2010 16th International Conference on Parallel and Distributed Systems

Approximate Data Collection for Wireless Sensor Networks

Chao Wang
Beijing Key Lab of Intelligent Telecomm. Software and Multimedia
Beijing University of Posts and Telecomm.
Beijing, China
Email: wangchao.cs@gmail.com

Huadong Ma
Beijing Key Lab of Intelligent Telecomm. Software and Multimedia
Beijing University of Posts and Telecomm.
Beijing, China
Email: mhd@bupt.edu.cn

Yuan He
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong, China
Email: heyuan@cse.ust.hk

Shuguang Xiong
Department of Computer Science and Technology
Harbin Institute of Technology
Harbin, China
Email: n2xiong@gmail.com

***Abstract*—Data collection is a fundamental issue in wireless sensor networks. In many application scenarios for 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 given 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 a global approximate data collection on the sink according to model parameters uploaded by cluster heads. Specifically, we propose a local estimation model to approximate the readings of several subsets of sensor nodes, 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 use a greedy heuristic algorithm to find an approximate solution. We also propose a monitoring algorithm to adjust these subsets according to the changes of sensor readings.Our trace-driving simulation results show that our data collection approach ADC can notably reduce the communication cost with given error bounds.**

***Keywords*-wireless sensor network, approximate data collection, minimum dominating set.**

I. Introduction

Recent advances in low-power wireless technologies have enabled wireless sensor networks (WSNs) to be used in a variety of applications, such as environment monitoring [5] and scientific observation [6]. In WSNs, data collection is a fundamental but challenging task, due to the constraints in communication bandwidth and energy budget [15, 23]. 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 cost of the 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 set from a resource-constrained WSN. From the point of view of sensor networks, directly sending a large amount of raw data to the sink node can leads to many serious problems. First, the data quality may also be deteriorated by packet dropping due to the limited bandwidth of sensor nodes. Second, intensive data collection can lead to excessive energy consumption. It further potentially results in network congestions and in turn causes severe packet losses, which greatly deteriorate the data quality. In many practical application scenarios for sensor networks, the gathered sensor data usually have spatial-temporal correlations. For example, Figure 1 shows the temperature readings of five nearby sensor nodes deployed in a garden over ten hours at night. The temperatures recorded by the five nodes keep decreasing in the first 4 hours, and then become stable in the following 6 hours. By utilizing compressing techniques based on exploring these correlations, the obtained data can be within pre-specified, application-dependent error bounds. The granularity provided by such approximate data collection is more than sufficient, especially considering the low measuring accuracy of the sensor devices equipped by sensor nodes.

Approximate data collection is a wise choice for real WSN applications, e.g., the GreenOrbs project [22, 24]. GreenOrbs is an all-year ecological surveillance sensor network in the forest motivated by the need of long-term large-scale sensing for continuous environmental surveillance, precise forestry measurements and forestry research. GreenOrbs collects various sensory data including tempera-

1521-9097/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPADS.2010.32

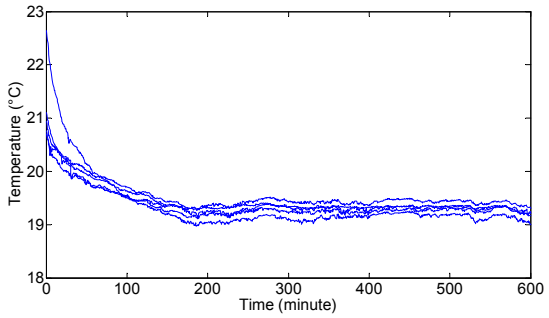

Figure 1. Temperature readings of five sensors in a garden over ten hours.

ture, humidity and illumination over an area around 20,000 square meters. This data was never before available to the forestry research. In this paper, we focus on efficient approximate data collection in WSNs with bounded errors. There are several factors should be considered when we design our energy-efficient approximate data collection scheme. First, such scheme must be adaptive to physical phenomena changes. Physical phenomena are usually complex and hard to be modeled in its entirety by a simple estimation model. Second, we need a simple and effective model to describe spatial correlations between sensor nodes. In dense-deployed sensor networks, it is very common that nearby sensor nodes have similar sensor readings [14]. It is not easy to find a simple and efficient spatial correlation model to suppress spatial redundancy.Our goal is to find a simple and efficient mechanism to suppress both temporal and spatial redundancy.

