Exploiting Ubiquitous Data Collection for Mobile Users in Wireless Sensor Networks

Zhenjiang Li, *Member*, *IEEE*, Yunhao Liu, *Senior Member*, *IEEE*, Mo Li, *Member*, *IEEE*, Jiliang Wang, *Member*, *IEEE*, and Zhichao Cao, *Member*, *IEEE*

Abstract—We study the ubiquitous data collection for mobile users in wireless sensor networks. People with handheld devices can easily interact with the network and collect data. We propose a novel approach for mobile users to collect the network-wide data. The routing structure of data collection is additively updated with the movement of the mobile user. With this approach, we only perform a limited modification to update the routing structure while the routing performance is bounded and controlled compared to the optimal performance. The proposed protocol is easy to implement. Our analysis shows that the proposed approach is scalable in maintenance overheads, performs efficiently in the routing performance, and provides continuous data delivery during the user movement. We implement the proposed protocol in a prototype system and test its feasibility and applicability by a 49-node testbed. We further conduct extensive simulations to examine the efficiency and scalability of our protocol with varied network settings.

Index Terms—Wireless sensor networks, data collection, mobile user

1 Introduction

Past several years have witnessed a great success of Wireless Sensor Networks (WSNs). Recent advances in Wireless Sensor Network technologies provide people the ability to better understand the physical world. With the data collected from the entire network, the sensor network supports a variety of applications, including security surveillance [1], [2], [3], localization [4], information enquiry, and transmission [5], [6], [7], [8], etc. In this paper, we consider the ubiquitous data collection by mobile users in the wireless sensor network. Mobile users are equipped with handheld devices that communicate with sensor nodes and instantly access the network through nearby sensors. Such a pervasive usage of sensor networks explores in-situ interactions with human beings, provides people facilitated means of data collection, and thus significantly expands the capability of wireless sensor networks.

A typical application that we have envisioned is the forest surveillance. In the GreenOrbs project [9], more than 300 sensor nodes are deployed in Tianmu Mountain to collect scientific data of the forest, such as temperature, humidity, concentration of carbon dioxide and so on. On the other hand, there are a number of forest rangers patrolling around the mountain to detect any accidents in the forest, like the fire indication, the vegetation damage, etc. Equipping the rangers with communicational devices and enabling them collect the field data of interest from the sensor network *anywhere* and *anytime* would largely benefit their work (as illustrated in Fig. 1).

The ubiquitous data collection problem considered in this paper essentially differs from traditional data collection problems in static settings. In a static sensor network, an optimal data collection tree is usually built to collect the network-wide data. The data collection tree is fixed and suffices to efficiently deliver data to the static sink [10], [11], [12], [13], [14], [15]. In the presence of user mobility and the requirement of ubiquitous data access, however, the data collection tree constructed at one point is normally not enough as the mobile user moves. To efficiently deliver network-wide data to the mobile user, the data collection tree needs to be *constructed* or *updated* from time to time according to the mobile user's movement. Directly adopting traditional data collection paradigm results in building a series of independent data collection trees when the mobile user is at different positions. Unveiled by Kusy et al. [16], building the data collection tree introduces a large volume of communication overheads. Besides, the routing transitions between different data collection trees contain a nonnegligible time delay and may lead to discontinuity or even loss of the data delivered to the mobile user, which significantly decreases the QoS of ubiquitous data collection.

There have been attempts made to efficiently deliver data to mobile users in wireless sensor networks. Most existing works, however, assume that the mobile user has a planned mobility path or the path can be accurately predicted, such that a variety of schemes can be applied to compensate the time cost of the data collection transitions [16], [17]. None of those works focus on essentially optimizing the routing transitions, with reduced transition

- *Z. Li is with the School of Computer Engineering, Nanyang Technological University, N4-B2A-03, 50 Nanyang Avenue, 639798 Singapore. E-mail: lzjiang@ntu.edu.sg.*
- *Y. Liu is with the School of Software and TNLIST, Tsinghua University, and Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail:yunhao@greenorbs.com.*
- *M. Li is with the School of Computer Engineering, Nanyang Technological University, N4-02C-108, 50 Nanyang Avenue, 639798 Singapore. E-mail: limo@ntu.edu.sg.*
- *J. Wang is with the School of Software, Tsinghua University, Beijing, China, 100084. E-mail: jiliang@greenorbs.com.*
- *Z. Cao is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: caozc@cse.ust.hk.*

Manuscript received 1 May 2011; revised 18 Dec. 2011; accepted 24 Feb. 2012; published online 9 Mar. 2012.
Recommended for acceptance by W. Jia.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2011-05-0264.
Digital Object Identifier no. 10.1109/TPDS.2012.92.

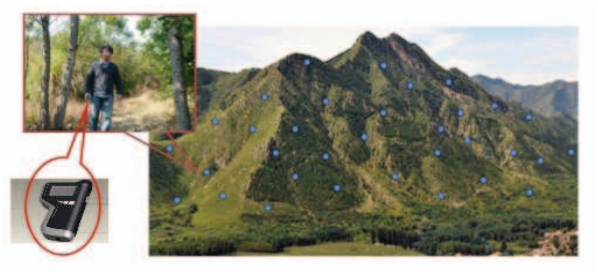


Fig. 1. Ubiquitous data collection in Tianmu mountain.

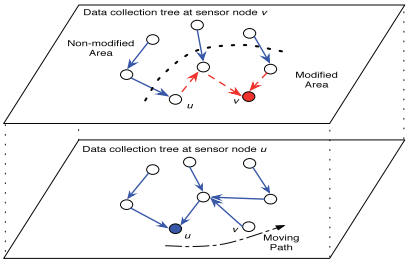


Fig. 2. Spatial correlation between $\mathcal{T}_u$ and $\mathcal{T}_v$.

overhead, continuous data delivery, and facilitated data collection for mobile users with unlimited mobility paths.

In this paper, we observe that there exist strong spatial correlations among routing structures at different positions, and take advantage of such an observation to additively update the routing structure with the user's movement. The contributions of this work are as follows: First, we propose an additive approach that updates the data collection tree. In particular, through a limited modification of existing data collection tree in the network, a new collection tree can be constructed in a lightweight manner in terms of time efficiency and overheads. Moreover, the proposed approach is easy to implement and the resulting routing performance on the new collection tree is bounded and controlled with regard to the optimal value. Second, the proposed approach in this work supports delivering continuous data streams even with routing transitions. When the mobile user moves within the sensor network, the data collection tree keeps updated to stream the unreceived data toward the mobile user. Such a property ensures a low data collection delay, providing a real-time data acquisition for the mobile user. Third, we implement a prototype system in a 49 Telos Mote testbed. The experiment results validate the feasibility and applicability of the proposed approach in practice. We further conduct extensive and large scale simulations to examine the efficiency and scalability of our protocol. Compared to existing approaches, we achieve efficient data collection with highly reduced network overheads.

The rest of this paper is organized as follows: the preliminary is presented in Section 2. We introduce our ubiquitous data collection protocol and related properties in Section 3. In Section 5, we implement the proposed protocol in a 49-node testbed and further examine its performance by simulation in Section 6. In the end, we conclude this paper in Section 8.

2 Preliminary

In this section, we formally formulate the problem considered in this paper. In addition, we present assumptions and requirements of the system performance in this section as well.

2.1 Motivation

We consider the problem of ubiquitous data collection by the mobile user in a wireless sensor network. The mobile user uses a handheld device to communicate with sensor nodes in the network. The mobile user roams within the network and instantly accesses network-wide data on a need basis. For instance, in the aforementioned GreenOrbs project, the mobile user needs to collect data from the entire network in many scenarios, such as the periodical measurement of concentration of carbon dioxide for the scientific analysis, the network abnormal event detection, etc.

We claim that the ubiquitous data collection problem studied in this paper is essentially different from conventional mobile sink-based data collection [18], [19]. In previous works, the sink mobility has been introduced to benefit the data collection operation, e.g., to improve the network lifetime. The mobile sink is essentially cooperative to the data collection. However, in this study we explore the pervasive usage of sensor networks to support the in-situ interactions with human beings. Although the data collection and the movement may still coexist in our problem, they are two separate actions without cooperation. Referring to the aforementioned example, the data collection is for the mobile user to understand the forest situation and the movement is for the user to reach the destination. We do not restrict the moving trajectory to facilitate the data collection; meanwhile, we do not constrain the scope of the data collection to fasten the movement either.

In this paper, we aim to develop an efficient routing transition scheme such that the data collection tree can be carefully maintained and updated as the movement of the mobile user. We provide an example in Fig. 2 to introduce our design principle. Suppose that at sensor node u the mobile user collects data for the first time. After one of existing data collection tree formation protocols, like CTP [10], Information potential [12] or ICTP [11], is performed, a routing structure (denoted as $\mathcal{T}_u$) will be constructed, based on which the mobile user can collect the network-wide data as shown by the bottom scenario in Fig. 2. Once accessing enough data at node u, the mobile user will stop collecting data and continue patrolling in the network. In the simplest application scenario, each mobile user collects the data from the entire network at each place. As we will further elaborate in Section 4, benefiting from the data streaming property, the data collection can be achieved during user movement. Furthermore, we assume that the mobile user accesses sensory data for the second time at sensor node v. At this new data collection position, another routing structure is needed. Parallel to the routing tree formation process at node u, $\mathcal{T}_v$ can be built as shown by the top scenario in Fig. 2. Similarly, as the mobile user keeps patrolling, plenty of routing trees need to be built at different positions. As unveiled by recent literatures, the data collection tree formation is a time and energy consuming process. If traditional tree formation schemes are applied for the mobile user to conduct ubiquitous data collection, a series of independent routing trees must be constructed. Actually, through our study, we find that it is unnecessarily needed.

