Adaptive Approximate Data Collection for Wireless Sensor Networks

Chao Wang, Huadong Ma, *Member*, *IEEE*, Yuan He, *Member*, *IEEE*, and Shuguagn Xiong

Abstract—Data collection is a fundamental task in wireless sensor networks. In many applications of wireless sensor networks, approximate data collection is a wise choice due to the constraints in communication bandwidth and energy budget. In this paper, we focus on efficient approximate data collection with prespecified error bounds in wireless sensor networks. The key idea of our data collection approach ADC (Approximate Data Collection) is to divide a sensor network into clusters, discover local data correlations on each cluster head, and perform global approximate data collection on the sink node according to model parameters uploaded by cluster heads. Specifically, we propose a local estimation model to approximate the readings of sensor nodes in subsets, and prove rated error-bounds of data collection using this model. In the process of model-based data collection, we formulate the problem of selecting the minimum subset of sensor nodes into a minimum dominating set problem which is known to be NP-hard, and propose a greedy heuristic algorithm to find an approximate solution. We further propose a monitoring algorithm to adaptively adjust the composition of node subsets according to changes of sensor readings. Our trace-driven simulations demonstrate that ADC remarkably reduces communication cost of data collection with guaranteed error bounds.

Index Terms—Wireless sensor network, approximate data collection, minimum dominating set.

1 Introduction

Recent advances in low-power wireless technologies have enabled wireless sensor networks (WSNs) in a variety of applications, such as environment monitoring [1], [2], coal mine monitoring [3], object tracking [4], and scientific observation [5], [6]. They enable people to gather data that were difficult, expensive, or even impossible to collect by using traditional approaches [7]. Data collection is a fundamental but challenging task for WSNs, due to the constraints in communication bandwidth and energy budget [7], [8]. On one hand, many applications require persistent long-term data collection, since the gathered data make sense only if the data collection procedure lasts for months or even years without interruption. On the other hand, sensor nodes are often battery powered and deployed in harsh environments, hence data collection strategy must be carefully designed to reduce energy consumption on sensor nodes, so as to prolong the network lifetime as much as possible.

In many applications, it is often difficult and unnecessary to continuously collect the *complete* data from the resource-constrained WSNs. From the point of view of WSNs, directly sending a large amount of raw data to the sink can lead to several undesired problems. First, the data quality may be deteriorated by packet losses due to the limited bandwidth of sensor nodes. Second, intensive data collection incurs excessive communication traffic and potentially results in network congestions. Packet losses caused by such congestions further deteriorate the data quality. Experiments with TinyOS [9] show that packet delivery ratio can be greatly increased by reducing the data traffic within a sensor network. Third, intensive data collection can lead to excessive energy consumption. It is reported in [10] that the lifetime of a sensor network can be increased extraordinarily from 1 month to more than 18 months by lowering the data flow rates of sensor nodes.

Approximate data collection is a wise choice for long-term data collection in WSNs with constrained bandwidth. In many practical application scenarios with densely deployed sensor nodes, the gathered sensor data usually have inherent spatial-temporal correlations [8], [11], [12], [13]. For example, Fig. 1 shows the temperature readings of five nearby sensor nodes deployed in a garden more than 10 hours at night. The temperature readings recorded by the five nodes keep decreasing in the first 4 hours and then become stable in the next 6 hours, which exhibit apparent spatial and temporal correlations among themselves. By exploring such correlations, the sensor data can be collected in a compressive manner within prespecified, application-dependent error bounds. The data traffic can be reduced at the expense of data accuracy [8], [11]. The granularity provided by such approximate data collection is more than sufficient, especially considering the low measuring accuracy of sensors equipped on the sensor nodes. Study on approximate data collection is thus

- *C. Wang and H. Ma are with the Beijing Key Lab of Intelligent Telecommunication Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: wangchao2000@126.com, mhd@bupt.edu.cn.*
- *Y. He is with the TNLIST, School of Software, Tsinghua University and with the CSE Department, Hong Kong University of Science and Technology, Room 823, Main Building, Tsinghua University, Beijing 100084, China. E-mail: heyuan@cse.ust.hk.*
- *S. Xiong is with the IBM China Research Lab, Tower A, Diamond Building, Zhongguancun Software Park, Haidian District, Beijing 100193, China. E-mail: n2xiong@gmail.com.*

Manuscript received 25 May 2011; revised 6 Sept. 2011; accepted 6 Sept. 2011; published online 19 Oct. 2011.
Recommended for acceptance by X. Cheng.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2011-05-0330.
Digital Object Identifier no. 10.1109/TPDS.2011.265.

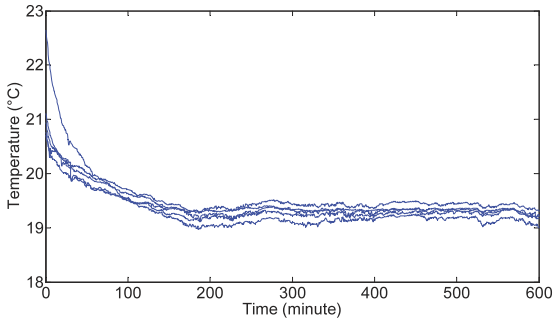

Fig. 1. Temperature readings of five sensors in a garden over ten hours.

motivated by the need of long-term operation of large-scale WSNs, e.g., the GreenOrbs project [6], [14].

There are several factors to be considered in the design of an approach for approximate data collection. First, the data collection approach should be scalable. In many real applications, sensor networks consist of hundreds or even thousands of sensor nodes. For example, GreenOrbs [6], [14] has deployed 330 nodes and expects to deploy $1{,}000+$ sensor nodes in a network. Some existing models, e.g., BBQ [8], are centralized and require complete sensor data set of a WSN. In practice, in large WSNs, the information exchange between the sink and the related sensor nodes may consume considerable bandwidth and the acquisition of complete sensor data set of a WSN is too costly to be practical.

Second, in approximate data collection, the spatial-temporal correlation model used for data suppression should be light-weight and efficient so as to meet the constraints on sensor node's memory and computation capacity. For densely deployed WSNs, many models can be used to describe temporal and/or spatial correlation of sensor data, such as [8], [11], [15], [16], [17]. But it is often nontrivial to build a light-weight correlation model to suppress spatial-temporal redundancy simultaneously. Most of the existing models are too expensive, i.e., consuming a large amount of computing capacity or storage capacity, to be run on the existing sensor nodes [8]. Some of them are too simple to contain enough information, e.g., [15] ignores the trend of sensor readings, or only consider either temporal correlation or spatial correlation separately, e.g., [11], [16], [17]. Our approach shows that simplicity and efficiency can be achieved by exploiting implicit sensor node cooperation and elaborately distributing data processing tasks to sensor nodes.

Third, the data collection scheme should be self-adaptive to environmental changes. Note that physical environmental changes are usually complex and hard to be modeled comprehensively with a simple estimation model [11]. For long-term data collection, the approximate data collection scheme should be capable of automatically adjusting its parameters according to the environmental changes so as to guarantee its correctness.

In this paper, by leveraging the inherent spatial-temporal correlation in sensor data, we propose an efficient approach for approximate data collection in WSNs to simultaneously achieve low communication cost and guaranteed data quality (namely bounded data errors). Our approach, Approximate Data Collection (ADC), is well designed to satisfy the above criterions. ADC achieves low communication cost by exploiting the fact that physical environments generally exhibit predictable stable state and strong temporal and spatial correlations, which can be used to infer the readings of sensors.

Both the scalability and simplicity of ADC are achieved by exploiting implicit cooperation and distributing data processing among sensor nodes. ADC can discover local data correlations and suppress the spatial redundancy of sensor data in a distributive fashion. The distributed spatial data correlation discovery and spatial redundancy suppression is achieved by dividing a WSN into several clusters. The sink can estimate the sensor readings according to the model parameters updated by the cluster heads. This distributed data process scheme makes ADC can be easily applied to WSNs with different system scales. As the sensor network scale increases, ADC only needs to increase the number of clusters. Furthermore, by using clustering-based distributed data process scheme, sensor data can be processed locally in ADC. First, each sensor node is responsible for processing sensor readings generated by itself. Second, the spatial redundancy of sensor data is suppressed by cluster heads that are close to the data source. There are no explicitly control data exchange between sensor nodes and their cluster heads. The sensor data process cost is distributed to all sensor nodes and the sensor data process burden of each cluster head can be easily controlled by adjusting the cluster size.

In order to simultaneously exploit temporal and spatial correlations inherent in sensor data and distribute the data processing task to each sensor node, we propose a novel spatial-temporal correlation model for ADC, which consists of two parts: the local estimation and the data approximation. The local estimation builds a local temporal correlation model for each sensor node to estimate its local readings and reduce the communication cost between each sensor node and its cluster head. The local estimation achieves self-adaption by periodically checking the differences between its estimation and the actual sensor readings. If the actual sensor readings consistently differ from the model in function, the local estimation will regulate its parameters automatically. The data approximation employs a novel tool called correlation graph to describe the spatial correlation among sensor nodes based on the sensor reading information provided by the local estimation. Based on the correlation graph, the sink can recover sensor readings of all sensor nodes with the local estimation data of a small portion of sensor nodes. The errors of recovered data are within the prespecified error bound. Each cluster head is responsible for computing the spatial correlation model, tracking the changes of its local spatial correlation model, and cooperating with the sink to derive a bounded-loss approximation of all the sensor readings.