Our approach, ADC (Approximate Data Collection), exploits the fact that physical environments frequently exhibit predictable weak stable state and strong temporal and spatial correlations that can assist us in inferring the readings of sensors. The key idea of ADC is to divide a sensor network into clusters, discover local data correlations on each cluster head, and perform a global approximate data collection on the sink according to model parameters uploaded by cluster heads. ADC consists of two parts: the local estimation scheme and the data approximation scheme. In the local estimation phase, each sensor node builds a data model to estimate its local readings, while in the data approximation phase, each cluster head maintains the parameters of the data models in the cluster, and cooperates with the sink to derive a bounded approximation of all the sensor readings.

In summary, the main contributions of our work are: (1) By exploiting the spatial and temporal correlations within a WSN, we propose a local estimation model to approximate the readings of a subset of sensor nodes. 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 into a minimum dominating set problem, which is NP-hard. And we propose a greedy heuristic algorithm to find an approximate solution. We also propose a monitoring algorithm to adjust these subsets according to the changes of physical environment. (3) We evaluate the proposed scheme with trace-driven experiments. Simulation results show that our data collection approach can notably reduce the amount of communication cost in sensor networks as much as 21% compared with existing works.

The rest of the paper is organized as follows: Section II briefly discusses the related works. Section III describes the local estimation scheme. The details of data approximation are introduced in section IV. We evaluate ADC by trace-driving simulations in Section V. Finally, Section VI concludes the paper.

II. Related Work

There are many works on data collection in wireless sensor networks. Directed diffusion [11] is a general data collection mechanism that uses a data-centric approach to choose how to disseminate queries and gather data. Cougar and TinyDB [12, 13] provide query-based interfaces to extract data from sensor networks. None of these works consider efficient approximate data collection.

Query-based remote continuously approximation data collection in sensor networks is closely related to the problem we study here. One such approach is approximation caching [8, 9, 10, 14] which give approximate answers to queries in distributed environments with a fixed error bound. The idea is that the sink node uses a constant to reconstruct a piecewise constant approximation of the real sensor readings. No updates are sent until a sensor notices that its value has diverge by more than a given upper bound from the last reading sent to the sink. CONCH [14] also provides an simple spatial-temporal 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. These approaches, though simple, ignore the trend of sensor readings and only offer a narrow range of predictive capabilities. Approximation caching may also suffer from large update message transmission when many sensor readings change dramatically. Our approach further exploits the temporal correlation by utilizing a linear trend component which enhances the estimation capability of our approach.

Several works aim at extracting data from sensor networks by using statistical models to capture the correlations of sensor readings, such as BBQ [15]. BBQ samples a small fraction of the sensor data from the network and utilizes a correlation model to estimate the non-sampled sensor readings. BBQ tries to find the best subset of sensors to sample by using a greedy algorithm. Generally, these models require an expensive long training phase and their correctness cannot be guaranteed, since they are base on the hypothesis that the statistical model obtained during

the training phase is similar to that during the operation phase. Actually, this hypothesis cannot be satisfied in long-term continuous data gathering. Our approach continuously checks the correctness of the estimated values and automatically adjusts its parameters according to the updates of the sensor readings. This makes our approach competent for long-term continuous data gathering applications.

Distributed source coding is a losslessly compression technique trying to addresses the problem of losslessly compressing correlated sources that are not co-located and cannot communicate with each other to minimize their joint description costs. In [17], 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 between the attributes, and will return wrong answers (without warning) if this condition is not satisfied. In practice, the cost of acquire precise and perfect knowledge of the correlations between the attributes is extremely high. Our approach smartly utilizes a simple probability model to depict spatial correlations between sensor nodes at cluster heads based on rough data from other sensor nodes.

Another technique that is widely used to reduce communication cost in sensor networks is called time-series forecasting. In [19], Lazaridis and Mehrotra 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 [20], Chatterjea and Havinga describe an adaptive sensor sampling scheme where nodes change their sampling frequencies autonomously based on time-series forecasting, in order to reduce energy consumption. They use time-series forecasting to predict the future sensor readings. The sampling frequency decrease against prediction accuracy,otherwise increase the sample frequency. The skipped samples are replaced by prediction values. Compared with [20], our approach can make sure the error of the gathered data is below the application specified threshold.