Revisiting Fig. 2, we can observe that $\mathcal{T}_u$ and $\mathcal{T}_v$ are not completely independent. On contrary, they share certain number of common edges. In other words, there exists a *spatial correlation* between them. Through our study, we find that $\mathcal{T}_v$ can be quickly formed based on $\mathcal{T}_u$ by taking advantage of their spatial correlation. As shown in Fig. 2, by conducting a local updating on $\mathcal{T}_u$, $\mathcal{T}_v$ can be formed. Clearly, such an updating operation influences only a local area. Hopefully, 1) lightweight communication and energy costs, 2) short routing tree formation delay, and 3) the overall data collection delay can be achieved. In the rest of this paper, we will specify how to utilize such a design principle to approach an efficient ubiquitous data collection for the mobile user in WSNs.

2.2 Assumptions

To facilitate our discussion, we make the following assumptions in this paper:

- The mobile user carries a handheld device like 802.15.4 compatible PDA that communicates with sensor nodes.
- One sensor node within the communication radius of the mobile user is designated as a *virtual sink*. The network-wide data are first delivered to the virtual sink and then sent to the mobile user via a direct communication.
- Throughout this paper, we use the network hop distance as an indication of the routing path quality for simplicity of presentation. The proposed protocol in this paper, however, is still valid if we use other routing metrics, like the expected transmissions (ETX) [10], to build the data collection tree.

2.3 Performance Requirements

The objective of this work is to build the data collection tree and additively update it for the mobile user to access the network data ubiquitously. In addition, several requirements need to be satisfied:

- We require that the maintenance of the routing structure is **scalable**, i.e., the update of the data collection tree for the routing transition should be *limited* and *distributed*.
- We require that the resulting data collection process is **efficient**, i.e., in the data collection tree, the data delivery path from an arbitrary sensor node to the virtual sink should not be excessively long. Compared with the optimal routing path, the data delivery path length in our approach should be *bounded* and *controlled*.
- We require that the data collection process is **fluent**, i.e., the routing structure should fluently deliver data toward the mobile user even though there exist routing transitions due to the user's mobility.
- We require that the system is **compatible** with mobility prediction mechanisms, i.e., existing mobility prediction techniques can be seamlessly integrated in our approach to further improve the performance.

From the next section, we will introduce our proposed protocol to satisfy such performance requirements.

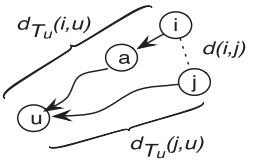


Fig. 3. Illustration of $d(i,j)$ and $d_{\mathcal{T}_u}(i,j)$ (Scenario one).

3 System Design

We elaborate the design of our protocol. The main idea of our protocol is utilizing the spatial correlation to efficiently build and update the data collection tree. Whenever the mobile user moves and changes the virtual sink to access the network, a new data collection tree can be efficiently formed by locally modifying the previously constructed data collection tree in the network. Based on such an observation, in the following section, we present the design details of three components in our protocol: 1) *Data Collection Tree Initialization*, 2) *Data Collection Tree Updating*, and 3) *Data Routing*.

3.1 Data Collection Tree Initialization

We consider the entire wireless sensor network as a graph $G = \{V, E\}$, where the vertex set V represents the static sensors and the edge set E represents the communicational links. Without loss of generality, the initial virtual sink is denoted as $u \in V$, through which the mobile user accesses the network-wide data at the beginning.

There have been many research studies proposed for constructing a global data collection tree for a given sink node [10], [11], [13], [14], [15]. Similar to these existing schemes, *Data Collection Tree Initialization* in our protocol is realized by the flooding control in an iterative manner, like [10]. More precisely, an optimal routing tree can be formed as follows: The sink node launches the routing tree construction by broadcasting a control message and the initial value of the communication cost[1] to the sink node at each sensor side is set to be infinity. In general, by exchanging information, sensor i configures H_i to be the neighbor with the minimum cost to the sink compared with all other neighbors, where H_i is the child node of sensor i in the routing tree, i.e., sensor i only transmits or relays packets to sensor H_i. Once H_i is updated, sensor i will inform its neighbors and the neighbors can update their own configurations accordingly.

To facilitate the presentation, the data collection tree formed in the initialization phase is denoted as $\mathcal{T}_u$. After $\mathcal{T}_u$ has been formed, the mobile user can collect data through the virtual sink u. In addition, each sensor (e.g., sensor i) is required to record its distance to the virtual sink u in $\mathcal{T}_u$, denoted as $d_{\mathcal{T}_u}(i,u)$. $d_{\mathcal{T}_u}(i,u)$ can be obtained during the construction of $\mathcal{T}_u$. After $\mathcal{T}_u$ has been formed, the distance between any two sensors, e.g., sensors i and j will be defined in two ways. $d(i,j)$ denotes the minimum distance between i and j, while $d_{\mathcal{T}_u}(i,j)$ indicates the distance between i and j constrained by $\mathcal{T}_u$. We define $d_{\mathcal{T}_u}(i,j) \triangleq d_{\mathcal{T}_u}(i,u) + d_{\mathcal{T}_u}(j,u)$, as shown in Figs. 3 and 4. One point we want to emphasize is that such a distance definition between any two nodes i and j

1. The communication cost can be the hop-count distance, the path ETX aggregates [10], the information potential [12], the ICTP [11], etc.

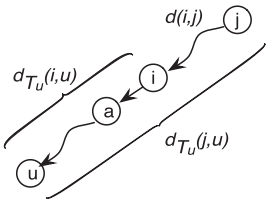


Fig. 4. Illustration of $d(i,j)$ and $d_{\mathcal{T}_u}(i,j)$ (Scenario two).

is an estimation of their real distance constrained by $\mathcal{T}_u$. We prefer such an estimation due to its simplicity and efficiency. As shown in the next section, sensor nodes i and j can efficiently estimate their mutual distance on $\mathcal{T}_u$ without extra communication overhead. Note that the protocol proposed in this paper is not limited to the distance definition given in this section. It is generally compatible to any feasible distance metrics.

3.2 Data Collection Tree Updating

The mobile user keeps moving around and the virtual sink that connects the user to the sensor network changes accordingly. When the mobile user moves away from the original virtual sink u and designate a new virtual sink v, a new data collection tree at virtual sink v must be constructed, namely $\mathcal{T}_v$. A natural solution is to reconstruct $\mathcal{T}_v$ with the same process of building $\mathcal{T}_u$, i.e., *Data Collection Tree Initialization* can be relaunched to form $\mathcal{T}_v$. Some alternative updating methods are proposed as well [12]. Nevertheless, most existing works presume that the new collection tree shall preserve the optimality and they take a long time delay and heavy communication overheads to achieve such a goal. As a result, a series of optimal routing trees are built as the mobile user moves around in the network.

In this study, however, we find that it is not necessary to form an optimal routing tree at every data collection point. The basic idea of our approach is that, when we build a new data collection tree from the new virtual sink v, we do not update the routing paths for all the sensors over the network. As a matter of fact, once we reverse the path direction between u and v in $\mathcal{T}_u$, all the sensor nodes can reach the new virtual sink through the routing paths on the original data collection tree, and a large portion of those paths are of reasonable lengths compared with the optimal paths. We use a threshold λ to quantify such an effect and only update certain sensor nodes whose original routing paths are excessively longer than the optimal ones. In such a way, we are able to flood a small limited area to update part of the data collection tree and guarantee the routing performance. Later, we will show that although the routing paths on the new data collection tree are sub-optimal, their length distortions are bounded and controlled compared to the optimal ones. By doing so, the time and communication costs of building a data collection tree can be significantly reduced. The lightweight communication cost results in less energy consumption of sensors and the rapid updating process leads to a fluent routing transition. To formally describe our protocol, we introduce several notations at the first:

- λ is the user defined threshold in Algorithm 1, where $\lambda > 1$.
- u is the first virtual sink selected by the mobile user.
- $\mathcal{T}_i$ is the routing tree formed at virtual sink i.
- H_i is the child node of sensor i in the routing tree, i.e., sensor i always transmits or relays packets to sensor H_i.
- $d(i,j)$ is the minimum distance between sensors i and j.
- $d_{\mathcal{T}_k}(i,j)$ is the distance between sensors i and j in $\mathcal{T}_k$.
- $EST_{\mathcal{T}_k}(i,j)$ is the minimum distance from i to j in $\mathcal{T}_k$ known so far, which will be used in the routing tree updating process.

An instrumental explanation of our proposed protocol is as follows: The protocol is generally triggered by a serials of flooding messages. One flooding message contains two types of information: 1) $d_{\mathcal{T}_u}(v,u)$ and 2) $EST_{\mathcal{T}_v}(j,v)$, where j is the sender of this message. Such a message is denoted as $M_j(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(j,v))$. If sensor i receives it, $d_{\mathcal{T}_u}(v,u)$ can be used to calculate the distance from i to v in $\mathcal{T}_u$, i.e., $d_{\mathcal{T}_u}(i,u) + d_{\mathcal{T}_u}(v,u)$, and $EST_{\mathcal{T}_v}(j,v)$ can be used to update $EST_{\mathcal{T}_v}(i,v) \leftarrow EST_{\mathcal{T}_v}(j,v) + d(i,j)$.

