment, we follow the definition in [19][7][8][4] and define that the network lifetime is the time that the first sensor dies. For the concern of load balance, we prefer smaller value of the network lifetime. In Figure 4.3, we observed that the lifetime of Max-Min increases rapidly as k increases while the lifetime of the other algorithms remains almost constant. In Max-Min, sensors which have the largest ID are selected as cluster-heads. It is expected that these sensors with larger IDs are always selected and therefore their energy will be run out soon. Since DCglobal, DClocal, and Snapshot selected cluster-heads based on the similarity between sensors, they will not suffer this drawback. The lifetime of Snapshot is lowest because the cluster-heads only have to stand for their 1-hop neighbors, however, the r-nodes in DCglobal and DClocal stand for neighbors in multi-hops distance.

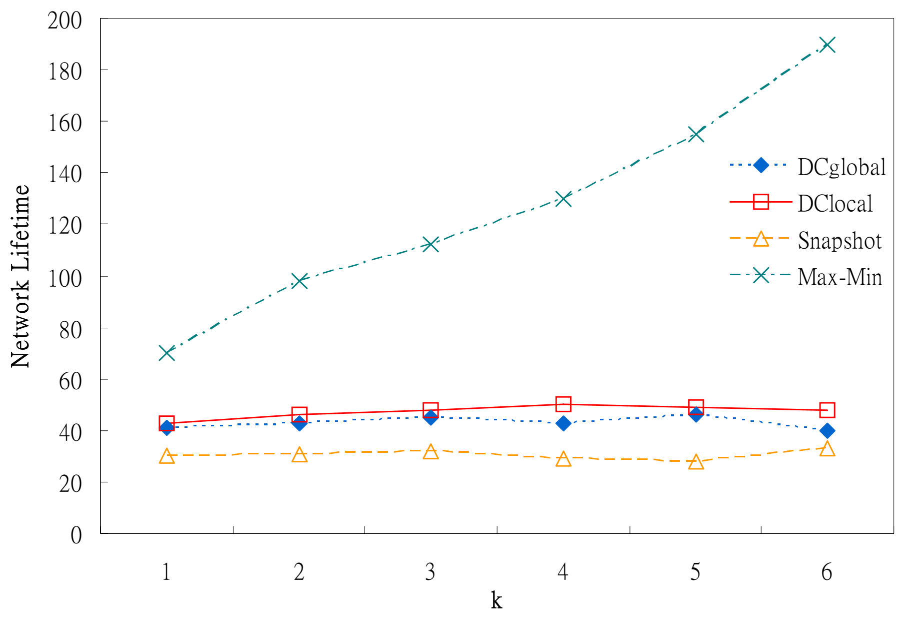


Figure 4.3: The network lifetime of different algorithms when the value of k varies.

From the discussion of Figure 4.2, we can observe that the energy consumption speed of Snapshot is faster than that of DClocal and Max-Min since the representatives selected by Snapshot is more. Although DClocal and Max-Min select almost the same number of representatives, we can observe from Figure 4.3 that the energy consumption speed of DClocal is more balance than that of Max-Min.

4.3 Sensitivity Analysis

In this subsection, we experiment the impact of different parameters by varying the value of one parameter while keeping other parameters unchanged. In each experiment we let the sensors operates for 1000 time-units. We repeat each experiment 10 times and present the average values.

In the first experiment, the communication range of each sensor is set to be 40. The error threshold is set to 0.5 °C. We varied k, the number of hops for each sensor to exchange its readings in algorithm DClocal, from 1 to 6. In Figure 4.4, the total remaining energy of