The main contributions of this work can be summarized as follows: 1) By exploiting the spatial and temporal correlations within WSNs, we propose a novel estimation model to approximate all sensor readings of a WSN using a subset of the sensor readings. Moreover, we prove rated error bounds of data collection using this model. 2) In the process of model-based data collection, we formalize the problem of selecting

the minimum subset of sensor nodes as a minimum dominating set problem, which is NP-hard. We accordingly propose a greedy heuristic algorithm to get an approximate solution. We also propose a monitoring algorithm to adjust the composition of node subsets according to the changes of sensor readings. 3) We evaluate the proposed scheme with trace-driven experiments. The data traces are collected from a real deployed WSN. Simulation results demonstrate that ADC remarkably reduces communication cost by 21 percent, compared with existing approaches.

The rest of the paper is organized as follows: Section 2 briefly discusses the related works. Section 3 elaborates on the local estimation. The details of data approximation are introduced in Section 4. We evaluate ADC by trace-driving simulations in Section 5. Section 6 concludes the paper.

2 Related Work

There have been many related works on data collection in WSNs. Directed diffusion [18] is a general data collection mechanism that uses a data-centric approach to choose how to disseminate queries and gather data. Cougar and TinyDB [19], [20] provide query-based interfaces to extract data from sensor networks. Those works mainly focus on query-based data gathering, but none of them consider the case of efficient long-term large-scale data collection.

Query-based remote continuously approximate data collection in sensor networks is closely related to the problem we study here. One such approach is approximate caching [15], [21], [22], [23] which gives approximate answers to queries in distributed environments with a fixed error bound. The idea is that the sink uses a constant to reconstruct a piecewise constant approximation of the real sensor readings. No updates are sent until a sensor node notices that its value has diverged by more than a given upper bound from the last reading sent to the sink. CONCH [15] also provides a simple spatial suppression technique to suppress update messages of nearby sensors with similar sensor readings. Sensor nodes in CONCH do not update their readings if they hear similar update readings from their neighbors. This kind of approaches, though simple, ignore the trend of sensor readings and only offer a narrow range of predictive capabilities. Approximate caching may also suffer large overhead of update message transmission when many sensor readings change dramatically. Compared with approximate caching, our approach ADC has two obvious advantages. First, ADC exploits the temporal correlation by utilizing a linear trend component which enhances the estimation capability. Second, in a distributed manner, ADC selects only a portion of sensor nodes to update their readings based on a more sophisticated spatial correlation model, which utilizes the trend information of sensor readings generated by the local estimation to further reduce the communication cost.

Other query-based approaches extract data from sensor networks by using Gaussian joint distribution to capture the correlations of sensor readings, such as [8], [24], [25]. BBQ [8] is the first one using multivariate Gaussian joint distribution to capture the correlations of sensor readings. It samples a small fraction of sensor data from a WSN and utilizes Gaussian joint distribution model to estimate the nonsampled sensor readings. Gaussian joint distribution-based approaches have several drawbacks that make them unsuitable for long-term large-scale WSNs. First, this kind of models need an expensive long training phase and a complete data set of every sensor node within a sufficiently long period. Gathering complete data set is too energy consuming and even impractical for large-scale WSNs with limited bandwidth. Second, the correctness of this kind of models requires continuous model update which needs periodically gathering the data generated by every sensor node and disseminating the update information to related sensor nodes. Both of the two tasks are costly for energy-constrained WSNs, even when the update frequency is low. Third, it is almost impossible for this kind of models to precisely control the data error. A Gaussian process (GP) is associated with a mean function $\mathcal{M}(\cdot)$, and a positive-definite kernel function $\mathcal{K}(\cdot,\cdot)$, often called the covariance function. An important property of GPs is that the posterior variance of one of its variable depends on the covariance function $\mathcal{K}(\cdot,\cdot)$, instead of the actual observed value. Hence, the estimation errors of the nonsampled sensor readings are unknown and the estimation quality of the nonsampled sensor readings cannot be guaranteed. In comparison, the data processing burdens of ADC are distributed to each sensor node. The local estimation and the data approximation of ADC are, respectively, settled on each sensor node and each cluster head. This enables sensor data to be processed near or at their sources. The correctness of local estimation is guaranteed by each sensor node locally and the data error bound of ADC is jointly controlled by the local estimation and the data approximation. No explicit control message exchange is required. And the data error bound of ADC can be flexibly adjusted according to the requirements of applications. Such features make our approach ADC scalable and efficient for long-term continuous data gathering applications.

Distributed source coding is a lossless compression technique to address the problem of compressing correlated sources that are not colocated and cannot communicate with each other to minimize their joint description costs. In [26], Slepian and Wolf show that it is possible to compress the data at a combined rate equal to the joint entropy of the correlated source. Distributed source coding technology requires precise and perfect knowledge of the correlations among attributes, and will return wrong answers (without warning) if this condition is not satisfied. In practice, the cost of acquiring precise and perfect knowledge of the correlations among attributes is extremely high. Our approach smartly utilizes a simple probabilistic model to depict spatial correlations among sensor nodes based on rough data generated by each sensor node. All information is processed locally and sensor nodes implicitly cooperate with each other to ensure the data error bound of ADC.

Another technique widely used to reduce communication cost in WSNs is called time-series forecasting. In [16], Lazaridis and Mehrotra propose to use time-series method to create piecewise linear approximations of signals generated by sensor nodes, and send those approximations to the sink. Their approach gathers a large amount of data

and tries to approximate them, rather than exploiting the temporal correlations among sensor readings. In [17], Chatterjea and Havinga describe an adaptive sensor sampling scheme where nodes change their sampling frequencies autonomously based on time-series forecasting, so as to reduce energy consumption. They use time-series forecasting to predict the future sensor readings. The sampling frequency decreases against prediction accuracy, otherwise increases the sample frequency. The skipped samples are replaced by prediction values. In [11], the sink uses simple linear time-series model that consists of a trend component and a stationary autoregressive component to predict the reading of each sensor. Each sensor node updates its linear time-series model individually, instead of raw data. The drawback of these existing works [11], [16], [17] is that they only exploit temporal correlation within sensor data to reduce the communication cost, without considering sensor readings' similarity of nearby sensor nodes, which can be used to suppress the update messages of nearby sensor nodes with similar sensor readings. In ADC, by further exploiting the data generated by the local estimation settled on each sensor node by using time-series forecasting, we propose a novel tool, called correlation graph, to model the spatial correlation within nearby sensor nodes. Based on correlation graph, ADC only needs to collect corresponding data of a subset of sensor nodes, saving the energy cost of information update.

3 Local Estimation

In this section, we present the local estimation which aims to reduce the communication cost among sensor nodes and their cluster heads. By utilizing the local estimation, a sensor node can estimate a newly generated reading through a data model learned from its historic data. Each sensor node sends the parameters of its local estimation data to its cluster head, rather than the raw sensor readings. If the difference between the estimated value and the original value is no larger than a given threshold, the sensor node does not upload its data to its cluster head. As a result, the communication cost is reduced. The update message is send only when the difference between the estimated value and the original value exceeds the prespecified threshold. In the following part of this section, we present the data model used in the local estimation, and then describe how to learn its parameters. The used notations in this paper are summarized in Table I, and the computation procedures on the cluster heads and the sink are discussed in the next section.

TABLE 1
Used Notations

Notation	Meaning
$\mathbb{F}$	The sensor readings of the whole network
$\mathbb{P}$	A Δ-approximation of $\mathbb{F}$
s_i	A sensor node with node ID i
F_i	The readings of sensor node s_i
k	A constant used in local estimation specified by applications
Δ	The error bound depends on applications
ϵ_i	The upper bound of the local estimation error of s_i
$v_i(t)$	The reading of s_i at time t
$p_i(t)$	The estimated value of $v_i(t)$
$e_i(t)$	The estimation error of $v_i(t)$
$m_i(t)$	The linear trend component of s_i
δ_i	The standard deviation of the white noise of the local estimation of s_i
$\delta_i^{'}$	An estimation of δ_i
T	The time interval of sampling
$\chi_i(t)$	A weakly stationary autoregressive component at s_i
C_i	The i^{th} cluster of sensor network S
$G_s(t)$	A partition of sensor network S at time t
$G_i(t)$	A partition of cluster i at time t
W_i	The i^{th} Θ-similar set
$\mathbb{S}(t)$	A predictor set of sensor network S at time t
$\mathbb{S}_i(t)$	A predictor set of cluster i at time t
$g_{i,j}$	The j^{th}cell that belongs to partition $G_i(t)$
$D_{ij}(t)$	The estimation distance between s_i and s_j at time t
$E_{ij}(t)$	$\lvert v_i(t) - p_j(t)\rvert$
$w_i(t)$	The predictor of Θ-similar set W_i
$E_{ir_j}(t)$	$\lvert v_i(t) - w_j(t)\rvert$
r_i	The radius of Θ-similar set W_i
R_i	The representation node of Θ-similar set W_i
$\mathbb{G}_s(V,E,t)$	The correlation graph of sensor network S at time t
$\mathbb{G}_i(V,E,t)$	The correlation graph of cluster i at time t
$\mathbb{D}_s(t)$	The dominating set of $\mathbb{G}_S(V,E,t)$ at time t
$\mathbb{D}_i(t)$	The dominating set of $\mathbb{G}_i(V,E,t)$ at time t
$w_i(t)$	The predictor of W_i at time t