Algorithm 1. Limited Updating Algorithm at Sensor i

```
1: while Receiving a flooding message from sensor j do
2:   if EST_{T_v}(j,v) + d(i,j) < EST_{T_v}(i,v) then
3:     if (d_{T_u}(v,u) + d_{T_u}(i,u)) / (EST_{T_v}(j,v) + d(i,j)) > λ then
4:       EST_{T_v}(i,v) ← EST_{T_v}(j,v) + d(i,j)
5:       H_i ← j
6:       Flood M_i(d_{T_u}(v,u), EST_{T_v}(i,v)) to its neighbors
7:     else
8:       Discard M_j(d_{T_u}(v,u), EST_{T_v}(j,v))
9:     end if
10:  else
11:    Discard M_j(d_{T_u}(v,u), EST_{T_v}(j,v))
12:  end if
13: end while
```

Virtual sink v reverses the path $v \Rightarrow u$ (in $\mathcal{T}_u$) to $u \Rightarrow v$ first, and then launches the *Data Collection Tree Updating* process by broadcasting $M_v(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(v,v))$ to all its neighbors. Note that $EST_{\mathcal{T}_v}(v,v) = 0$ and the initial value of $EST_{\mathcal{T}_v}(i,v)$ for any $i \neq v$ equals $+\infty$. In general, after sensor i receives $M_j(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(j,v))$, sensor i calculates

$$\frac{d_{\mathcal{T}_u}(v,u) + d_{\mathcal{T}_u}(i,u)}{EST_{\mathcal{T}_v}(j,v) + d(i,j)}.$$

If $EST_{\mathcal{T}_v}(j,v) + d(i,j) < EST_{\mathcal{T}_v}(i,v)$ and

$$\frac{d_{\mathcal{T}_u}(v,u) + d_{\mathcal{T}_u}(i,u)}{EST_{\mathcal{T}_v}(j,v) + d(i,j)} > \lambda,$$

sensor i updates the value of $EST_{\mathcal{T}_v}(i,v)$ as $EST_{\mathcal{T}_v}(j,v) + d(i,j)$, changes H_i to j and broadcasts $M_i(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(i,v))$ to its neighbors; otherwise, sensor i simply discards the flooding message from neighbor j.

The rationale behind operations in Algorithm 1 can be interpreted as follows: $\frac{d_{\mathcal{T}_u}(v,u)+d_{\mathcal{T}_u}(i,u)}{EST_{\mathcal{T}_v}(j,v)+d(i,j)} > \lambda$ implies the distance of the routing path in the original routing tree from i to v is excessively long, i.e., the delay distortion is λ times greater than that of the optimal distance to v. In such a case, H_i is changed to j and sensor i will send data to j if there is any. As we will prove, all the sensors performing such an operation form a cluster $\mathcal{U}$, where

1. $\mathcal{U}$ is a limited region and the size of $\mathcal{U}$ is *reverse proportional* to λ;
2. following $H_i(i \in \mathcal{U})$, each sensor in $\mathcal{U}$ can reach[2] virtual sink v;
3. the routing path formed via Algorithm 1 within $\mathcal{U}$ provides excellent routing delay performance;
4. following the portion of $\mathcal{T}_u$ not modified by Algorithm 1, all the sensors outside $\mathcal{U}$ can reach virtual sink v through $\mathcal{U}$; and
5. the routing efficiency of each sensor outside $\mathcal{U}$ is bounded and controllable (i.e., the routing delay distortion is controlled by λ).

With above five properties, we can achieve a good balance between the cost of updating the data collection tree and the routing efficiency. In this section, we focus on the first four properties and the last one will be discussed in the next section.

Theorem 1. *The region $\mathcal{U}$ formed by Algorithm 1 is a bounded area, i.e., the influence of Algorithm 1 is limited.*

Proof. Suppose that a two-dimension coordinate system is embedded in the network and the previous virtual sink u sits at the point $(0, 0)$. We can always rotate the coordinate system such that the y-coordinate of the new virtual sink v is zero. We denote the coordinate of virtual sink v as $(l, 0)$, where $l \leq c \times d_{\mathcal{T}_u}(v, u)$ and c is the average physical distance between two neighboring nodes in the system. Now, we examine a sensor i (the coordinate is assumed to be (x, y)) whose H_i has been modified via Algorithm 1. According to line 3 in Algorithm 1, the boundary of region $\mathcal{U}$ is captured by the following:

$$\sqrt{x^2 + y^2} + l = \lambda \sqrt{(x - l)^2 + y^2}. \tag{1}$$

By substituting $r \cos\theta$ and $r \sin\theta$ for x and y, respectively, where $r \geq 0$ and $\theta \in [0, 2\pi)$, we can rephrase (1) as

$$r + l = \lambda \sqrt{(r\cos\theta - l)^2 + (r\sin\theta)^2}. \tag{2}$$

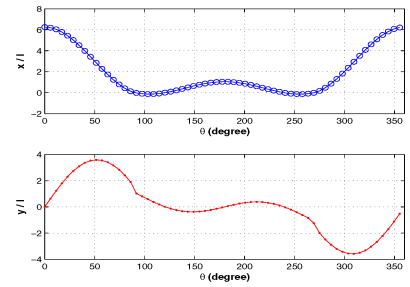
After solving (2), we get

$$\begin{cases} x = l \times \dfrac{(\lambda^2 \cos\theta + 1) \pm \sqrt{(\lambda^2 \cos\theta + 1)^2 - (\lambda^2 - 1)^2}}{\lambda^2 - 1} \times \cos\theta, \\ y = l \times \dfrac{(\lambda^2 \cos\theta + 1) \pm \sqrt{(\lambda^2 \cos\theta + 1)^2 - (\lambda^2 - 1)^2}}{\lambda^2 - 1} \times \sin\theta. \end{cases}$$

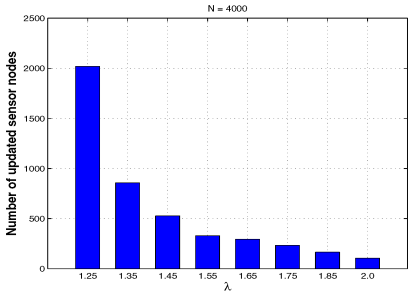
Based on the obtained expressions of x and y, it is not difficult to verify that both x and y are bounded. Therefore, the region $\mathcal{U}$ is a bounded area and the influence of Algorithm 1 is limited. □

In Fig. 5a, we provide some numerical results to facilitate understanding the bounded nature of both x and y derived in Theorem 1. Through our study, we find that various settings of λ produce a consistent conclusion. Without loss of generality, it is enough for us to check one setting of λ in this section, and how different λ settings impact other system performance will be discussed in detail later. In

2. Since $EST_{T_v}(i, v)$ is iteratively updated, H_i may switch multiple times during the execution of Algorithm 1. Nevertheless, we can prove that sensor i can always reach v through sensors in $\mathcal{U}$.



(a) Numerical results



(b) Simulation results

Fig. 5. Limited data collection tree updating.

Fig. 5a, λ and l are set to be 1.45 and 10, respectively. The maximum value of x can be achieved when θ is equal to 0. After some mathematical manipulation, we can obtain $x \leq l \times \frac{\lambda+1}{\lambda-1}$. The upper figure in Fig. 5a demonstrates that x is at most $5.4 (= \frac{1.45+1}{1.45-1})$ times longer than l indeed. On the other hand, although we hardly derive the exact expression for the maximum value of y, the lower figure in Fig. 5a reveals that the variance of y is small as well when θ changes from 0 to 2π, i.e., the ratio between y and l is no more than 4 in the experiment.

To further verify the conclusion made in Theorem 1, we also conduct a set of simulations and illustrate the results in Fig. 5b. Four thousand sensor nodes are uniformly deployed in the field at random. The first data collection tree is built at the left-bottom sensor node and the mobile user performs *Data Collection Tree Updating* at the center of the network. We vary λ from 1.25 to 2.0. For each λ, we run the simulation 50 times and demonstrate the average. Fig. 5b shows that there are 51 percent sensor nodes updated at most when λ is 1.25. When λ becomes larger, the updated area shrinks rapidly. Only less than 200 nodes are updated when λ is chosen to be 2. In Section 6, we will demonstrate the limited influence of our protocol by testbed as well. Another direct indication from the proof of Theorem 1 is that the size of region $\mathcal{U}$ is proportional to the distance between two virtual sinks. According to Theorem 1, the limited influence of Algorithm 1 ensures a rapid routing structure formation and light communication overheads. Due to the rapid transition of the mobile user, such a property is even more valuable and we will further verify this claim in Section 6.

Now, we will analyze the routing path length of each sensor node in the updating area $\mathcal{U}$. To facilitate our analysis, we introduce the following lemma first.

Lemma 1. *For any pair of sensors, $i, j \in \mathcal{U}$ and $j = H_i$, $\frac{d_{\mathcal{T}_u}(v,u)+d_{\mathcal{T}_u}(i,u)}{EST_{\mathcal{T}_v}(H_i,v)+d(i,j)} < \frac{d_{\mathcal{T}_u}(v,u)+d_{\mathcal{T}_u}(j,u)}{EST_{\mathcal{T}_v}(H_j,v)+d(j,H_j)}$.*

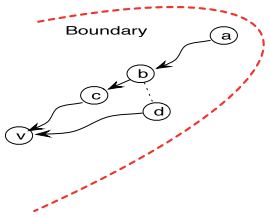


Fig. 6. Type-one sensor node in region $\mathcal{U}$.