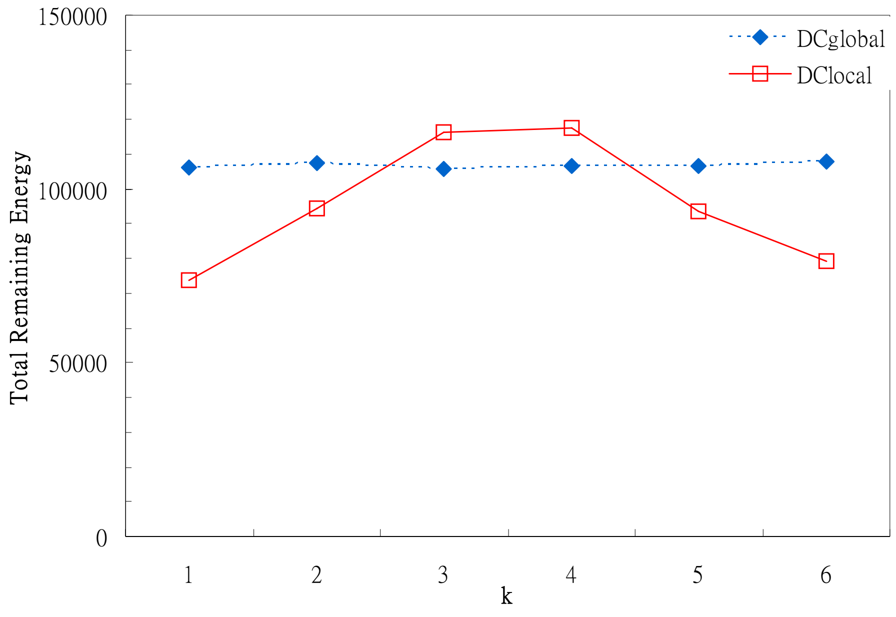


Figure 4.4: The impact of the value of k on the total remaining energy.

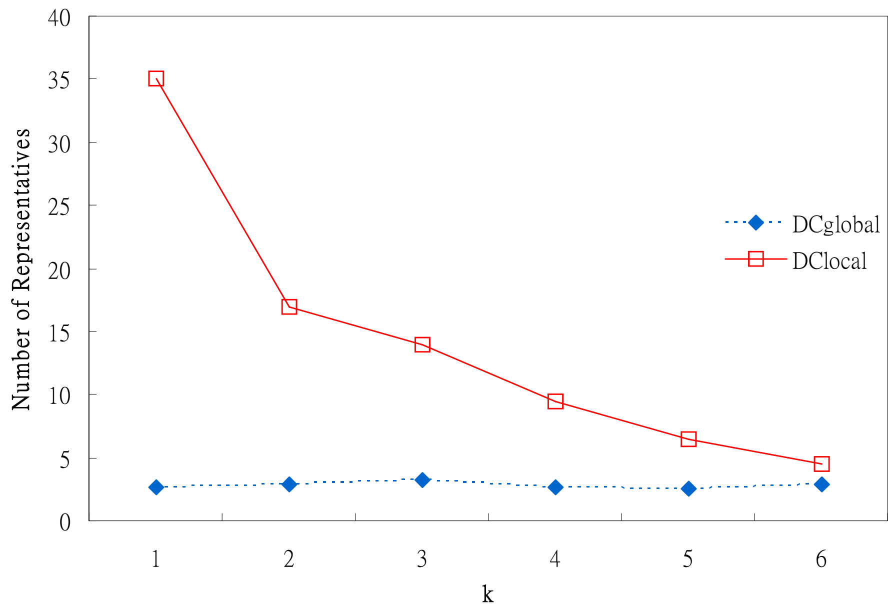


Figure 4.5: The impact of the value of k on the number of r-nodes.