3.1 Data Model

A sensor network S consists of a collection of n sensor nodes $\{s_1, s_2, \ldots, s_n\}$ and a sink node. All data generated by sensor network S can be written as $\mathbb{F} = \{F_1, F_2, \ldots, F_n\}$, where F_i $(1 \leq i \leq n)$ is a time series $v_i(t_1), v_i(t_2), v_i(t_3), \ldots$ generated by sensor node s_i every T seconds. The whole sensor network is grouped into clusters. Each sensor node belongs to one cluster and sends its data to the cluster head through a multihop path. Each cluster head processes the data from sensor nodes in its cluster, and sends the result to the sink through a multihop path. Given a fiducial probability, the sink node requires a Δ-loss approximation of $\mathbb{F}$, denoted as $\mathbb{P} = \{P_1, P_2, \ldots, P_n\}$, in which $P_i = p_i(t_1)$, $p_i(t_2), p_i(t_3), \ldots$ for $1 \leq i \leq n$, $\forall i, t, |v_i(t) - p_i(t)| \leq \Delta$.

We use the model proposed in [11] to estimate the sensor readings of each sensor node. This linear model has many characters that suit for our data collection scheme. First, it is capable of predicting data that evolve slowly over time. Utilizing this linear model, we can abstract the spatial correlation between different sensors and the error of our spatial correlation model can be easily controlled. Second, this linear data model does not require a large amount of training data or a priori knowledge of the distribution of sensor values. Hence, it is suitable for sensor nodes with limited computation capability. Each sensor node learns its data model locally and updates its data model when it is no longer a good approximation for its sensor readings. The burden of computing and maintaining the data model of each sensor node is distributed to each sensor node. In this model, the reading $v_i(t)$ generated by

sensor node s_i at time t is modeled as $m_i(t) + \chi_i(t)$, where $m_i(t)$ is a linear trend component that grows over time, and $\chi_i(t)$ is a three-degree weakly stationary autoregressive component. The linear trend component $m_i(t) = a_i + b_i t$, where a_i and b_i are real constants, and the stationary component $\chi_i(t)$ is defined as follows:

$$\chi_i(t) = \alpha\chi_i(t-1) + \beta\chi_i(t-2) + \gamma\chi_i(t-3) + \delta_i N(0,1), \tag{1}$$

where α, β, γ are real constants, and $\alpha + \beta + \gamma < 1$ since $\chi_i(t)$ is stationary. The function δ_i is the standard deviation of the white noise, and it also provides a measurement of the accuracy of the local estimation. The estimation $p_i(t)$ of value $v_i(t)$ is given by the sum of the current trend $m_i(t)$ and the predicator $\chi_i(t)$, which can be rewritten as a linear combination of the differences of the last three sensor readings and their trend components:

$$\begin{aligned} p_i(t) = m_i(t) &+ \alpha(v_i(t-1) - m_i(t-1)) \\ &+ \beta(v_i(t-2) - m_i(t-2)) + \gamma(v_i(t-3) - m_i(t-3)). \end{aligned} \tag{2}$$

Let $e_i(t) = v_i(t) - p_i(t)$ be the estimation error on node s_i at time t, the following lemma gives the error bound and error probability associated with $p_i(t)$. The detailed proof and analysis of Lemma 1 can be found in [11].

Lemma 1. *Let $\epsilon_i = k\delta_i$, where k is an application specified real constant larger than 1, the actual value $v_i(t)$ is contained in $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability at most $1/k^2$.*

3.2 Parameter Learning

Each sensor node generates a reading every T seconds and inserts it into a queue Q of length N. During the parameter learning phase, each sensor node compute the coefficient a and b of the trend component based on the N readings contained in Q by applying least-squares regression [28]. Then it computes the difference between each reading stored in Q and its estimated trend value $\chi_i(t) = v_i(t) - m_i(t)$ and stores all the values in a queue D. After that, the sensor node uses the data in D to compute the coefficients α, β, γ by applying least-squares regression. Finally, the standard deviation of the white noise can be computed by the following equation:

$$\delta_i' = \left(\sum_{i=1}^{N} (e_j(t_i) - e_j)^2 / (N-1) \right)^{-1/2}, \tag{3}$$

where e_j refers to the average value of the items in D of sensor node s_j.

Since the local estimation model can be uniquely identified by the above-mentioned five coefficients, a sensor node transmits them to it cluster head, and the cluster head can reconstruct the model to estimate the readings of this sensor node.

3.3 Local Estimation Updating

In practice, the environment changes nonlinearly. Our linear model must be self-adaptive to effectively predict nonlinear phenomena. In order to maintain the accuracy of the local estimation model, each sensor node periodically checks the correctness of its local estimation model and updates the parameters of its local estimation as needed. If the estimation error falls outside $[-\epsilon_i, \epsilon_i]$, we call it is a anomaly. Each sensor node detects the anomalies according to the history of its sensor data. The anomalies can be outliers, which transiently diverge from its data model, or distribution changes, which persistently diverge from current data model and suggest that the model needs to be relearned. The local estimation model should be updated only when the distribution changed. In order to distinguish the distribution changes from outliers, we open a monitor window of size WS to monitor the occurrences of anomalies. A sensor node opens a monitor window when it detects an anomaly. At the end of the monitor window, the sensor node relearns its local estimation model only if all estimation errors within the monitor window fall outside $[-\epsilon_i, \epsilon_i]$. Otherwise, we consider sensor readings outside $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ as outliers. Hence, each sensor node requires $WS{*}T$ seconds to determine whether the parameters of its local estimation model should be relearned or not.

In our simulation, WS is set to be 3. Lemma 1 proves that a sensor reading whose estimation error exceeds ϵ_i does not belong to the data distribution with error probability smaller than $1/k^2$. If the estimation error falls outside $[-\epsilon_i, \epsilon_i]$, it is either an outlier, or the data distribution has changed. But, the probability that outliers continuously make estimation errors fall outside $[-\epsilon_i, \epsilon_i]$ is very small. Therefore, WS should be larger than 2. On the other hand, the time required for checking the correctness of the local estimation model is proportional to WS. Increasing WS can reduce the false positive rate of the data distribution change, but also increase the time required for checking the correctness of the local estimation model. At the same time, in our simulation, k is set to be larger enough $(k > 10)$. It is not necessary to set WS to be a large number to reduce the false positive rate of the data distribution change. Hence, we set WS to be 3. According to Lemma 1, we can get the following lemma that depicts the relationships among the real sensor readings $v_i(t)$ and the estimated values $p_i(t)$ maintained at each cluster head.

Lemma 2. *Let $v_i(t)$ be the actual sensor reading of sensor node i and $p_i(t)$ be the estimated value of $v_i(t)$ at time t stored at the cluster head, then $v_i(t) \in [p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability at most $1/k^2$.*

4 Adaptive Data Approximation

Utilizing the data generated by the local estimation of all sensor nodes, the cluster heads cooperate with the sink node to derive a Δ-loss approximation of $\mathbb{F}$. In this section, we first introduce how to obtain a bounded-loss approximation of the actual sensor data, and then present our adaptive data approximation algorithm for approximation data collection in detail.

4.1 Finding a Bounded-Loss Approximation

Sensor readings of nearby sensor nodes are very similar. Since the cluster heads do not know the real sensor readings of their sensor nodes, the cluster heads can only use the

local estimation value of a sensor node to estimate the real readings of the others. In order to reduce the estimation errors, the sensor readings of the estimator should be very close to that of the estimates (the sensor nodes to be estimated). So, we need a metric to measure the sensor reading similarity between two sensor nodes. Definition 1 provides a metric, called estimation distance, to measure the sensor reading similarity between any two sensor nodes based on their local estimation.

Definition 1. *The estimation distance between any two sensor nodes s_i and s_j in S at time t is defined as $D_{ij}(t) = |p_i(t) - p_j(t)|$.*

We find that the estimation error of estimating the real sensor readings of sensor node s_i by utilizing the local estimation of sensor node s_j is related to the estimation distance between s_i and s_j and the estimation error of the local estimation of s_j. Lemme 3 shows their relationship.

Lemma 3. *Let $E_{ij}(t)$ be the estimation error of estimating $v_i(t)$ by $p_j(t)$, we have $E_{ij}(t) \in [0, \epsilon_i + D_{ij}(t)]$ with error probability less than $1/k^2$.*

Proof. According to Definition 1, we can get

$$\begin{aligned} E_{ij}(t) &= |v_i(t) - p_j(t)| = |v_i(t) - p_i(t) + p_i(t) \\ &\quad - p_j(t)| \le |v_i(t) - p_i(t)| + |p_i(t) - p_j(t)| \\ &= |v_i(t) - p_i(t)| + D_{ij}(t). \end{aligned}$$

By Lemma 2, $v_i(t) \in [p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability at most $1/k^2$. Therefore, it is easy to see that the estimation error $E_{ij}(t) \in [0, \epsilon_i + D_{ij}(t)]$ with error probability at most $1/k^2$. □

According to Lemma 3, we can build a model called correlation graph to describe the spatial correlation inherent in sensory data. Before introducing correlation graph, we would like to give the following definitions.