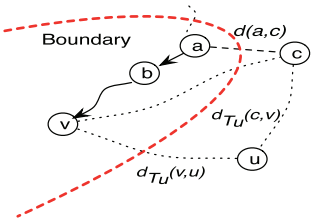


Fig. 7. Type-two sensor node in region $\mathcal{U}$.

Proof. Since sensor H_j is the child of sensor j, according to Algorithm 1, we obtain $EST_{T_v}(H_i, v) = EST_{T_v}(j, v) = EST_{T_v}(H_j, v) + d(j, H_j)$. Thus, we have

$$\begin{aligned}\frac{d_{T_u}(v,u)+d_{T_u}(i,u)}{EST_{T_v}(H_i,v)+d(i,j)} &= \frac{d_{T_u}(v,u)+d_{T_u}(i,u)}{EST_{T_v}(H_j,v)+d(j,H_j)+d(i,j)} \\ &\le \frac{d_{T_u}(v,u)+d_{T_u}(j,u)+d(i,j)}{EST_{T_v}(H_j,v)+d(j,H_j)+d(i,j)} \\ &< \frac{d_{T_u}(v,u)+d_{T_u}(j,u)}{EST_{T_v}(H_j,v)+d(j,H_j)}.\end{aligned}$$

The last inequality is based on the fact that if $a, b, c > 0$ and $a/b > 1$, $\frac{a+c}{b+c} < \frac{a}{b}$. □

To analyze the routing path length in $\mathcal{U}$, we need to check two types of sensors: 1) the entire optimal routing path to v is in $\mathcal{U}$ as shown by Fig. 6, and 2) the optimal routing path to v contains some sensors outside $\mathcal{U}$ as shown by Fig. 7.

We examine type-one sensors first. As shown in Fig. 6, we focus on sensor a in $\mathcal{U}$. In Fig. 6, the routing path formed by Algorithm 1 is $a \Rightarrow b \Rightarrow c \Rightarrow v$. In order to demonstrate this routing path being optimal, we can assume that the optimal path from a to v is $a \Rightarrow b \Rightarrow e \Rightarrow v$ instead of $a \Rightarrow b \Rightarrow c \Rightarrow v$. It is clear that minimum distance from b to v via e should be smaller than the minimum distance from b to v via c. In other words, we have

$$d(b,e) + d(e,v) < d(b,c) + d(c,v). \tag{3}$$

During the updating procedure of Algorithm 1, both nodes c and e will send flooding messages to node b to trigger its routing table updating (maybe multiple times). We have assumed that the optimal path from node b to v is completely within $\mathcal{U}$. The optimal path from node e to v should be in $\mathcal{U}$ as well since e is on the optimal path from b to v. $EST_{T_v}(e, v)$ will be identical to $d(e, v)$ after sufficient updating. Therefore, eventually, after H_b points to node e, $EST_{T_v}(b, v)$ is equal to the sum of $d(b, e)$ and $EST_{T_v}(e, v)$. Afterwards, based on (3), the condition in line 2 of Algorithm 1 cannot be satisfied, i.e., $EST_{T_v}(c, v) + d(c, b) \ge d(c, v) + d(c, b) > d(b, e) + d(e, v) = EST_{T_v}(b, v)$. Thus, H_b should point to e instead of c, which is a contradiction to that the routing path formed by Algorithm 1 is $a \Rightarrow b \Rightarrow c \Rightarrow v$. Hence, the original routing path is optimal.

Now, we examine type-two sensors. Note that type-two sensors are close to the boundary of region $\mathcal{U}$. The optimal path from one type-two sensor to the virtual sink v may pass through the boundary of $\mathcal{U}$ multiple times. It is sufficient for us to discuss a typical structure on the optimal path as shown in Fig. 7 where the neighbor c of a is outside $\mathcal{U}$. In Fig. 7, H_a is b but we assume that the optimal routing path to v passes $a \Rightarrow c$ (instead of $a \Rightarrow b$) and eventually reaches v. In this scenario, since sensor node a has been updated by Algorithm 1 (this is why node a is in $\mathcal{U}$), we denote *condition one* to be $\mathrm{I} \triangleq \frac{d_{T_u}(v,u)+d_{T_u}(a,u)}{EST_{T_v}(b,v)+d(a,b)} > \lambda$. On the other hand, as sensor node c has not been updated by Algorithm 1, we can similarly denote *condition two* as

$$\mathrm{II} \triangleq \frac{d_{T_u}(v,u)+d_{T_u}(c,u)}{EST_{T_v}(a,v)+d(c,a)} \le \lambda.$$

However as we will show soon, only certain sensors in $\mathcal{U}$ may satisfy those two conditions simultaneously. If condition II really happens, then we check condition I:

$$\begin{aligned}\mathrm{I} &\le \frac{d_{T_u}(v,u)+d_{T_u}(b,u)+d(a,b)}{EST_{T_v}(a,v)+d(c,a)} \frac{EST_{T_v}(a,v)+d(c,a)}{EST_{T_v}(a,v)} \\ &\le \left[\lambda + \frac{d(a,b)}{EST_{T_v}(a,v)+d(c,a)}\right] \frac{EST_{T_v}(a,v)+d(c,a)}{EST_{T_v}(a,v)} \\ &\le \frac{3}{2}\lambda + \frac{1}{2}.\end{aligned} \tag{4}$$

Equation (4) follows the fact that $EST_{T_v}(a, v)$ is greater or equal to 2. This is because the optimal path from a to c to v is at least 2. Thus, $EST_{T_v}(a, v)$ cannot be smaller than 2. Equation (4) implies that the necessary condition for both conditions I and II simultaneously happening is $\frac{d_{T_u}(v,u)+d_{T_u}(a,u)}{EST_{T_v}(b,v)+d(a,b)} \in (\lambda, \frac{3}{2}\lambda + \frac{1}{2}]$. As a result, for a sensor node i, whose distance ratio is greater than $\frac{3}{2}\lambda + \frac{1}{2}$, its routing path will be completely in $\mathcal{U}$, i.e., it is a type-one sensor node. Thus, its routing path is optimal. Moreover, based on Lemma 1, sensors along the routing path constructed in $\mathcal{U}$ from such a node i to the virtual sink v guarantees to be optimal as well. Thus, all the sensors within $\mathcal{U}$ with the optimal routing path form another region $\mathcal{V}$. On the other hand, for a sensor node i whose $\frac{d_{T_u}(v,u)+d_{T_u}(i,u)}{EST_{T_v}(H_i,v)+d(i,H_i)}$ is within $(\lambda, \frac{3}{2}\lambda + \frac{1}{2}]$, the routing path formed by Algorithm 1 is not optimal. However, as we will show soon, the total amount of such nodes counts for only a small portion in $\mathcal{U}$.

Theorem 2. *For any sensor $i \in \mathcal{V}$ in $\mathcal{U}$, the routing path from sensor i to the virtual sink v formed by Algorithm 1 is optimal.*

Since only those sensors, whose locations are close to the boundary of $\mathcal{U}$ and performance ratios are between λ and $\frac{3}{2}\lambda + \frac{1}{2}$, are excluded from region $\mathcal{V}$, the size of region $\mathcal{V}$ will be close to region $\mathcal{U}$. To check region $\mathcal{V}$ in practice, we further separate the sensor nodes with or without optimal routing paths in the experiment conducted in Fig. 5b and illustrate the results in Fig. 8. Consistent with our analysis, the majority of sensors in $\mathcal{U}$ is of the optimal routing path. According to the statistics, the portion of sensors with the optimal routing path length is no less than 88 percent when λ varies.

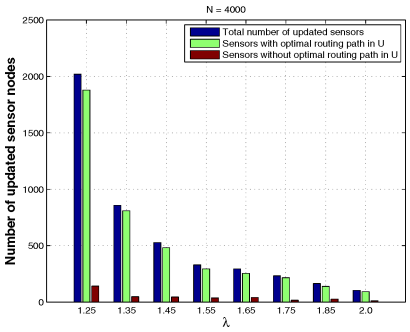


Fig. 8. Region $\mathcal{V}$ as suggested by simulations.

So far, we have demonstrated that an efficient local routing tree has been formed in region $\mathcal{U}$ by Algorithm 1, in which most sensors can reach virtual sink v through an optimal routing path and we will postpone discussing the routing path length for sensors outside $\mathcal{V}$ in the next section. As shown in Theorem 3 then, the unchanged portion in the original collection tree $\mathcal{T}_u$ together with $\mathcal{U}$ jointly and seamlessly form a complete routing tree rooted at virtual sink v.

Theorem 3. *The region $\mathcal{U}$ formed by Algorithm 1 and the nonmodified portion in $\mathcal{T}_u$ jointly form $\mathcal{T}_v$.*

Proof. After $\mathcal{T}_u$ has been constructed, each sensor $i \in V$ sets H_i as the neighbor to transmit or relay packets. During the execution of Algorithm 1, if H_i is ever changed, sensor i belongs to $\mathcal{U}$. We immediately know that sensor i can reach virtual sink v. On the other hand, it is also possible that H_i never changes. Following H_i, sensor i can reach another sensor, namely j. j shares two similar possibilities as sensor i: inside $\mathcal{U}$ or outside $\mathcal{U}$. In general, following the unchanged routing directions specified by $\mathcal{T}_u$, any sensor outside $\mathcal{U}$ can reach the original virtual sink u or some sensor within $\mathcal{U}$ eventually. According to our protocol, no matter which possibility occurs, virtual sink v is always reachable from this sensor. □

3.3 Data Routing

After *Data Collection Tree Updating* completes, a new routing structure is built. If sensor i has data to send or helps other sensors to relay data, it simply transmits data to the neighbor indicated by H_i. Data are guaranteed to be delivered toward the mobile user by Theorem 3. In this section, we examine the routing efficiency for the sensors outside $\mathcal{V}$. Theorem 4 will show that the routing delays of those sensors are bounded and controllable, and the mobile user can easily adjust the routing efficiency according to his requirement.