In [1], sink node uses simple linear time series model that consists of a trend component and a stationary auto autoregressive component to predict the reading of each sensor . Each sensor node updates its linear time series model individually, without considering the similarity of sensor readings between each sensor node. Their approach ignores spatial correlations of nearby sensor nodes and cannot suppress the update messages of nearby sensors with similar sensor reading.

III. Local Estimation

In this section, we present a local estimation scheme to reduce the message cost of an individual sensor node and its cluster head. In this scheme, a sensor node can estimate a newly-generated reading through a data model learned from its historic data. The parameters of the data model are sent to the cluster head, rather than the 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 the data to the cluster head, hence the message cost is reduced. In the following part of this section, we present our data model used in this scheme, and then describe how to learn parameters and update the data model. The used notations in this paper are summarized in Table 1, and the computation procedures on the cluster heads and the sink are discussed in the next section.

Table I
Used notations

Notation	Meaning
$\mathbb{F}$	The sensor readings of the whole network
$\mathbb{P}$	The 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
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 Gaussian white noise at s_i
T	The time interval of sampling
$\chi_i(t)$	A weakly stationary autoregressive component at s_i
$\mathbb{S}$	A set of partitions of all the cluster heads
G_i	A partition of cluster C_i
W_i	The i^{th} Θ-similar set
$g_{i,j}$	A Θ-similar set that belongs to G_i
$D_{ij}(t)$	The estimation distance between s_i and s_j at time t
r_i	The radius of Θ-similar set W_i
Δ	The error bound depends on applications
w_i	The predictor of W_i

A. Data Model

A sensor network consists of a collection of n sensor nodes $S = \{s_1, s_2, \ldots, s_n\}$ and a sink node. The data generated by the whole sensor network 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 $v_i(0), v_i(1), v_i(2), \ldots$ 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 multi-hop path. Each cluster head processes data from sensor nodes in the cluster, and sends the result to the sink through a multi-hop 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 $P_i = p_i(0), p_i(1), p_i(2), \ldots$ for $1 \leq i \leq n$. $\forall i, t, |v_i(t) - p_i(t)| \leq \Delta$.

We use the model proposed in [1] to estimate the sensor readings. The reading $v_i(t)$ generated by sensor node s_i at time t can be 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 3-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:

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

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 Gaussian white noise $N(0,1)$. 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:

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

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 [1].

Lemma 1. *Let $\epsilon_i = k\delta_i$, where k is a 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 $1/k^2$.*

B. 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 [2]. 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 variance of the white noise can be computed using the estimation error $e_i(t)$ by the following equation:

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

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 the sensor nodes in this cluster.

C. Estimation Updating

In order to maintain the accuracy of the estimation model, each sensor node periodically checks its readings and updates its local estimation model when its sensor readings consistently diverge from its current model. By Lemma 1, the estimation error measured in absolute value exceeds ϵ_i with probability less than $1/k^2$. If this happens for several times continuously, the current model probably cannot fit for the newly-generated readings. Specifically, if the generated sensor reading continuously falls outside $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ for three times, the sensor node relearns its local estimation model using the last N readings in queue Q. Otherwise, we consider sensor readings outside $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ as outliers. According to Lemma 1, the false-estimation probability is less than k^{-6}.

In applying local parameter learning and estimation updating, the following lemma depicts that the relation between the real sensor reading $v_i(t)$ and the estimated value $p_i(t)$ maintained at the cluster head.

Lemma 2. *Let $v_i(t)$ be the real 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 $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability less than $2/k^2$.*

Proof: According to Lemma 1, the actual value $v_i(t)$ at time t is contained in $[p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability $P_1 < 1/k^2$ at the very beginning. When the sensor reading distribution changes, the probability, denoted by P_2, that our approach does not detect the sensor reading distribution change is less than $1/k^6$. Hence, the probability P_3 that $v_i(t) \notin [p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ is $P_3 = P_2 + (1 - P_2)P_1 = P_1 + P_2 - P_1P_2 < P1 + P2 < 2/k^2$. ■

IV. Data Approximation

Having the estimation models from the 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 some basic definitions, and then present our data approximation scheme in detail.

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

Since sensor readings of nearby sensor nodes within a short period are probably similar, we can use the local estimation value of a sensor node to estimate the readings of nearby sensor nodes. Lemma 3 provides the upper bound of the estimation error of sensor node s_i if we use the local estimation data of s_j to estimate the readings of s_i.