Definition 2. *Sensor node s_i is Θ-similar to s_j at time t, if and only if $\epsilon_i + D_{ij}(t) \le \Theta$, where Θ is a positive real constant.*

Definition 3. *Θ-similar set W_j is a set of sensor nodes that $\exists s_j \in W_j$, $\forall s_i \in W_j - \{s_j\}$, $\epsilon_i + D_{ij}(t) \le \Theta$, where Θ is a positive real constant and refers to the radius of the Θ-similar set W_j. We define sensor node s_j as the representation node of Θ-similar set W_j, denoted as R_j, and $D(s_i, W_j, t) = |v_i(t) - p_j(t)| = \epsilon_i + D_{ij}(t)$ as the distance between s_i and W_j at time t.*

By Definition 3 and Lemma 3, we immediately have the following lemma.

Lemma 4. *The sensor reading of $\forall s_i \in W_j$ at time t can be estimated by the local estimation data of the representation node of W_j. The estimation error $E_{ir_j}(t)$ is less than Δ with error probability at most $1/k^2$, if the radius of Θ-similar set W_j is less than Δ. We define the local estimation data of the representation node of W_j as the predictor of Θ-similar set W_j at time t, denoted as $w_j(t)$.*

Proof. Let s_j be the representation node of Θ-similar set W_j, the estimation error of $\forall s_i \in W_j$ is $E_{ir_j}(t) = |w_j(t) - v_i(t)| = |p_j(t) - v_i(t)| = E_{ij}(t)$. According to Definition 3, we can get $(\epsilon_i + D_{ij}) \le \Theta \le \Delta$. By Lemma 3, it is very clear that $E_{ir_j}(t) \in [0, \Delta]$ with error probability at most $1/k^2$. □

Definition 4. *For any arbitrary Θ-similar set W_j of sensor network S, the Δ-loss approximation of Θ-similar set W_j at time t is a function, $f(t)$, that, for $\forall s_i \in W_j$, $|v_i(t) - f(t)| \le \Delta$ with error probability at most $1/k^2$.*

Definition 5. *For sensor network S, a function set $\mathfrak{V}(t) = \{f_{1,s}(t), f_{2,s}(t), \ldots, f_{x,s}(t)\}$ is a Δ-loss approximation of sensor network S at time t if for $\forall s_i \in S$, $\exists f(t) \in \mathfrak{V}(t)$ that $|v_i(t) - f(t)| \le \Delta$ with error probability at most $1/k^2$.*

By Definition 4 and Lemma 4, we can get that $w_j(t)$ is a Δ-loss approximation of W_j. According to Definition 5, if we divide a sensor network into several Θ-similar sets, we can estimate the actual sensor readings of this sensor network at time t by the predictors of these Θ-similar sets. Let $G_s(t)$ be a partition of sensor network S at time t, referred by $G_s(t) = \{g_{1,s}, g_{2,s}, \ldots, g_{x,s}\}$, where $g_{j,s}$ is a Θ-similar set with $\Theta \le \Delta$. Now, according to the partition $G_s(t)$, we can obtain a predictor set of S at time t, referred by $\mathbb{S}(t) = \{w_{1,s}(t), w_{2,s}(t), \ldots, w_{x,s}(t)\}$, where $w_{j,s}(t)$ is the predictor of $g_{j,s}$ at time t. By Lemma 4, it is easy to see that $\mathbb{S}(t)$ is a Δ-loss approximation of the data generated by sensor network S at time t. Therefore, we have the following theorem.

Theorem 1. *Let $\mathbb{F}(t) = \{v_1(t), v_2(t), \ldots, v_n(t)\}$ be the data generated by sensor network S at time t, any arbitrary predictor set $\mathbb{S}(t)$ of S is a Δ-loss approximation of $\mathbb{F}(t)$.*

Now, we would like to introduce how to find a Δ-loss approximation of sensor network S. Definition 3 describes the spatial correlation between any two sensor nodes. For all sensor nodes that belonged to a wireless sensor network, if we add a directed edge from one sensor node to another one when the first one is Θ-similar to the second one, we can build a directed graph for this wireless sensor network and the spatial correlation between any two sensor nodes can be clearly depicted by this directed graph. We define this directed graph as correlation graph.

Definition 6. *For a prespecified positive real constant Θ, the correlation graph $\mathbb{G}_s(V, E, t)$ of sensor network S is a directed graph at time t, where each vertex in V is a sensor node and $e_{i,j} \in E$ is a directed edge from s_i to s_j. $e_{i,j}$ exists if and only if $(\epsilon + D_{ij}(t)) \le \Theta$. We define Θ as the radius of correlation graph $\mathbb{G}(V, E, t)$. $\forall s_j \in V$ is a neighbor of s_i, if there is a directed edge from s_i to s_j.*

Comparing the definition of representation node of Θ-similar set with the definition of dominating set in graph theory, we find that any arbitrary representation node set $\mathbb{R}_S = \{R_{1,s}, R_{2,s}, \ldots, R_{x,s}\}$ of sensor network S is a dominating set of correlation graph $\mathbb{G}_s(V, E, t)$, where $R_{j,s}$ is the representation node of Θ-similar set $g_{j,s} \in G_s(t)$, and any arbitrary dominating set of correlation graph $\mathbb{G}_s(V, E, t)$, denoted as $\mathbb{D}_s(t)$, is also a representation node set of sensor network S. This means that we can get a representation node set of sensor network S by constructing a dominating set for the correlation graph $\mathbb{G}_s(V, E, t)$. By Theorem 1, we can convert the problem of finding a Δ-loss approximation

of wireless sensor network S into that of finding a dominating set of correlation graph $\mathbb{G}_s(V, E, t)$. Therefore, we have the following lemma.

Lemma 5. *Let $\mathbb{D}_s(t) = \{R_{1,s}, R_{2,s}, \ldots, R_{x,s}\}$ is a dominating set of correlation graph $\mathbb{G}_s(V, E, t)$, where node $R_{j,s}$ is the representation node of Θ-similar set $g_{j,s} \in G_s(t)$. The predictor set of $\mathbb{D}_s(t)$, referred as $\mathbb{S}(t) = \{w_{1,s}(t), w_{2,s}(t), \ldots, w_{x,s}(t)\}$, is a Δ-loss approximation of $\mathbb{F}(t)$, where $w_{j,s}(t)$ is the local estimation value of representation node $R_{j,s}$.*

Unfortunately, the correlation graph $\mathbb{G}_s(V, E, t)$ of sensor network S cannot be directly computed, since the estimation distances between any two sensor nodes cannot be obtained unless the local estimation data of all sensor nodes had been send to the sink node. Therefore, we cannot directly find a Δ-loss approximation $\mathbb{S}(t)$ of $\mathbb{F}(t)$ by finding a dominating set of correlating graph $\mathbb{G}_s(V, E, t)$. For any arbitrary cluster C_i, the local estimation data of every senor node is timely updated to its cluster head h_i. Instead of generating a correlation graph $\mathbb{G}_s(V, E, t)$ for sensor network S, we can generate a correlating graph $\mathbb{G}_i(V, E, t)$ for any arbitrary cluster C_i in its cluster head and compute a Δ-loss approximation for cluster C_i by finding a dominating set of correlating graph $\mathbb{G}_i(V, E, t)$. As a result, we divide the task of finding a Δ-loss approximation of wireless sensor network S into finding a dominating set of the correlation graph of each cluster in its cluster head. Dividing a sensor network into several clusters brings two benefits. The first one is that it makes ADC scalable. ADC can be applied to large-scale wireless sensor network without considering the size of WSNs. The data processing complexity can be easily controlled by the size of each cluster. The second one is that it makes the data processing close to the data source which can reduce the burden of data transmission. According to Lemma 5, we can get the following corollary.

Corollary 1. *Let $\mathbb{D}_i(t) = \{R_{1,i}, R_{2,i}, \ldots, R_{x,i}\}$ be a dominating set of correlation graph $\mathbb{G}_i(V, E, t)$ of cluster C_i and $\mathbb{S}_i(t) = \{w_{1,i}(t), w_{2,i}(t), \ldots, w_{x,i}(t)\}$ be a predictor set of $\mathbb{D}_i(t)$, we can get that $\mathbb{S}(t) = \cup\mathbb{S}_i(t)$ is a Δ-loss approximation of $\mathbb{F}(t)$. Let $\mathbb{S} = \{\mathbb{S}(t_1), \mathbb{S}(t_2), \ldots, \mathbb{S}(t_n)\}$, where for $\forall i$ $t_{i+1} - t_i = T$, we can get that $\mathbb{S}$ is a Δ-loss approximation of $\mathbb{F}$.*

4.2 Adaptive Approximate Data Collection

Corollary 1 provides an approach for finding a Δ-loss approximation of $\mathbb{F}$. Since sensor readings change slowly according to the change of physical phenomena, our adaptive data approximation algorithm should be self-adaptive to the changes of the sensor readings timely. Our data approximation algorithm consists of two parts: data approximation learning algorithm and data approximation monitoring algorithm. The data approximation learning algorithm runs on every cluster head and is responsible for finding a Δ-loss approximation of the true sensor data of each cluster. The data approximation monitoring algorithm consists of two parts. One runs on every cluster head continuously. It monitors the changes of the parameters of the local estimation and decides whether to send an update message to the sink node or not. The other part, which runs on the sink node, is responsible for updating the Δ-loss approximation according to the update messages from each cluster head.