Theorem 4. *In the routing tree formed by Algorithm 1, the routing delay distortion from one sensor outside $\mathcal{V}$ to the virtual sink v is bounded and controlled by λ compared to its optimal routing delay.*

Proof. We first define the *accessing point* of one sensor outside $\mathcal{U}$. The accessing point of sensor i is defined as the *first* encountered sensor in $\mathcal{V}$ if we trace the routing path from sensor i to v in $\mathcal{T}_v$. Then, to show the correctness of Theorem 4, we need to check the following two different cases.

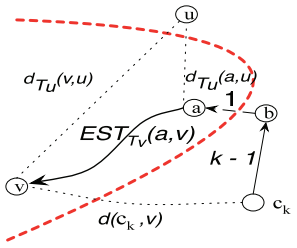


Fig. 9. Case one in Theorem 4.

Case 1 captures the scenario in which a sensor c_k reaches the virtual sink v without passing the original virtual sink u as shown in Fig. 9, where sensor c_k means that this sensor is $k \geq 1$ hop(s) away from its *accessing point*. At a result, the routing path length distortion at sensor c_k can be defined as $\frac{k+EST_{T_v}(a,v)}{d(c_k,v)}$. Then, we have

$$\begin{aligned}\frac{k+EST_{T_v}(a,v)}{d(c_k,v)} &\leq \frac{k+EST_{T_v}(a,v)}{d_{T_u}(a,u)+k-d_{T_u}(v,u)} \\ &\leq \frac{k+EST_{T_v}(a,v)}{\lambda EST_{T_v}(a,v)+k-2d_{T_u}(v,u)}.\end{aligned} \tag{5}$$

After a simplification of (5), we can further derive

$$(5) \leq 1+\frac{(k-1)+2d_{T_u}(v,u)}{d(a,v)+k-2d_{T_u}(v,u)}. \tag{6}$$

In (6), $d_{T_u}(v,u)$ is a constant. Additionally, as we have shown, $\mathcal{U}$ (together with $\mathcal{V}$) expands as λ decreases. As a result, sensor c_k will encounter an accessing point with a longer distance to the virtual sink v when λ decreases. Therefore, the delay distortion for sensor c_k is bounded and controllable through the user defined threshold λ.

Case 2 captures the scenario in which sensors reach the virtual sink v passing the original virtual sink u as shown in Fig. 10. To examine Case 2, we first **logically** reorganize all the sensors outside $\mathcal{V}$ by levels. All level k' sensors form a cluster, denoted as $L_{k'}$, satisfying

$$L_{k'} = \{i | i \notin \mathcal{V} \wedge d_{i,\mathcal{V}} = k'\}, \tag{7}$$

where $k' \geq 1$. $L_{k'}$ contains all the sensors whose minimum distance to $\mathcal{V}$ is k'. In this scenario, the delay distortion at sensor $c_{k'}$ can be expressed as $\frac{d_{T_u}(c_{k'},u)+d_{T_u}(v,u)}{d(c_{k'},v)}$. To prove the correctness of Case 2, we apply the constructive mathematical induction on the **level** of each sensor outside region $\mathcal{V}$.

Hypothesis: $\frac{d_{T_u}(c_{k'},u)+d_{T_u}(v,u)}{d(c_{k'},v)} \leq \lambda + \sum_{l=d(a,v)+1}^{d(a,v)+k'} \frac{1}{l} - 1$, where a indicates the accessing point of sensor $c_{k'}$.

Basis: when $k' = 1$. According to line 3 in Algorithm 1,

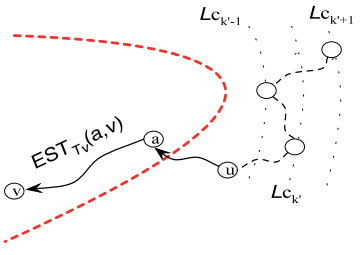


Fig. 10. Case two in Theorem 4.

$$\frac{d_{\mathcal{T}_u}(c_1,u)+d_{\mathcal{T}_u}(v,u)}{d(c_1,v)} \le \lambda.$$

Equivalently, $\frac{d_{\mathcal{T}_u}(c_1,u)+d_{\mathcal{T}_u}(v,u)}{d(c_1,v)} \le \lambda+1-1$.

Induction: suppose that the hypothesis holds for all levels up to k'. We now check the case $k'+1$.

$$\begin{aligned}\frac{d_{\mathcal{T}_u}(c_{k'+1},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'+1},v)} &\le \frac{d_{\mathcal{T}_u}(c_{k'},u)+1+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)+1}\\ &\le \left(\lambda+\sum_{l=d(a,v)+1}^{d(a,v)+k'}\frac{1}{l}-1\right)+\frac{1}{d(c_{k'},v)+1} \qquad (8)\\ &= \lambda+\sum_{l=d(a,v)+1}^{d(a,v)+k'+1}\frac{1}{l}-1.\end{aligned}$$

Equation (8) implies that at sensor $c_{k'}$:

$$\frac{d_{\mathcal{T}_u}(c_{k'},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)} \le \lambda+\ln\frac{d(a,v)+k'}{d(a,v)}. \qquad (9)$$

Similar to Case 1, the delay distortion of sensor $c_{k'}$ is bounded and controllable through the user defined threshold λ. By setting a smaller λ, the routing efficiency in $\mathcal{T}_v$ gets close to the optimal result while it suffers a longer delay to construct the data collection tree. □

According to Theorem 4, the routing path length distortion from any sensor outside region $\mathcal{V}$ formed by Algorithm 1 is bounded, which means although the routing structure formed by the proposed algorithm is suboptimal, the routing delay is not excessively long. More importantly, the delay distortion is under the control of λ (e.g., *proportional* to λ) and the size of region $\mathcal{V}$ (as well as region $\mathcal{U}$) is reverse proportional to λ. As a result, the mobile user is able to achieve a balance between the routing efficiency and the cost of building the routing structure.

4 System Properties

In Section 3, we have described the design details of the proposed approach. In this section, we discuss the system properties tailored for the mobility of the user.

4.1 System Dynamics

As stated previously, the mobile user keeps moving around in the network and may acquire data at arbitrary places. At the initial data collection point u, *Data Collection Tree Initialization* is launched to construct the first routing tree $\mathcal{T}_u$ in the system. Afterwards, *Data Collection Tree Updating* is launched at each v_k to efficiently form $\mathcal{T}_{v_k}$, where $k=1,2,3,\ldots$

Theorem 1 has proven that region $\mathcal{U}$ formed by our protocol is a bounded area. In this section, we further reveal some properties of region $\mathcal{U}$ related to the mobility of the mobile user. The size of $\mathcal{U}$ is determined by both λ and the distance[3] between the original virtual sink u and the current virtual sink v_k. Fig. 11 depicts the size of region $\mathcal{U}$ according to varied λ values, showing that the size of $\mathcal{U}$ is *reverse proportional* to λ. On the other hand, the size of $\mathcal{U}$ is *proportional* to the distance between u and v_k. As the mobile user moves a distance from u, if λ is fixed, Algorithm 1 approaches updating the entire network, i.e., our protocol migrates to form a globally optimal data collection tree in the network.

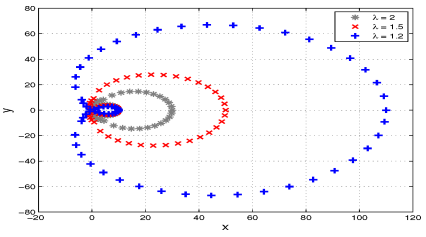


Fig. 11. Impact of λ on region $\mathcal{U}$.

In order to achieve the balance between the routing efficiency and the cost of the routing tree construction, we dynamically adjust λ. In Section 6, we use a simple linear adjustment policy to update λ, which is shown effective in our experiments. We observe that in a network with 5,000 sensors, at most 30 percent sensors are updated while the delay distortion can be maintained smaller than 1.5 in the worst case.

4.2 Data Streaming Property

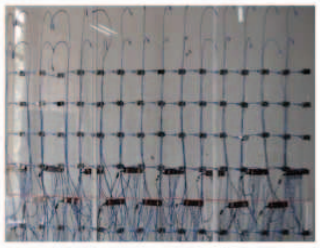
As mentioned before, the entire ubiquitous data collection process may comprise multiple rounds due to the mobility of the mobile user. In such a system, the data streaming property measures how smoothly data flow during the transition between two consecutive places of data access. During the movement, if the data to be collected can always flow toward the mobile user, those data can be rapidly collected once the mobile user accesses the network. The quality of the data streaming property depends on how the underlying data collection tree migrates during transitions. In this section, we examine the data streaming property of the approach proposed in this paper.