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

Proof: According to the definition, $E_{ij}(t) = |v_i(t) - p_j(t)| = |v_i(t) - p_i(t) + p_i(t) - p_j(t)| \leq |v_i(t) - p_i(t)| + |p_i(t) - p_j(t)| = |v_i(t) - p_i(t)| + D_{ij}(t)$. By Lemma 2, $v_i(t) \in [p_i(t) - \epsilon_i, p_i(t) + \epsilon_i]$ with error probability less than $2/k^2$. Therefore, it is easy to see that the estimation error $E_{ij}(t) \in [0, \epsilon_i + D_{ij}(t)]$ with error probability less than $2/k^2$. ■

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

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

Definition 4. *The predictor of Θ-similar set W_i at time t, $w_i(t)$, is the local estimation value of W_i's representation node s_j.*

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

Lemma 4. *The estimation error $E_{ir_j}(t) \in [0, \Delta]$ with error probability less than $2/k^2$, if the radius of Θ-similar set W_j is less than Δ.*

Lemma 4 provides an approach to obtain a Δ-loss approximation of the sensor readings in any given Θ-similar set. Let G_i be a partition of cluster C_i, denoted as $G_i = \{g_{1,i}, g_{2,i}, \ldots, g_{x,i}\}$, where $g_{j,i}$ is a cell of G_i. Now we can obtain a predictor set of C_i, referred by $H_i = \{w_{1,i}, w_{2,i}, \ldots, w_{x,i}\}$, where $w_{j,i}$ is the predictor of $g_{j,i}$. For any sensor network, we can obtain a set $\mathbb{S} = \cup H_i$. By Lemma 4, H_i is a Δ-loss approximation of the data generated by cluster C_i at time t. It is easy to see that $\mathbb{S}$ is a Δ-loss approximation of $\mathbb{F}$. In order to get such a Δ-loss approximation, 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 a sensor network at time t, There exists $\mathbb{S}$ which is a Δ-loss approximation of $\mathbb{F}(t)$.*

Our data approximation scheme is based on Theorem 1 and consists of two algorithms: data approximation learning algorithm and data approximation monitoring algorithm. In this scheme, each cluster head maintains a data approximation model to process the uploaded messages of local estimation.

The learning algorithm. The data approximation learning phase starts after the local parameter learning phase ends. By Theorem 1, each cluster head h_i only needs to send H_i 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 $|H_i|$. Therefore, at the end of this phase, the number of messages generated by all cluster heads is $|\mathbb{S}|$. Since $\min|\mathbb{S}| = \min\cup|H_i| = \cup\min|H_i|$, the problem of minimizing $|\mathbb{S}|$ can be converted to minimizing each $|H_i|$.

Algorithm 1 Finding A Dominating Set of $\mathbb{G}_i(V, E)$

```
1:  i = 0;
2:  while |V| > 0 do
3:      v=FindLargestOutDegree(V);
4:      H[i].representation node=v;
5:      H[i].similarity set=AllNeighbor(v);
6:      V− = {v};
7:      V− =H[i].similarity set;
8:      i++;
9:  end while
10: return H;
```

For arbitrary cluster C_i, we construct a directed graph $\mathbb{G}_i = (V, E)$. Each vertex in $\mathbb{G}_i$ represents a sensor node in C_i, and $e_{i,j} \in E$ is a direct edge from s_i to s_j. $e_{i,j}$ exists if and only if $(\epsilon_i + D_{ij}(t)) \leq \Theta$, where $\Theta = \Delta$. Since H_i is a dominating set of $\mathbb{G}_i$, the problem of minimizing $|H_i|$ can be transformed into finding the minimum dominating set of the directed graph, which is known to be NP-hard [3]. An approximate minimum dominating set can be obtained in $O(|V|^2)$ time using the greedy algorithm proposed in [4]. The greedy heuristic algorithm finds a Θ-similar set in each iteration and stops when all nodes are removed from V (Algorithm 1). In each iteration, the cluster head finds a node v with largest out degree and sets v as the representation node of W_i (line 3-4), and then adds all neighbors of v to W_i (line 5). Then, all the nodes in W_i are removed from vertex set V (line 6-7). The algorithm stops when V is empty. At the end of this phase, each cluster head sends its predictor set H_i to the sink.