The learning algorithm. Cluster head h_i starts the data approximation learning algorithm after having received a local estimation from each sensor node within cluster C_i. By Corollary 1, cluster head h_i only needs to send $\mathbb{S}_i(t)$ to the sink node, instead of the local estimation of all its sensor nodes. The cluster head sends one message to the sink node for each Θ-similar set. Each message contains all coefficients of the predictor of a Θ-similar set and the IDs of the sensor nodes that belong to this Θ-similar set. The number of messages required by h_i is the cardinality of $\mathbb{S}_i(t)$, denoted as $|\mathbb{S}_i(t)|$. Therefore, the number of messages generated by the data approximation learning algorithm is $|\mathbb{S}(t)|$, which should be minimized to reduce the communication cost between cluster head h_i and the sink node. Since we cannot get the correlation graph $\mathbb{G}_s(V, E, t)$ and divide a wireless sensor network into several clusters, we minimize $\cup|\mathbb{S}_i(t)|$. By Lemma 5, $\min\cup|\mathbb{S}_i(t)| = \cup\min|\mathbb{S}_i(t)| = \cup\min|\mathbb{D}_i(t)|$. Hence, the problem of $\min\cup|\mathbb{S}_i(t)|$ is converted into finding a minimum dominating set of each correlation graph of each cluster, respectively. Finding a minimum dominating set for a directed graph is known to be NP-hard [29]. An approximate minimum dominating set can be obtained in $O(|V|^2)$ time using the greedy algorithm proposed in [30].

The details of the data approximation learning algorithm are shown in Algorithm 1. The cluster head h_i first generates a correlation graph $\mathbb{G}_i(V, E, t)$ (line 1). Then, we adapt the greedy algorithm proposed in [30] to find an approximate minimum dominating set for $\mathbb{G}_i(V, E, t)$. The greedy heuristic algorithm finds a Θ-similar set in each iteration and stops when all nodes are removed from V. In each iteration, the cluster head finds the node v with largest out degree and sets v as the representation node of w_i (line 4-5), and then adds all neighbors of v to w_i (line 6). Then, all the nodes in w_i are removed from vertex set V (line 7-8). The algorithm stops when V is empty. At the end of this phase, each cluster head sends its predictor set and all Θ-similar sets to the sink node.

Algorithm 1. The Data Approximation Learning Algorithm

```
1: Generate correlation graph G_i(V, E, t);
2: i = 0;
3: while |V| > 0 do
4:     v = FindLargestOutDegree(V);
5:     w[i].representation_node=v;
6:     w[i].similarity_set=AllNeighbor(v);
7:     V− = {v};
8:     V− = w[i].similarity_set;
9:     i++;
10: end while
11: return w;
```

The monitoring algorithm. Our data approximation monitoring algorithm guarantees that the predictor set $\mathbb{S}$ stored in the sink node is a Δ-loss approximation of $\mathbb{F}$ at all times. Each cluster head starts the data approximation monitoring algorithm after the data approximation learning algorithm. The data approximation monitoring algorithm updates all local estimation data according to the received local estimation update messages and checks the estimation

error of each Θ-similarity set every T seconds. As we have discussed in Section 3, each sensor node requires WS^*T seconds to check the correctness of it local estimation model, the estimation error check is delayed by WS^*T seconds. If the radius of any Θ-similar set exceeds Δ, the cluster head will adjust its local Θ-similarity sets and send the changes to the sink node. The sink node updates $\mathbb{S}$ according to the update messages from the cluster heads.

The details of the data approximation monitoring algorithm for cluster heads are shown in Algorithm 2. The algorithm first updates all local estimations according to all local estimation update messages M received in last T seconds (line 1). Line 2-12 search each Θ-similar set and find out all sensor nodes that are no longer Θ-similar to their representation nodes, then add them into node list $\mathbb{C}$. All empty Θ-similar sets are removed (line 9-10). Each sensor node in $\mathbb{C}$ tries to find a Θ-similar set to join in by invoking the procedure Join() (line 14). If there is no such a set, a new Θ-similar set will be created for this node by invoking the procedure CreatNewSet() (line 16). Line 20 sends the update messages to the sink node.

Algorithm 2. Monitoring Algorithm for the Cluster Heads

```
1: UpdateMessagePrc(M);
2: for all W ∈ G do
3:     for all s ∈ W do
4:         if D(s, W, t) > Δ then
5:             ℂ = ℂ ∪ {s};
6:             W− = {s};
7:         end if
8:     end for
9:     if W = Ø then
10:        G− = W;
11:    end if
12: end for
13: for all s ∈ ℂ do
14:    flag=Join(s);
15:    if flag==0 then
16:        W = CreatNewSet(s);
17:        G = G ∪ W;
18:    end if
19: end for
20: SendUpdatemsg();
```

The data approximation monitoring algorithm only requires two kinds of update messages: the Θ-similar set creating message and the Θ-similar set updating message. The former creates a new Θ-similar set at the sink node, while the latter is used to update the predictor of a Θ-similar set or add new sensor nodes into it. Note that explicitly sending a message for removing a sensor node from a Θ-similar set is not necessary, because no sensor node belongs to two or more Θ-similar sets simultaneously. Adding a node into a Θ-similar set means removing it from another one.

The details of the data approximation monitoring algorithm for the sink node are shown in Algorithm 3. After receiving an updating message M, the sink node first checks its message type. If it is a Θ-similar set creating message, it first removes all the nodes contained in M from current exiting Θ-similar sets (line 2), then creates a new Θ-similar set and adds all these nodes contained in M into the new Θ-similar set (line 3). If M is a Θ-similar set updating message, the sink node first removes all the nodes contained in M from current exiting Θ-similar sets (line 7), then updates the predictor of the specified Θ-similar set or add all the node contained in M into the specified Θ-similar set (line 8). Finally, all empty sets are removed (line 10-14).

Algorithm 3. Monitoring Algorithm for the Sink Node

```
1: if msgtype is Θ-similar set creating message then
2:     Remove(M);
3:     W = CreatNewSet(M);
4:     G = G ∪ {W};
5: end if
6: if msgtype is Θ-similar set updating message then
7:     Remove(M);
8:     SetUpdate(M);
9: end if
10: for all W ∈ G do
11:     if W = Ø then
12:         G− = W;
13:     end if
14: end for
```

5 Performance Evaluation

In this section, we present an extensive performance evaluation of our approximation data collection approach using real-world data. Our trace-driving simulations demonstrate that the proposed approximation data collection mechanism ADC can notably reduce communication cost.

5.1 Experimental Setup

We conduct trace-driven simulations to evaluate the performance of our scheme using the data traces collected from a real-world deployment. We deploy 88 TelosB sensor motes in a garden and collect sensor readings of temperature generated every 30 seconds for 10 hours at night. There are many trees and bush in the garden. Sensor nodes are randomly in the garden. In order to guarantee the collectivity of the whole sensor network, we vary the distance between two nearby sensor nodes from 3 meters to 6 meters and make sure that each sensor node has at least two nearby sensor nodes in sight. In our experience, the power level of the wireless radio is set to be 2 by using the interface provided by TinyOS. We also collect the topology information of our sensor network, including the neighbor sets of the sensor nodes and the packet loss rates of each links. Fig. 1 shows the temperature readings of five randomly chosen sensor nodes. We can see that the temperature drops from $22.7°C$ to $19°C$.

Note that the used data set includes a large number of missing readings due to unreliable wireless multihop transmissions. We use linear interpolation to infer the missing sensor readings and drop the data of 20 sensor nodes that cannot be recovered, in which eight sensor nodes have 100 percent packet loss rate, eight sensor nodes have packet loss rate larger than 90 percent, and four sensor nodes encounter sensor device errors.

The topology used in our simulation is the same as the real topology of the sensor network deployed in the garden

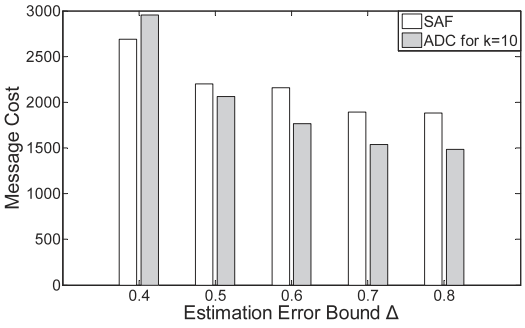

Fig. 2. Communication cost versus estimation error bound Δ.