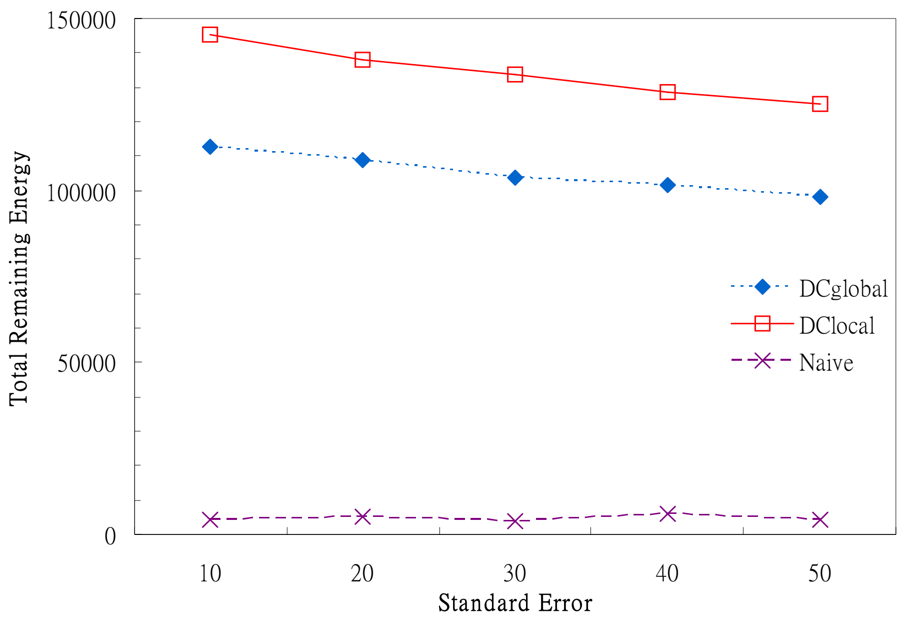


Figure 4.6: The impact of the standard error on the total remaining energy.

DCglobal and DClocal at the end of the simulation is plotted.

As can be seen in Figure 4.5, due to the property of DCglobal, the number of r-nodes selected by DCglobal is not affected by k and the available energy is almost constant. On the other hand, when the value of k is less than 3, the available energy of DCglobal is higher than that of DClocal. This is because that, for smaller k, more r-nodes are selected by DClocal. Therefore, the energy consumed in answering queries is higher. In Figure 4.5, when the value of k increases, the number of r-nodes selected by DClocal decreases. Since the number of r-nodes for answering queries decreases, the available energy of DClocal increases. As shown in Figure 4.4, the available energy of DClocal is higher than that of DCglobal when k is equivalent to 3 or 4.

However, the energy consumption in exchanging readings also increases as the value of k increases. As mentioned before, if the cost of increasing the value of k is higher than the benefit of increasing, the available energy decreases, as can be seen in Figure 4.4 when the value of k is larger than 4. Therefore, from the observation we can concluded that in this example the optimal value of k is 3 or 4.

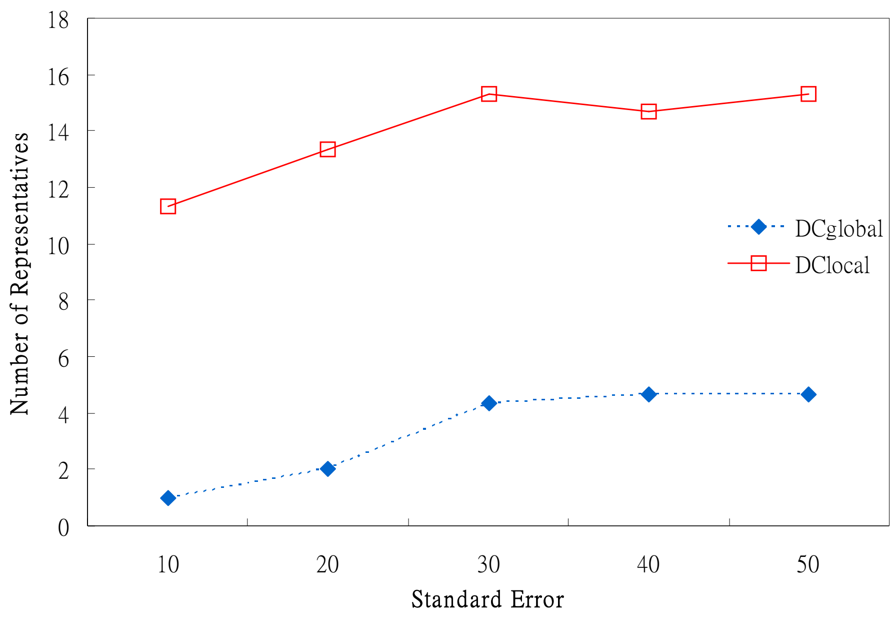


Figure 4.7: The impact of the standard error on the number of r-nodes.

In the second experiment, we varied the standard error of the readings of the events in our network simulator. The standard error controls the volatility of the reading of sensors since the readings of sensors are the weighted average of the events in the network. The total remaining energy of different situation at the end of the simulation is shown in Figure 4.6.