The monitoring algorithm. Our data approximation monitoring algorithm makes sure that the predictor $\mathbb{S}$ is a Δ-loss approximation. Each cluster head starts the data approximation monitoring algorithm immediately after the data approximation learning algorithm ends. During this phase, each cluster head checks the estimation errors of its Θ-similarity sets every T seconds. If the error of any set exceeds Δ, the cluster head will adjust its local Θ-similarity sets and send the changes to the sink node. The sink node then updates $\mathbb{S}$ accordingly.

The details of the monitoring algorithm are illustrated in Algorithm 2. The algorithm first updates all local estimations at the cluster head by invoking procedure UpdateMessagePrc(M) (line 1), in which M are the local estimation update messages. Line 2-12 search each Θ-

Algorithm 2 Monitoring at A Cluster Head

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

similar set at the cluster head and find out all sensor nodes that are no longer Θ-similar to their representation nodes, then add them into $\mathbb{C}$. The algorithm removes all empty Θ-similar sets. Each sensor node in $\mathbb{C}$ tries to find a Θ-similar set to join in by invoking the procedure Join(), as shown in 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(), as shown in line 16. Line 20 sends the update messages to the sink.

There are two kinds of messages sent from a cluster head to the sink: Θ-similar set creating message and 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 there is no need to explicitly send a message to remove a sensor from a Θ-similar set, because no sensor node can belong to two or more Θ-similar sets simultaneously. Adding a node into a Θ-similar set means removing it from another set.

The details of update message processing algorithm for the sink are shown in Algorithm 3. After receiving an updating message, the sink node first checks its message type. If it is a Θ-similar set creating message M, it first removes all the nodes listed in M from current Θ-similar sets, then invokes CreatNewSet() to create a new Θ-similar set and adds all these nodes listed in M into the set (line 2-4). If M is a Θ-similar set updating message, the sink node first removes all the nodes listed in M from current Θ-similar sets, then invokes SetUpdate() to update the predictor of the specified Θ-similar set or add all the node listed in M into the specified Θ-similar set (line 6-9).

Algorithm 3 UpdateMsgProcessing(M)

```
1: if msgtype is Θ-similar set creating message then
2:    Remove(M);
3:    g =CreatNewSet(M);
4:    G = G ∪ {g};
5: end if
6: if msgtype is Θ-similar set updating message then
7:    Remove(M);
8:    SetUpdate(M);
9: end if
```

V. Performance Evaluation

In this section, we present an extensive performance evaluation of our approximation data collection approach using real-world data. Our goal is to demonstrate that the proposed approximation data collection approach ADC can notably reduce message cost compared with SAF [1].

A. Experimental Setup

We conduct trace-driven simulations to evaluate the performance of our scheme using the data traces collected from a real-world GreenOrbs deployment. We deploy 88 TelosB sensor motes in a garden and collect sensor readings of temperature generated every thirty seconds for 10 hours at night. We also collect the topology information of our sensor network, including the neighbor sets of the sensor nodes and the packet loss rates of the links.

Note that the used data set includes a large number of missing readings due to unreliable wireless multi-hop 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 8 sensor nodes have 100% packet loss rate, 8 sensor nodes have packet loss rate lager than 90%, and 4 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 and we discard the links with packet loss rates larger than 90%, so that the routing can be built on relatively reliable links. 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.

B. Message Cost and Data Error

We evaluate the performance of ADC in terms of total message cost and data error, and compare ADC with SAF [1], which aims reducing message cost by exploiting time-series forecasting techniques. In the simulations, the length of the learning phase in SAF and that in ADC are set to 10 minutes.

Message Cost. We begin with investigating the message costs of the two approaches, which is defined as the sum of the messages sent by the sensor nodes. As shown in Figure 2, the message costs of ADC and SAF decrease against the

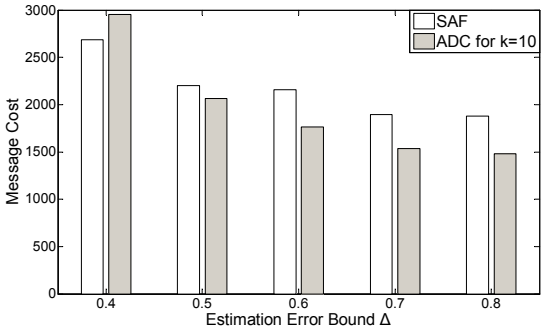

Figure 2. Message cost v.s. estimation error bound Δ.