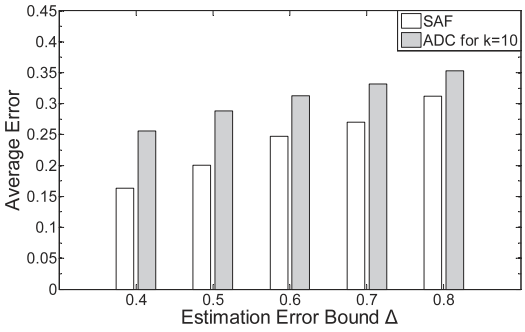

Fig. 3. Average error versus estimation error bound Δ.

and we discard the links with packet loss rates larger than 20 percent, so that the route can be built on relatively reliable links. For experiences of Sections B and C, we divide the whole sensor network into two clusters according to the locations of the sensor nodes. One cluster contains 29 sensor nodes and the other one contains 39 sensor nodes.

5.2 Communication Cost and Data Error

We evaluate the performance of ADC in terms of total communication cost and data error, and compare ADC with SAF [11] which aims reducing communication cost of wireless sensor networks. ADC and SAF use the similar mechanism to reduce the communication cost of wireless sensor network. They both achieve efficiency in communication cost at the expense of data accuracy. Both ADC and SAF use a linear model to describe the sensor reading of a sensor node. SAF use a linear model to compress the sensor data, but our scheme ADC reduce the communication cost by further exploiting the spatial correlation of sensor data based on the linear model use to compress the sensor data. Hence, we compare ADC with SAF. In the simulations, the length of the learning phase in SAF and that in ADC are set to 10 minutes.

Communication cost. We begin with investigating the communication costs of the two approaches, which is defined as the sum of the messages sent by all sensor nodes. As shown in Fig. 2, the communication costs of ADC and SAF decrease against the error bound Δ, and both the two decreasing trends become smooth as Δ increases. Compared with SAF, ADC has less communication costs when $\Delta > 0.4$, and ADC achieves more message saving as Δ increases. When $\Delta = 0.8$, the communication cost of ADC is only about 79 percent of that of SAF.

It should be noted that when $\Delta = 0.4$, the communication cost of ADC is larger than that of SAF. The first reason is that the message paths used by ADC are longer than that used by SAF, because the message paths used by ADC traverse the cluster heads and SAF directly send its message to the sink node. The second one lies in that ADC brings in exorbitant communication cost on informing the sink node the updates of the Θ-similar sets when Δ is small. Specifically, when Δ is small, the radius of Θ-similar sets is small and the number of Θ-similar sets increases. More messages are required to update the changes of the Θ-similar sets. Moreover, it is more likely that a sensor node may frequently leave or join in a Θ-similar set with small prespecified error bound Δ, because its expected estimation error is more likely to exceed small prespecified error bound Δ. This increases the number of messages required to inform the sink node the updates of the Θ-similar sets. More details of the impact of the radius of Θ-similar sets on the communication cost will be given in the next section.

Data error. Next we compare the data errors (measured in absolute value) introduced by SAF and ADC. Fig. 3 illustrates the average data errors of SAF and ADC for varying data error bounds. Recall that the data error introduced by ADC can be divided into two parts: local estimation error and data approximation error. The former depends on the Gaussian white noise $N(0,1)$ and k, while the latter is the difference between the local estimation of a sensor node and that of its representation node. Since the radius of the Θ-similar sets is set to Δ in ADC, the increase of Δ allows the Θ-similar sets to contain sensor nodes with larger estimation errors. As a consequence, the average data error of ADC increases with Δ, as shown in Fig. 3. Although the average data error of ADC is larger than that of SAF, the difference between them is very small: it is always less than $0.1^{\circ}\mathrm{C}$ and decreases slowly against Θ.

Fig. 4 depicts the CDFs of data error of ADC and SAF under variant Δ. We can see that as Δ increases from 0.5 to 0.8, the distribution of data error in SAF is more balanced than in ADC, i.e., more sensor nodes have a small data error in SAF, which is in accordance with the results in Fig. 3. These results imply that ADC achieves efficiency in communication cost at the expense of data error, and more importantly, the error bound can be adjusted by users.

5.3 Impacts of Parameters

The following simulations focus on the impacts of parameter k and Δ on the efficiency and accuracy of ADC.

The communication cost of ADC can be divided into two parts: the communication cost of the local estimation (CCLE) (the green parts in Fig. 5), which is used to transmit the local estimation data of each sensor node from each senor node to its cluster head, and the communication cost of the data approximation (CCDA) (the red parts in Fig. 5), which is used to transmit the information related to each Θ-similar set from cluster heads to the sink node. As shown in Fig. 5, the experiment results reveal that the values of k and Δ have notable impacts on communication cost. First, the communication cost reduces against Δ. Since the radius of Θ-similar set, Θ, is set to Δ in ADC and a large Θ means a large distance between a sensor node and its Θ-similar set. A larger distance can endure larger sensor reading changes and this reduces the frequency of

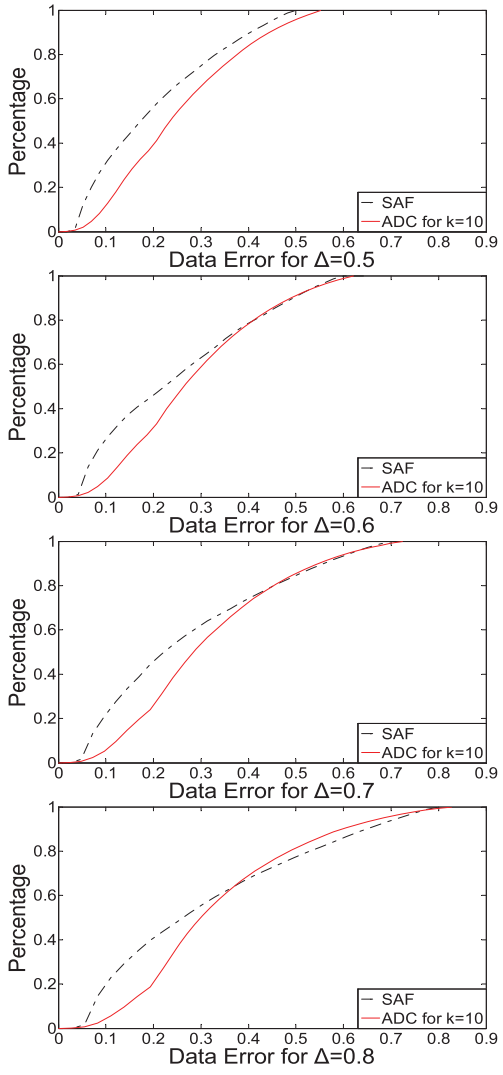

Fig. 4. CDFs of the data errors of SAF and ADC for varying Δ.

Θ-similar set member changes caused by sensor reading changes. Hence, the communication cost reduces against Δ. CCLE is only affected by the parameter k and reduces against K since the local estimation can endure more data error as k increase. Second, Θ has more impact on the communication cost than k. As show in Fig. 5, CCDA drops faster than CCLE as Δ increase. A small Δ and a large k can make Θ-similar sets unstable and increase CCDA of ADC, as illustrated in Fig. 6 when $\Delta = 0.5$. In ADC, for each sensor node, the upper bound of the

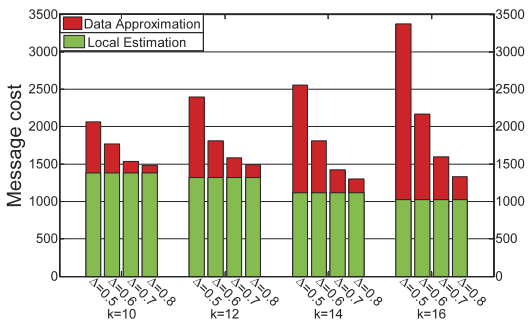

Fig. 5. Communication cost versus k and Δ.

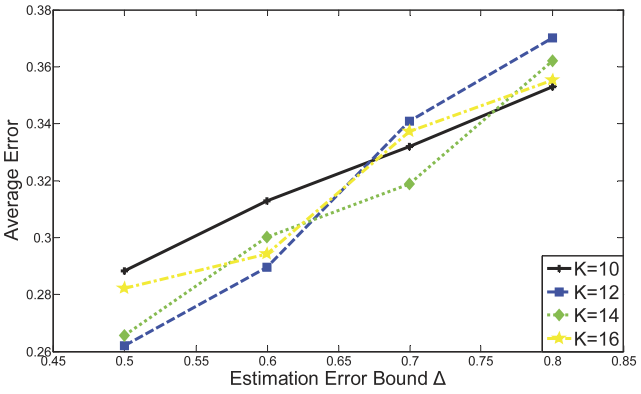

Fig. 6. Average error versus k and Δ.

distance between a sensor node and its representation node, denoted by UD, is determined at the local estimation phase and is equal to the difference between Δ and it local estimation upper bound which is given by $k\delta_i$ in Lemma 2 and increases as K. Hence, for each sensor node, UD can be regarded as a constant before the local estimation be recomputed, and UD is small when Δ is small and k is large. Since a sensor node leaves its Θ-similar sets only when the distance between it and its representation exceed UD, it is more likely that a sensor node will leave or join in a Θ-similar set when UD is small.