In this experiment, we set the value of k is 3, thus the available energy of DClocal is always higher than that of DCglobal for different values of standard error. In Figure 4.6, for both DCglobal and DClocal, the available energy decreases as the standard error increases. This is because that for more volatile environment, more r-nodes are selected to fully data-cover the whole network. The number of r-nodes selected by DCglobal and DClocal can be seen in Figure 4.7. Therefore the energy consumption of the network in more volatile environment is larger. Since no r-nodes are selected in Naïve scheme, the available energy of Naïve is always the smallest one.

In the third experiment, we varied the values of the error threshold. The available energy of DClocal with different value of k and DCglobal is shown in Figure 4.8. As can be seen in the figure, the available energy of each algorithm is inversely proportional to the error threshold.

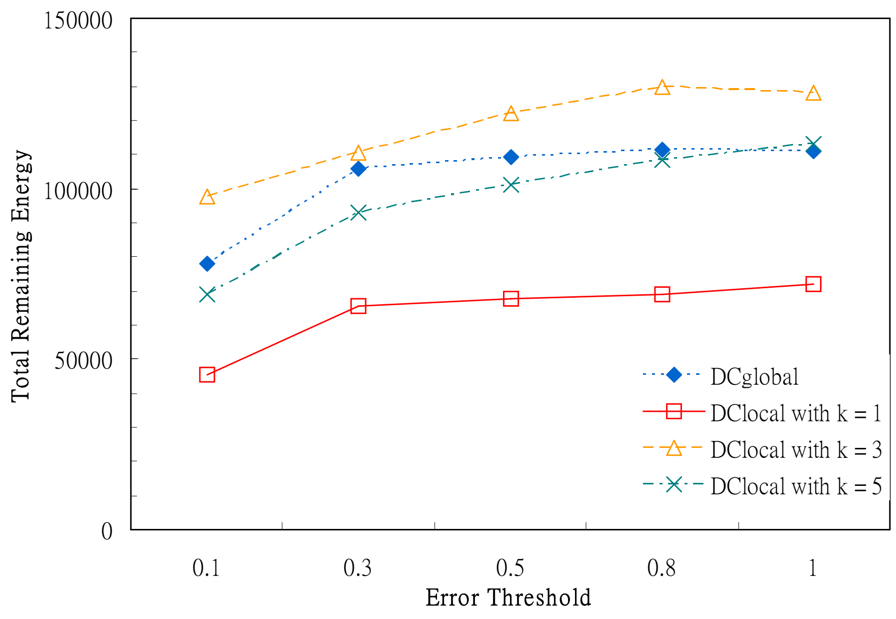


Figure 4.8: The impact of the error threshold on the total remaining energy.

This is because that, for larger value of the error threshold, the number of r-nodes needed to fully data-cover the whole network is less. The number of r-nodes selected by DClocal with different value of k and DCglobal can be seen in Figure 4.9. In addition, the available energy of DClocal with $k = 3$ is the highest among all the cases, which is consistent with the discussion of Figure 4.4. The available energy of DClocal with $k = 1$ is the lowest, since the number of r-nodes in this case is the most.

In addition, for the value of error threshold larger than 0.8, the available energy of the 4 cases tends to be constant. This is because that the r-nodes almost cover all their neighbors within k-hops distance when the value of the error threshold is 0.8.