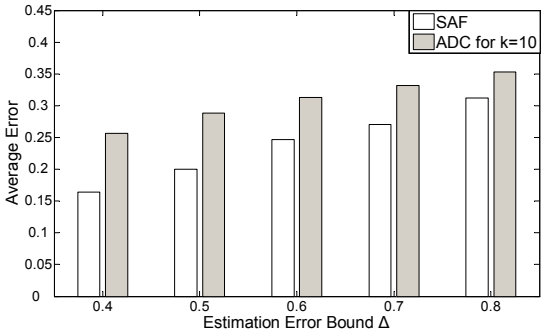

Figure 3. Average error v.s. estimation error bound Δ.

error bound Δ, and both the two decreasing trends become smooth as Δ increases. Compared with SAF, ADC has less message costs when $\Delta > 0.4$, and ADC achieves more message saving as Δ increases. When $\Delta = 0.8$, the message cost of ADC is only about 79% of that of SAF.

It should be noted that when $\Delta = 0.4$, the message cost of ADC is bigger than that of SAF. The main reason lies in that ADC brings in exorbitant message cost on informing the sink 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. Hence 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 radius, because its expected estimation error is more likely to exceed the given error bound Δ. This increases the number of messages required to inform the sink node the updates of the Θ-similar sets.

Data Error. Next we compare the data errors (measured in absolute value) introduced by SAF and ADC. Figure 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 the readings of a sensor node and that of its representation node. Since the radius of the Θ-similar sets is set to Δ in ADC,

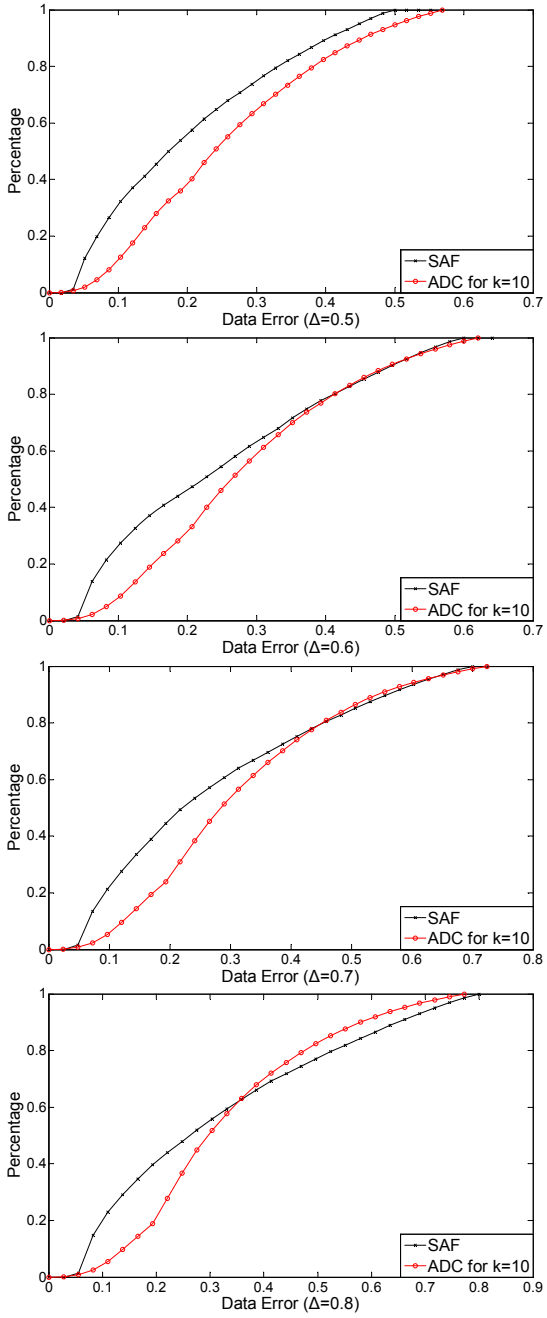

Figure 4. CDFs of the data errors of SAF and ADC for varying Δ.

the increase of Δ allows the Θ-similar sets to contain sensor nodes with larger estimation errors. As a consequence, the data error of ADC increases with Δ, as shown in Figure 3. Although the average data error of ADC is larger than that of SAF, the difference between the two is very small: it is always less than $0.1°C$ and decreases slowly against Θ.