Next we evaluate the impacts of k and Δ on the accuracy of ADC. Fig. 6 shows the data error of ADC for different k and Δ. Fig. 7 shows the cumulative distribution of the data error for different k and Δ. For a given Δ, the cumulative distribution curves of the data error for different k is similar, hence we only give the cumulative distribution curves of the data error for $k = 10$ and $k = 16$. As shown in Fig. 6, the average data error of ADC is not obviously affected by k. In Fig. 7, we can see that the CDF of data error does not change obviously as k changes. According to Figs. 6 and 7, we can infer that the accuracy of ADC mainly depends on Δ, while k has very little impact. The error upper bound of local estimation depends on k. And the increase of k results in the increase of the local estimation error. The data errors introduced by the representation nodes are limited by the difference between the local estimation error and Δ. Since whether a sensor node joins in a Θ-similar set or not depends on the sum of the two kinds of errors, the increase of k reduces the estimated distance between sensor nodes and their representation nodes when Δ is fixed.

Now, we investigate the maximum error of ADC. From Fig. 8, we can see that the maximum error of ADC is a little larger than the error bound Δ and increase with Δ. From Fig. 7, we can see that a small fraction of data error is larger than the error bound Δ. The local estimation of ADC relearns the parameters of its data model only when the absolute value of the estimation error continuously exceeds ϵ_i several times. Otherwise, these sensor readings with estimation error larger than ϵ_i are considered as outliers. The data errors of these outliers may be larger than Δ. Therefore, we conclude that the maximum error of ADC is approximately bounded by Δ.

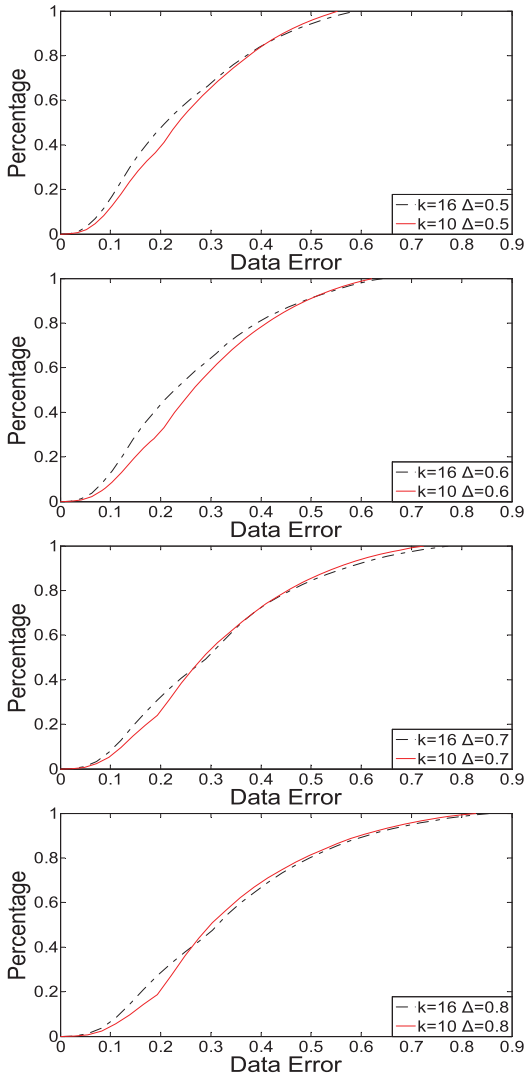

Fig. 7. CDFs of the data errors of ADC for varying k and Δ.

5.4 Impacts of Cluster Size

The following simulation focus on studying the impact of cluster size on the message efficiency and accuracy of ADC. We divide the whole sensor network into two clusters, three clusters and four clusters, respectively, according to the locations of the sensor nodes. The size of each cluster is given in Table 2.

Now we investigate the communication costs for different cluster size. Fig. 9 shows that the cluster size has difference impact on CCLE (the green part in Fig. 9)

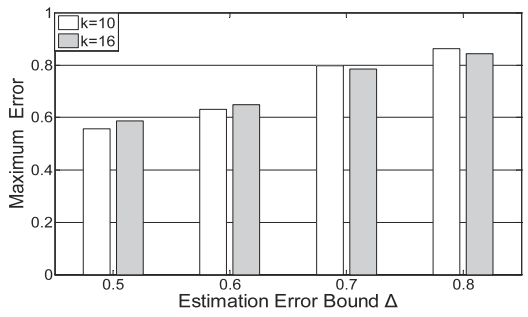

Fig. 8. The maximum error of ADC for varying k and Δ.

TABLE 2
Cluster Size

NO. of clusters	Cluster 1	Cluster 2	Cluster 3	Cluster 4
2 Clusters	29	39	None	None
3 Clusters	16	25	27	None
4 Clusters	12	19	18	19

and CCDA (the red part in Fig. 9). First, the CCLE reduces against the cluster size. The CCLE relates to the number of messages generated by each sensor nodes and the distances among sensor nodes and their cluster heads. According to the local estimation part, the number of messages generated by sensor nodes is decided by the parameter k. The distances between sensor nodes and their cluster heads decrease as cluster size. As a result, the CCLE reduces against the cluster size. Second, the CCDA decreases against the cluster size at first when the number of clusters changes from 2 to 3, but increases as the cluster size when the number of clusters changes from 3 to 4, especially when $\Delta = 0.5$ in Fig. 9. The CCDA mainly relates to two factors: the number of messages generated

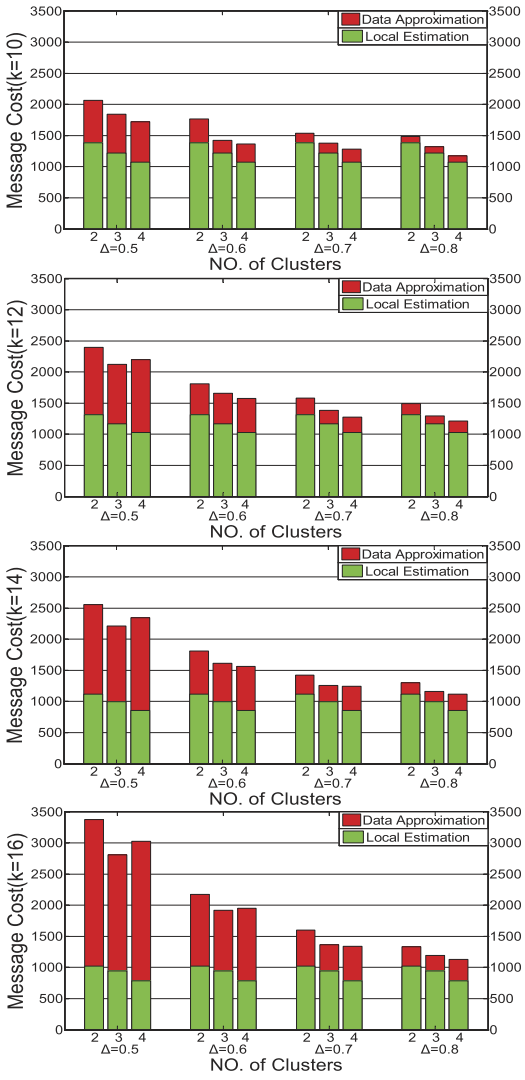

Fig. 9. Communication cost for varying cluster size, k and Δ.

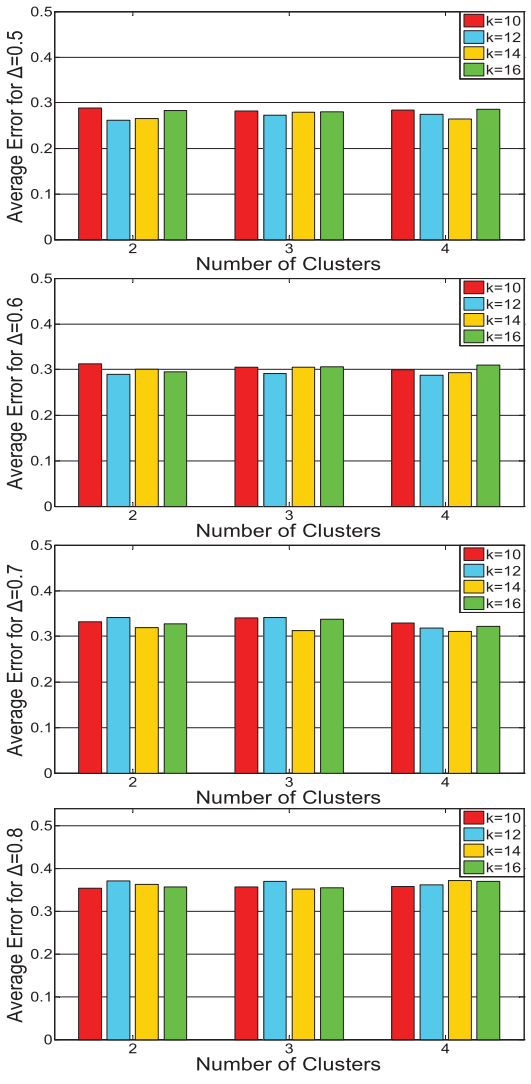

Fig. 10. The average data errors of ADC for varying cluster size, k and Δ.

by our data approximation algorithm and the distances between the cluster heads and the sink node. The number of sensor nodes that can be included in a Θ-similar sets decrease with the size of cluster. But, the total number of sensor nodes is not changed. Therefore, the number of Θ-similar sets increases when the cluster size reduce and this makes the number of messages that generated by our data approximation algorithm increase when the cluster size reduce. But, when the cluster size reduces, more cluster heads are close to the sink node. The reduction of path length can reduce the CCDA. As a result, those two reasons combined together make the CCDA decrease when the cluster size decrease at the beginning, but increase when the cluster size further decrease. Third, the total communication cost reduces when the cluster size reduce in most cast, and the impact of the cluster size on the CCDA decreases when Δ increases. Fig. 9 also shows that Δ is main fact that affect the efficiency of ADC, which is in accordance with the analysis in the previous section.