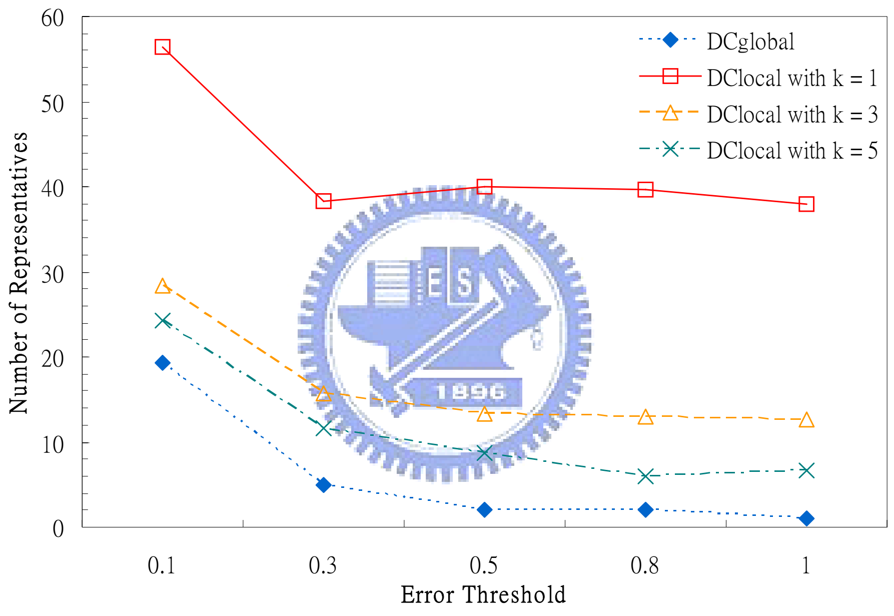


Figure 4.9: The impact of the error threshold on the number of r-nodes.

Chapter 5

Conclusions

In this paper we focused on approximate query processing in sensor networks. Since the readings of nearby sensors are usually correlated, it is power-efficient to group these sensors together and let only one of them respond to queries. We proposed a innovative concept called data-coverage to address the problem. We proved that the data-covering problem which selects minimal number of sensors as r-nodes with the union of their data-coverages data-cover all the sensors in the network is an NP-complete problem by reducing the set-covering problem to the data-covering problem. We then devised two heuristic algorithms DCglobal and DClocal to solve the data-covering problem. Algorithm DCglobal is a centralized algorithm executed at sink. The ratio between the number of selected r-nodes selected by DCglobal and that of the optimal solution is bounded. Since the message cost of selecting r-nodes in executing DCglobal is very high, we devised another distributed algorithm DClocal. Each sensor discovers its data-coverage by exchanging readings with only nearby neighbors. The sensors exchange their data-coverage size with nearby sensors to select sensors with locally largest coverage size as r-nodes. Through the experimental study, it can be seen the performance of DClocal is almost as good as DCglobal with energy consumption less than the energy consumption of DCglobal. The selected r-nodes answer queries within the pre-specified error threshold and the network lifetime is prolonged.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[2] A. D. Amis, R. Prakash, D. Huynh, and T. Vuong. Max-min d-cluster formation in wireless ad hoc networks. In *INFOCOM*, pages 32–41, 2000.

[3] P. Bonnet, J. Gehrke, and P. Seshadri. Towards sensor database systems. In *Mobile Data Management*, pages 3–14, 2001.

[4] C. Buragohain, D. Agrawal, and S. Suri. Power aware routing for sensor databases. In *Infocom*, Dec. 29 2004.

[5] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*, pages 1033–1038. The MIT Press, 1989.

[6] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31 - September 3 2004*, pages 588–599, 2004.

[7] Y. T. Hou, Y. Shi, and H. D. Sherali. Rate allocation in wireless sensor networks with network lifetime requirement. In *MobiHoc*, pages 67–77, 2004.

[8] I. Kang and R. Poovendran. Maximizing network lifetime of broadcasting over wireless stationary ad hoc networks. *MONET*, 10(6):879–896, 2005.

[9] Y. Kotidis. Snapshot queries: Towards data-centric sensor networks. In *Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5-8 April 2005, Tokyo, Japan*, pages 131–142, 2005.

[10] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. In *OSDI*, 2002.

[11] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 9-12, 2003*, pages 491–502, 2003.

[12] A. M. Mainwaring, D. E. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *WSNA*, pages 88–97, 2002.

[13] A. Meka and A. K. Singh. Distributed spatial clustering in sensor networks. In *EDBT*, pages 980–1000, Feb. 31 2006.

[14] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: a geographic hash table for data-centric storage. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA-02)*, pages 78–87, New York, Sept. 28 2002. ACM Press.

[15] W. Wang, J. Yang, and R. R. Muntz. Sting: A statistical information grid approach to spatial data mining. In *Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 186–195, 1997.

[16] A. Y. Wu, M. Garland, and J. Han. Mining scale-free networks using geodesic clustering. In W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 719–724. ACM, 2004.

[17] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. *SIGMOD Record*, 31(3):9–18, 2002.

[18] Y. Yao and J. Gehrke. Query processing in sensor networks. In *Proceedings of the 1st International Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 5-8, 2003*, 2003.

[19] O. Younis and S. Fahmy. Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Trans. Mob. Comput.*, 3(4):366–379, 2004.