Lemma 2. *If sensors v_{k-1} and v_k are any two consecutive virtual sinks (v_k is after v_{k-1}), during the routing tree construction at v_k by Algorithm 1, uncollected data in the system flow toward v_k.*

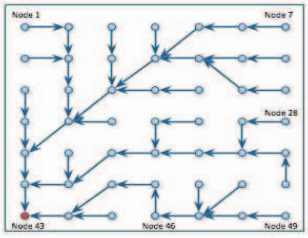
Proof. Before the execution of our protocol, following H_i of each sensor i, uncollected data flow toward v_{k-1}. During the execution of Algorithm 1, the modification of H_i will change the routing destination of sensor i from v_{k-1} to v_k. As a result, the routing destinations of all the ancestors of sensor i are changed to v_k automatically. Through the discussions in Section 3, we know that there is no other possibility. Therefore, we finish the proof of Lemma 2. □

Lemma 2 reveals one critical feature of our proposed protocol. By executing Algorithm 1, region $\mathcal{U}$ behaves like a potential source that attracts data. The data flows always move toward the mobile user. As the mobile user passes by the sensor nodes and updates the data collection tree along the moving trajectory, the data flows will be seamlessly guided toward the user without being stuck at any local

3. At virtual sink v_1, $\mathcal{T}_{v_1}$ can be constructed by taking advantages of the existing $\mathcal{T}_u$ as stated in Section 3. However, for each $v_k, k=2,3,\ldots$, there are two typical choices to form $\mathcal{T}_{v_k}$. $\mathcal{T}_{v_k}$ can be formed based on either $\mathcal{T}_u$ or $\mathcal{T}_{v_{k-1}}$. Since $\mathcal{T}_{v_{k-1}}$ is not optimal already in terms of the routing efficiency, the routing performance of $\mathcal{T}_{v_k}$ will be difficulty to guarantee if $\mathcal{T}_{v_{k-1}}$ is used as the basis. Therefore, in this paper, $\mathcal{T}_{v_k}$ is formed based on $\mathcal{T}_u$.



(a) Testbed



(b) Data collection tree at node 43

Fig. 12. Testbed-based experiments.

maximum points. Therefore, we have the following proposition.

Proposition 1. *Launching Algorithm 1 during the movement of the mobile user, a good data streaming property can be achieved such that the data flows are attracted by the user and will not be stuck at any intermediate nodes.*

4.3 Prediction Compatibility

Existing literatures [16], [17], [20] have explored the possibility of predicting the movement of mobile users. In some scenario, such as the in-door environment, the trajectory of the mobile user is highly constrained and its future movement can be accurately predicted. In such a case, the prediction mechanism can make the construction of the data collection tree to the predicted data access points such that data can be delivered to the predicted location. In some other scenarios, however, it is not easy to accurately predict the future location of the mobile user, e.g., some outdoor environments. The low prediction accuracy may significantly deteriorate the system performance due to two possible reasons: the improperly formed routing tree will 1) waste the energy of the system and 2) route data to undesired destinations. Therefore, whether the prediction mechanism should be adopted or not depends on the networking conditions.

The approach proposed in this work is compatible to existing prediction mechanisms. If the mobile user enables the prediction mechanism, the prediction module works as a black box such that whenever the mobile user moves to the next data collection point, the prediction module predicts such a location and the data collection tree will be formed in advance. Even the prediction is incorrect, the rapid formation of the data collection tree can largely compensate the prediction failure.

4.4 Support to Multiple Mobile Users

So far, we have taken the single user as a vehicle to introduce the basic design principle and curial properties of our proposed protocol. However, our solution is not limited to the single-user data collection case, and it can be further extended to support multiple users for collecting data simultaneously. To this end, in the system, multiple independent data collection trees will be formed and different collection trees are distinguished based on user IDs. Flooding messages and sensory data for different mobile users are labeled with user IDs as well. For each formed routing tree (denoted as $\mathcal{T}$), the sensor (e.g., sensor i) maintains a corresponding $H_i(\mathcal{T})$. Once a packet on the routing tree T needs to be transmitted, sensor i simply relays the packet to the neighbor indicated by $H_i(\mathcal{T})$.

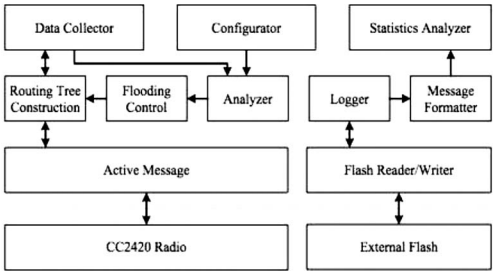


Fig. 13. The diagram of software module.

5 Experimental Evaluation

In previous sections, we elaborate the design principles and important properties of our ubiquitous data collection approach. In this section, we validate the feasibility and applicability of the proposed protocol in practice.

5.1 Experiment Setting

We implement our protocol on TelosB motes and use a 49-node testbed to examine its performance, as shown in Fig. 12a. Forty nine nodes are organized as a 7×7 grid. Due to the experimental space limitation, the power of each Telosb mote is set to be the minimum level and the communication range is about 10 centimeters. The average degree of each sensor node is around six. Starting from the left-top conner, sensors are placed following the left-to-right and top-to-bottom order based on their IDs.

The software on the experimental sensor nodes is developed based on TinyOS 2.1. Fig. 13 depicts the design diagram of the software modules in detail. The Data Collector module and the Configurator module provide the received flooding packet and the system parameter λ, respectively. Note that a virtual received flooding packet will be offered to the sink node at the initial stage of *Data Collection Tree Updating*. Based on the input information, the Analyzer module figures out whether the updating needs to be performed or not. If the answer is positive, the Flooding Control module conducts several necessary local information updatings and prepares the flooding message for the Routing Tree Construction module. The Logger module is in charge of data access (read and write) to the measurement serial flash. The Statistics Analyzer module merges and encapsulates the data from sensors, network, and flash, based on the preconfigured message formats.

We conduct two trails of experiments. During the experiment, the mobile user enters the sensor network

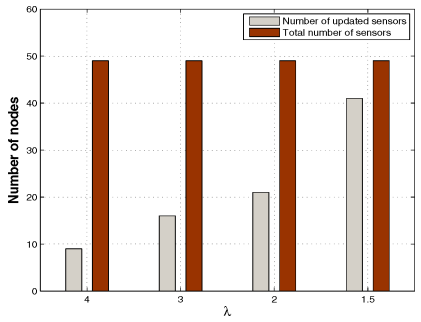


Fig. 14. Updated nodes versus λ.

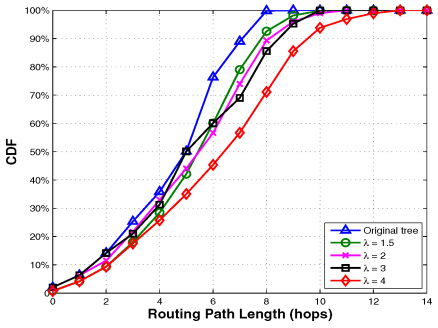


Fig. 15. CDF of routing path lengths (at node 49).

and builds the first routing tree at node 43, i.e., the left-bottom node. The resulting data collection tree is depicted in Fig. 12b.

5.2 First Trail of Experiments

In the first trail of our experiment, we investigate the system parameter λ when the mobile user moves to node 49 and accesses the network for data collection.

Investigation on system parameter λ. Fig. 14 depicts the number of sensor nodes that are updated during updating the data collection tree. We vary λ from 4 to 1.5. Consistent with our previous analysis, the number of updated sensors is reverse proportional to λ. A small number of sensors updated during the routing transition indicates 1) a fast routing tree formation and 2) the small communication overhead. From Fig. 14, we can observe that when λ is sufficiently large (e.g., $\lambda = 4$), only a small number of nodes are updated (e.g., less than 20 percent).

Investigation on the CDF of routing path lengths. In Fig. 15, we depict the cumulative distribution function (CDF) of the routing path lengths of the sensor nodes in the network when we build the data collection trees with the different values of λ. Fig. 15 reveals that though the routing path becomes shorter when we use a smaller λ, there is no significant performance difference with different λs. Such a result further demonstrates the benefit of our protocol, i.e., a small routing efficiency distortion can trade a significant reduction of the construction cost.

Investigation on individual routing path length. In Fig. 16,[4] the x-axis indicates the ID number of each sensor node in this experiment and the y-axis measures its corresponding routing path length by our protocol at node 49. We present

4. Due to the hardware issue, sensor 19 fails to send back its information after the user moves in the experiments.

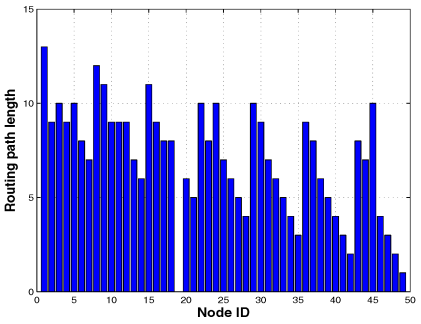


Fig. 16. Routing path length versus Node ID.

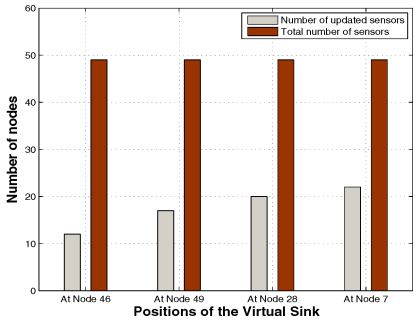


Fig. 17. Updated nodes versus moving position.