Figure 4 depicts the CDFs of data error in the two approaches 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 Figure 3. These results imply that ADC achieves efficiency in message cost at the expense of data error, and more importantly, the error bound can be adjusted by users.

VI. Conclusion

In this paper, we propose a novel approximate data collection strategy ADC in wireless sensor networks. ADC can approximate the readings of the whole 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% compared with existing works. In the future, we plan to implement and evaluate our work in real sensor networks.

Acknowledgment

The work reported in this paper was supported by the National Natural Science Foundation of China under Grant No.60833009,the National Science Funds for Distinguished Young Scientists under Grant No.60925010, and the National Basic Research Program of China (973 Program) under Grant No.2011CB302701.

References

[1] D. Tulone, S. Madden. An Energy Efficient Querying Framework in Sensor Networks for Detecting Node Similarities, in *Proc. of ACM MSWiM*, 2006.

[2] G. Box, G. M. Jenkins. Time Series Analysis: Forecasting and Control. Prentice Hall, 1994.

[3] S. Howe. Dominating Sets of Random 2-in 2-out Directed Graphs. The Electronic Journal of Combinatorics, 15(29), 2008.

[4] P. AK. Analysis of a Greedy Heuristic for Finding Small Dominating Sets in Graphs. Information Processing Letters, 39(5), 1991, 237–240.

[5] 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, in *Proc. of ACM SENSYS*, 2005.

[6] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and Yield in a Volcano Monitoring Sensor Network, in *Proc. of OSDI*, 2006.

[7] Y. L. Borgne, S. Santini and G. Bontempi. Adaptive Model Selection for Time Series Prediction in Wireless Sensor Networks. Signal Processing. Elsevier, 87(12), 2007, 3010–3020.

[8] C.Olston and J.Widom. Best Effort Cache Sychronization with Source Cooperation, in *Proc. of ACM SIGMOD*, 2002.

[9] C.Olston, J.Jiang, J.Widom. Adaptive Filters for Continuous Queries over Distributed Data Streams, in *Proc. of ACM SIGMOD*, 2003.

[10] I. Lazaridis, S. Mehrotra. Capturing Sensor-Generated Time Series with Quality Guarantee, in *Proc. of IEEE ICDE*, 2003.

[11] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks, in *Proc. of ACM MobiCom*, 2000.

[12] S. Madden, W. Hong, J. M. Hellerstein, and M. Franklin. TinyDB web page: http://telegraph.cs.berkeley.edu/tinydb.

[13] Y. Yao and J. Gehrke. Query Processing in Sensor Networks, in *Proc.CIDR*, 2003.

[14] A. Silberstein, R. Braynard and J. Yang. Constraint Chaining: on Energy-effcient Continuous Monitoring in Sensor Networks, in *Proc. of ACM SIGMOD*, 2006.

[15] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein and W. Hong. Model-driven Data Acquisition in Sensor Networks, in *Proc. of VLDB*, 2004.

[16] D. Tulone, S. Madden. PAQ: Time Series Forecasting for Approximate Query Answering in Sensor Networks, in *Proc. of EWSN*, 2006.

[17] D. Slepian and J. Wolf. Noise Less Coding of Correlated Information Sources. IEEE Transactions on Information Theory, 19(4), 1973.

[18] A. D. Wyner and J. Ziv. The Rate-distortion Function for Source Coding with Side Information at the Decoder. IEEE Transactions on Information Theory, vol IT-22, 1976.

[19] I. Lazaridis and S. Mehrotra. Capturing Sensor-generated Time Series with Quality Guarantees, in *Proc. of IEE ICDE*, 2003.

[20] S. Chatterjea and P. Havinga. An Adaptive and Autonomous Sensor Sampling Frequency Control Scheme for Energy-Efficient Data Acquisition in Wireless Sensor Networks, in *Proc. of DCOSS*, 2008.

[21] Moteiv. Telos Revb data sheet, December 2004. http://www.moteiv.com/pr/2004-12-09-telosb.php.

[22] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X-Y. Li, G. Dai. Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest, in *Proc. of ACM SenSys*, 2009.

[23] Chu D, Deshpande A. Hellerstein J M., Hong W. Approximate Data Collection in Sensor Networks Using Probabilistic Models, in *Proc. of IEEE ICDE*, 2006.

[24] The GreenOrbs project. http://www.greenorbs.org.