The impact of cluster size on the accuracy of ADC is shown in Fig. 10. For the same Δ, the average data error slightly fluctuates within a narrow range when the cluster sizes and k change. This is in accordance with Fig. 6. As we analyzed in the previous section, the data errors are limited by Δ and are the sum of two errors: the local estimation error and the data approximation error. Increasing one of them will reduce the other one when Δ is fixed. According to Figs. 10 and 6, we can conclude that the accuracy of ADC mainly depends on parameter Δ. Parameter k and the cluster size have very limited impact on the accuracy of ADC.

6 Conclusion

In this paper, we propose a novel approximate data collection strategy ADC in WSNs. ADC can approximate all readings of a sensor network by exploiting the fact that physical environments frequently exhibit predictable weak stable state and strong temporal and spatial correlations between sensor readings. Our work detects data similarities among the sensor nodes by comparing their local estimation models rather than their original data. The simulation results show that our approach can greatly reduce the amount of messages in wireless communications by as much as 21 percent compared with existing works. In the future, we plan to implement and evaluate our work in real sensor networks.

Acknowledgments

The research reported in this paper is supported in part by the National Basic Research Program of China (973 Program) under Grant No. 2011CB302701, the National Natural Science Foundation of China under Grants No. 61170213 and No. 60833009, the National Science Funds for Distinguished Young Scientists under Grant No. 60925010 and the Funds for Creative Research Groups of China under Grant No. 61121001.

References

[1] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A Macroscope in the Red Woods," *Proc. Third Int'l Conf. Embedded Networked Sensor Systems (SenSys '05)*, 2005.

[2] M. Li, Y. Liu, and L. Chen, "Non-Threshold Based Event Detection for 3D Environment Monitoring in Sensor Networks," *IEEE Trans. Knowledge and Data Eng.*, vol. 20, no. 12, pp. 1699-1711, Dec. 2008.

[3] M. Li and Y. Liu, "Underground Coal Mine Monitoring with Wireless Sensor Networks," *ACM Trans. Sensor Networks*, vol. 5, no. 2, pp. 1-29, 2009.

[4] Z. Yang and Y. Liu, "Quality of Trilateration: Confidence based Iterative Localization," *IEEE Trans. Parallel and Distributed Systems*, vol. 21, no. 5, pp. 631-640, May 2010.

[5] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and Yield in a Volcano Monitoring Sensor Network," *Proc. Seventh Symp. Operating Systems Design and Implementation (OSDI '06)*, 2006.

[6] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, "Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest," *Proc. Seventh ACM Conf. Embedded Networked Sensor Systems (SenSys '09)*, 2009.

[7] 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 '06)*, 2006.

[8] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-Driven Data Acquisition in Sensor Networks," *Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04)*, 2004.

[9] C. Wan, S.B. Eisenman, and A.T. Campbell, "Coda: Congestion Detection and Avoidance in Sensor Networks," *Proc. First Int'l Conf. Embedded Networked Sensor Systems (SenSys '03)*, 2003.

[10] C. Guestrin, P. Bodi, R. Thibau, M. Paski, and S. Madde, "Distributed Regression: An Efficient Frame Work for Modeling Sensor Network Data," *Proc. Third Int'l Symp. Information Processing in Sensor Network (IPSN)*, 2004.

[11] D. Tulone and S. Madden, "An Energy Efficient Querying Framework in Sensor Networks for Detecting Node Similarities," *Proc. Ninth ACM Int'l Symp. Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM '06),* 2006.
[12] A. Jindal and K. Psounis, "Modeling Spatially-Correlated Sensor Network Data," *ACM Trans. Sensor Networks,* vol. 2, no. 4, pp. 466-499, 2006.
[13] M. Li and Y. Liu, "Iso-Map: Energy-Efficient Contour Mapping in Wireless Sensor Networks," *IEEE Trans. Knowledge and Data Eng.,* vol. 22, no. 5, pp. 699-710, May 2010.
[14] The GreenOrbs project, http://www.greenorbs.org, 2011.
[15] A. Silberstein, R. Braynard, and J. Yang, "Constraint Chaining: On Energy-Effcient Continuous Monitoring in Sensor Networks," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '06),* 2006.
[16] I. Lazaridis and S. Mehrotra, "Capturing Sensor-generated Time Series with Quality Guarantees," *Proc. 19th Int'l Conf. Data Eng.,* 2003.
[17] S. Chatterjea and P. Havinga, "An Adaptive and Autonomous Sensor Sampling Frequency Control Scheme for Energy-Efficient Data Acquisition in Wireless Sensor Networks," *Proc. IEEE Fourth Int'l Conf. Distributed Computing in Sensor Systems (DCOSS '08),* 2008.
[18] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proc. MobiCom,* 2000.
[19] S. Madden, W. Hong, J.M. Hellerstein, and M. Franklin, "TinyDB Web Page: http://telegraph.cs.berkeley.edu/tinydb," 2011.
[20] Y. Yao and J. Gehrke, "Query Processing in Sensor Networks," *Proc. CIDR,* 2003.
[21] C. Olston and J. Widom, "Best Effort Cache Sychronization with Source Cooperation," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '02),* 2002.
[22] C. Olston, J. Jiang, and J. Widom, "Adaptive Filters for Continuous Queries over Distributed Data Streams," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '03),* 2003.
[23] I. Lazaridis and S. Mehrotra, "Capturing Sensor-Generated Time Series with Quality Guarantee," *Proc. 19th Int'l Conf. Data Eng. (ICDE),* 2003.
[24] C. Guestrin, A. Krause, and A.P. Singh, "Near-Optimal Sensor Placements in Gaussian Processes," *Proc. 22nd Int'l Conf. Machine Learning (ICML '05),* 2005.
[25] A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin, "Simultaneous Placement and Scheduling of Sensors," *Proc. Int'l Conf. Information Processing in Sensor Networks (IPSN '09),* 2009.
[26] D. Slepian and J. Wolf, "Noise Less Coding of Correlated Information Sources," *IEEE Trans. Information Theory,* vol. IT-19, no. 4, pp. 471-480, July 1973.
[27] A.D. Wyner and J. Ziv, "The Rate-Distortion Function for Source Coding with Side Information at the Decoder," *IEEE Trans. Information Theory,* vol. IT-22, no. 1, pp. 1-10, Jan. 1976.
[28] G. Box and G.M. Jenkins, *Time Series Analysis: Forecasting and Control.* Prentice Hall, 1994.
[29] S. Howe, "Dominating Sets of Random 2-in 2-Out Directed Graphs," *The Electronic J. Combinatorics,* vol. 15, no. 29, 2008.
[30] P. AK, "Analysis of a Greedy Heuristic for Finding Small Dominating Sets in Graphs," *Information Processing Letters,* vol. 39, no. 5, pp. 237-240, 1991.
[31] Moteiv, Telos Revb Data Sheet, http://www.moteiv.com/pr/2004-12-09-telosb.php, Dec. 2004.
[32] S. Pattem, B. Krishnamachari, and R. Govindan, "The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks," *Proc. Third Int'l Symp. Information Processing in Sensor Networks (IPSN '04),* 2004.

Chao Wang received the BA degree from Beijing University of Posts and Telecommunication in 2004. He is currently working toward the PhD degree at Beijing University of Posts and Telecommunication. His research interests include sensor networks and Internet of things.

Huadong Ma (M'99) received the BS degree in mathematics from Henan Normal University in 1984, the MS degree in computer science from Shenyang Institute of Computing Technology, Chinese Academy of Science in 1990, and the PhD degree in computer science from Institute of Computing Technology, Chinese Academy of Science in 1995. He is currently a professor and the director of Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Dean of School of Computer Science, Beijing University of Posts and Telecommunications, China. He visited UNU/IIST as a research fellow in 1998 and 1999, respectively. From 1999 to 2000, he held a visiting position in the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan. He was a visiting professor at The University of Texas at Arlington from July to September 2004, and a visiting professor at Hong Kong University of Science and Technology from Dec. 2006 to Feb. 2007. His current research interests include multimedia system and networking, Internet of things, and sensor networks, and he has published more than 100 papers and 4 books on these fields. He is member of the IEEE and the ACM.

Yuan He received the BE degree from University of Science and Technology of China in 2003, the ME degree from Institute of Software, Chinese Academy of Sciences in 2006, and the PhD degree from Hong Kong University of Science and Technology. He is a member of Tsinghua National Lab for Information Science and Technology. His research interests include sensor networks, peer-to-peer computing, and pervasive computing. He is a member of the IEEE, the IEEE Computer Society, and the ACM.

Shuguagn Xiong received the PhD degree in computer science from Harbin Institute of Technology, China, in 2011. He is currently a staff researcher in IBM China Research Lab, Beijing, China. His research interests include data management and networking in wireless ad hoc and sensor networks.