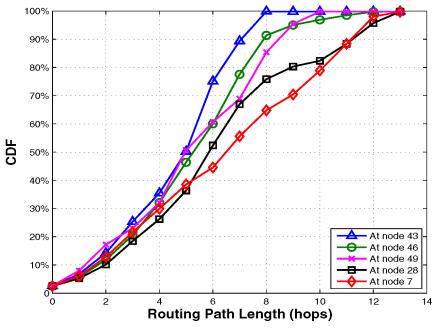


Fig. 18. CDF of routing path lengths (at different positions).

the routing path length of each sensor with an instance when $\lambda = 4$. The average routing path length of the sensors is 6.8. The average length of the routing path in the global optimal routing tree is 5.9. Fig. 16 demonstrates that even only a small number of sensors are updated, the network can achieve a comparable routing efficiency compared to the optimal performance.

5.3 Second Trail of Experiments

In the second trail of experiment, we examine the system performance with regard to the movement of the user. The moving path of the user is: node 43 $\Rightarrow$ node 46 $\Rightarrow$ node 49 $\Rightarrow$ node 28 $\Rightarrow$ node 7. During the process, λ is set to be 3.

Investigation on updated areas at different positions. Fig. 17 shows that during the movement of the mobile user, more sensors will be updated as the user moves further away from his original places. There are at most 21 sensor nodes updated when the user moves to node 7, which is 9 hops from the original start point node 43. Fig. 17 provides another good indication that the influence of our protocol is only limited.

Investigation on the CDF of routing path lengths. In Fig. 18, we depict the CDF of the routing path lengths of the sensor nodes in the data collection trees built at different places

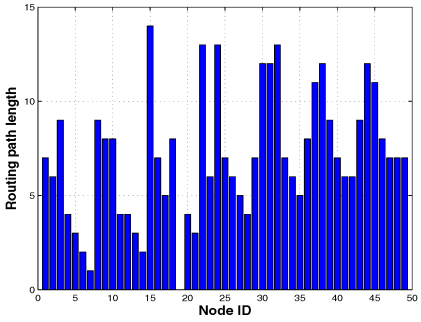


Fig. 19. Routing path length versus Node ID.

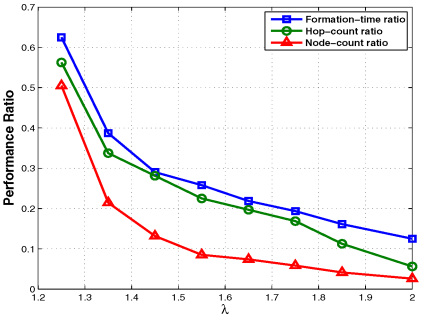


Fig. 20. Updating overhead versus λ.

(from node 43 to node 7). Consistent with our analysis, when the mobile user moves further away from the start point at node 43, the routing paths would be shifted longer. Such a routing efficiency distortion, however, is not significant as shown in the figure. According to statistics, the worst routing path length distortion is only 20 percent. Nevertheless, the cost in building the routing tree can be reduced up to 54 percent.

Investigation on individual routing path length: similar to Fig. 16, the x-axis and the y-axis in Fig. 19 indicate the ID number of each sensor node in this experiment and its corresponding routing path length by our protocol at node 7, respectively. In Fig. 19, we present the routing path length of each sensor with an instance when the mobile user is at node 7. The average routing path length in this case is 7.1. Most of sensors have short routing paths.

6 Simulation Evaluation

We conduct comprehensive and large-scale simulations to further examine the efficiency and scalability of our protocol. We compare the performance of our λ-Flooding based protocol with the approach directly using CTP protocol [10] at each data collection point and the movement prediction approach proposed in [16].

6.1 Simulation Setting

We simulate randomly deployed sensor nodes in a rectangular area with an average node degree ranging from 5 to 10. The mobile user enters the field from the boundary of the network. The velocity of the movement is set to $1\ \mathrm{m/s}$. At each data collection point, the sojourn time of the mobile user is chosen from 20 to 60 seconds uniformly at random. For each movement, the mobile user stochastically roams $15 \sim 30$ seconds. More detailed simulation settings will be specified in following sections. For each network setting, we perform 20 runs and demonstrate the average performance.

6.2 Simulation Results

Investigation on system parameter λ. We first examine the effect of parameter λ of our λ-Flooding approach in a 4,000-node network. At one data collection point (15 meters away from the original virtual sink in our experiment), we collect statistics of the system with different settings of λ and present the results in Fig. 20. We examine three performance metrics:

- Node-count ratio: the ratio of the number of updated sensor nodes over the total number of sensor nodes in the network.
- Hop-count ratio: the ratio of the average hop distance of region $\mathcal{U}$ over the average hop distance of the entire network.
- Formation-time ratio: the ratio of the formation time of region $\mathcal{U}$ over the formation time of a global optimal routing tree.

All three metrics measure the area and cost of the data collection tree updating in our approach. Fig. 20 demonstrates that region $\mathcal{U}$ formed by our approach is under control of λ. As λ increases, both node-count ratio and hop-count ratio decrease accordingly, i.e., the size of region $\mathcal{U}$ is reverse proportional to λ. The mobile user can thus control the system overhead through adjusting λ instantly. From Fig. 20, we also observe that our approach can efficiently update the routing tree, which is clearly indicated by the formation-time ratio. Rapid routing tree formation largely accelerates the data collection process as we will demonstrate later.

System performance as the mobile user's movement. In Fig. 21, we simulate the movement of a mobile user. The mobile user moves from the left-bottom conner of the network, roughly following three-quarter of an ellipse inscribed in the field, to the middle point of the bottom line of the network. There are 5,000 sensors deployed in the network and λ is 2.6. Fig. 21 presents the system performance in two aspects. The metrics of node-count ratio, hop-count ratio, and formation-time ratio are depicted in the upper figure to show the affected area in updating the collection tree. The lower figure depicts the sensors' path lengths in the data collection tree during the movement of the mobile user. According to Fig. 21, the affected area in updating the data collection tree

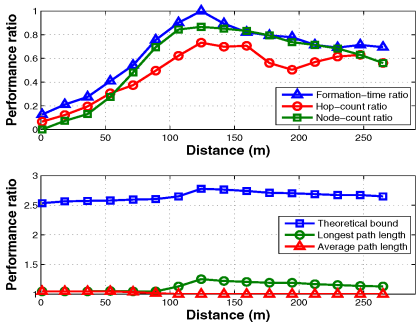


Fig. 21. System performance versus Distance (Fixed λ).

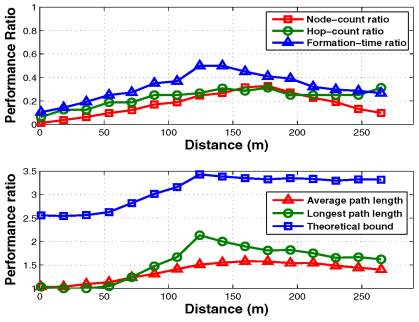


Fig. 22. System performance versus Distance (Adjusted λ).

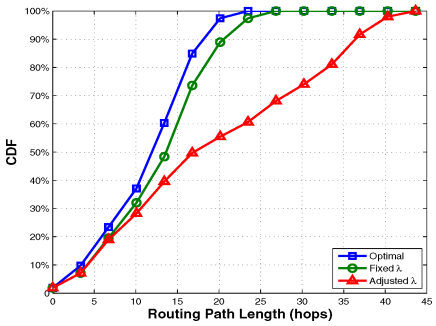


Fig. 23. CDF of routing path lengths.

gradually approaches the entire network during the user's movement, i.e., our protocol migrates to build an optimal routing tree when the mobile user moves sufficiently far away from the original virtual sink, say 100 meters as depicted in the upper figure in Fig. 21. As the lower figure depicts, the routing efficiency is approximately optimal as both average path length and longest path length ratios get close to 1. The derived theoretical bound is also shown in Fig. 21. which is loose in this scenario. The information delivered in Fig. 21 is that with fixed λ, the routing efficiency and the updating cost of our approach cannot be well balanced in a large scale network. As we will show soon, an adjusted λ can break such a barrier.

System performance with adjusted λ. Fig. 20 reveals that the size of $\mathcal{U}$ decreases as λ increases and Fig. 21 shows that $\mathcal{U}$ expands as the mobile user moves far away from the original sink. To further balance the system performance, we explore an adjusted λ during the movement of the mobile user. There can be a variety of schemes to implement such an idea. In our protocol, we apply a linear updating policy for λ:

$$\lambda_{v_k} \leftarrow \lambda_{v_{k-1}} + SNG \times c \times d_{\mathcal{T}_u(v_k,u)}, \tag{10}$$

where $SNG = +1$, if $d_{\mathcal{T}_u(v_k,u)} \geq d_{\mathcal{T}_u(v_{k-1},u)}$; $SNG = -1$, otherwise. c is chosen from $[0.1, 0.3]$ uniformly at random and v_k is the current virtual sink.

Fig. 22 presents the same metrics as those shown in Fig. 21, but with a dynamically adjusted λ during the user movement. From the upper figure in Fig. 22, we observe that the updated region $\mathcal{U}$ is maintained below 30 percent of the network scale, which ensures a low routing tree formation cost and a short construction time. On the other hand, according to the lower figure, the delay distortion introduced by our approach is not excessively long. More precisely, the distortion of the average path length is no more than 1.6 and the distortion of the longest path length is less than 2.5, which indicates a low delay distortion to the optimal value. The derived distortion bound matches the simulation result and bounds the longest path, i.e., smaller than 3.5 in the worst case. Fig. 22 shows that by dynamically adjusting λ, the mobile user can achieve an excellent tradeoff between the cost of the routing tree formation and the routing efficiency.

CDF of delay performance. We take a snapshot at the middle point of the mobile user's movement according to the simulation setting in Fig. 21. We illustrate the CDF of sensors' routing path lengths in Fig. 23. With the fixed λ setting, as most sensors have been updated by our approach, the CDF of sensors' routing path lengths is similar to that in the optimal routing tree. On the other hand, Fig. 23 also indicates that with the adjusted λ setting, most of sensors still have short routing paths and only a small number of sensors suffer a relatively long routing delay. The routing efficiency distortion with adjusted λ is not excessively large while the cost of updating the routing tree formation is significantly reduced.

System performance versus network size. To examine the scalability of our approach, we vary the network size from 1,000 to 10,000 and use the adjusted λ policy. We demonstrate the average performance for all data collection processes at each network size. Fig. 24 depicts the results, which shows that our protocol has a good scalability with regard to the variation of the network size. The influence of our approach is guaranteed to be limited, such that the time cost of the routing tree formation is always much less than that of building the optimal tree. Meanwhile, the routing efficiency distortion keeps low.

Comparison with other protocols. We examine the data collection efficiencies of different protocols. In the network, each sensor is set to possess one sensory data that is desired by the mobile user. After the mobile user finishes accessing all those data, the data collection process completes. The finish time is defined as the time difference from the time when the mobile user builds the first collection tree to the time when the data collection completes. A short finish time indicates a better system performance.

With the same sojourn time at each data collection point, the mobile user can access more data in λ-Flooding compared to CTP due to its rapid routing tree formation. The streaming property can further reduce the routing delay by "absorbing" the data flow nearby the mobile user instead of letting them flow to the previous virtual sink during the movement of the user. Fig. 25 demonstrates the performance of different protocols with different network sizes. According to the

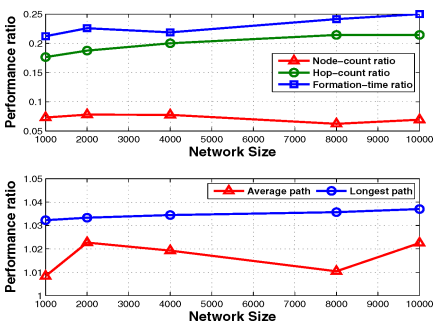


Fig. 24. System performance versus Network size.

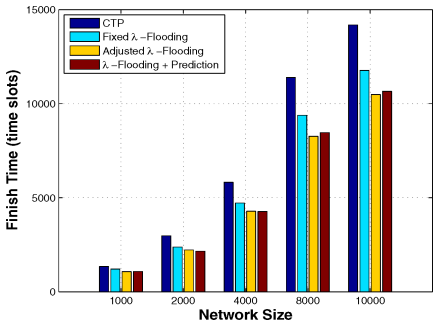


Fig. 25. Finish time of different protocols.

result, both two λ-Flooding schemes can largely reduce the finish time compared to CTP. As a matter of fact, the streaming property and the rapid tree formation attribute jointly help shorten the finish time of the data collection, thus, Adjusted λ-Flooding approach outperforms Fixed λ-Flooding approach. To conduct a comprehensive comparison, we examine the performance of our protocol combined with the prediction mechanism in [16], denoted as λ-Flooding+Prediction. With a high prediction accuracy (e.g., 80 percent in this section), the prediction mechanism indeed reduces the finish time compared to CTP. However, as mentioned before, the prediction mechanism is highly constrained by the network context. On contrast, our protocol achieves a comparable performance but is much less subjected to the network environment.

7 Related Work

As a basic operation, the data collection in WSNs has been extensively studied. A surge of works study the data gathering but with static settings. In addition, according to how does each packet transmitted, the data collection can be further divided into two categories: with aggregation or without aggregation. In the former category, in-network aggregating data results in a reduction in the amount of bits transmitted, and hence, saves energy. Typical examples include [15], [21]. Michael et al. [15] propose the first such protocol. In [21], authors study the construction of a data gathering tree to maximize the network lifetime. In the latter category, Rangwal et al. [14] propose to collect data through a tree structure with fair rate control. [12] proposes to form an information potential-based routing structure. In [11], Challen et al. present IDEA, a sensor network service enabling effective network-wide data collection framework. Even WSNs are capable to support large volume data accessing, while recent works [16], [17], [22] indicate that existing data collection schemes under the static setting incur a poor performance if they are used in the network with mobile users directly. The problem will become even worse if the transmission loss and interference are serious in the network [23].

In the network context with mobile users, most existing works explore how to plan the moving trajectory for the mobile user or sink to achieve an efficient data collection. [18] exploits reactive mobility to improve the target detection performance. Mobile sensors collaborate with static sensors and move reactively in [18]. Tan et al. [19] further jointly optimizes data routing paths and the data collection tour. In [24], the authors investigate the approach that makes use of a mobile sink for balancing the traffic load and in turn improving network lifetime. SinkTrail is proposed in [25] as a proactive data reporting protocol, and the SHDGP problem is studied in [26]. Moreover, on the application level, Gao et al. [27] propose to adopt HST tree to distributed manage resources in WSNs and [28] introduces a method to collect event data using mobile sinks. On the other hand, some recent works do not assume the fixed trajectory of mobile users or sinks. In [20], authors propose to use data traffic to probe the future position of the mobile user. The mobile user probing process does not introduce extra communication costs; nevertheless, [20] is not tailored for the optimization of routing tree transitions. In [16], authors propose to use mobility graphs to predict the future data collection position of the mobile user. Lee et al. [17] utilize linear programming to optimize the prediction accuracy. Those works mainly focus on predicting the movement of mobile users to improve routing efficiency. So far as we know, however, no works for directly optimizing the ubiquitous data collection process of mobile users have been proposed.

8 Conclusion

In this work, we study the ubiquitous data collection for mobile users in wireless sensor networks. Essentially different from existing works, we utilize the spatial correlation to efficiently build and update the data collection tree in the system. Whenever the mobile user moves and changes the virtual sink to access the sensor network, a new data collection tree can be efficiently formed by locally modifying the previously constructed data collection tree. With such an approach, the routing performance is bounded and controlled compared to the optimal performance while the overhead in updating the routing structure is significantly reduced. Such a property ensures low data collection delay, providing real-time data acquisition for the mobile user. In addition, our proposed protocol is compatible to existing mobility prediction mechanisms and easy to implement. We implement the proposed protocol in a 49-node testbed and test its feasibility and applicability in practice. We further conduct extensive simulations, which prove the efficiency and scalability of our approach.

Acknowledgments

This study is supported in part by the NSFC Major Program 61190110, NSFC Grant No. 60970123, National High-Tech R&D Program of China (863) under grant No. 2011AA010100, Singapore MOE AcRF Tier 2 grant MOE2012-T2-1-070, and NAP M4080738.020. A preliminary version of this study has been presented in IEEE INFOCOM 2011 [29].

References

[1] T. He, J. Stankovic, T. Abdelzaher, and C. Lu, "A Spatiotemporal Communication Protocol for Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 16, no. 10, pp. 995-1006, Oct. 2005.

[2] L. Wang and W. Liu, "Navigability and Reachability Index for Emergency Navigation Systems Using Wireless Sensor Networks," *Tsinghua Science and Technology,* vol. 16, no. 6, pp. 657-668, 2011.
[3] Y. Zhu and L. Ni, "Probabilistic Approach to Provisioning Guaranteed Qos for Distributed Event Detection," *Proc. IEEE INFOCOM,* pp. 592-600, 2008.
[4] I. Stojmenovic, "Localized Network Layer Protocols in Wireless Sensor Networks Based on Optimizing Cost over Progress Ratio," *IEEE Network,* vol. 20, no. 1, pp. 21-27, Jan./Feb. 2006.
[5] D. Guo, J. Wu, H. Chen, Y. Yuan, and X. Luo, "The Dynamic Bloom Filters," *IEEE Trans. Knowledge and Data Eng.,* vol. 22, no. 1, pp. 120-133, Jan. 2010.
[6] Y. Liu, Y. Zhu, and L.M. Ni, "A Reliability-Oriented Transmission Service in Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 22, no. 12, pp. 2100-2107, Dec. 2011.
[7] S. Tang, X. Mao, and X. Li, "Efficient and Fast Distributed Top-K Query Protocol in Wireless Sensor Networks," *Proc. IEEE 19th Int'l Conf. Network Protocols (ICNP),* pp. 99-108, 2011.
[8] I. Stojmenovic and X. Lin, "Loop-Free Hybrid Single-Path/Flooding Routing Algorithms with Guaranteed Delivery for Wireless Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 12, no. 10, pp. 1023-1032, Oct. 2001.
[9] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, and X. Li, "Does Wireless Sensor Network Scale? A Measurement Study on Greenorbs," *Proc. IEEE INFOCOM,* pp. 873-881, 2011.
[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection Tree Protocol," *Proc. ACM Seventh Conf. Embedded Networked Sensor Systems,* pp. 1-14, 2009.
[11] G. Challen, J. Waterman, and M. Welsh, "IDEA: Integrated Distributed Energy Awareness for Sensor Networks," *Proc. Eighth Ann. Int'l Conf. Mobile Systems, Applications and Services (Mobisys),* pp. 35-48, 2010.
[12] H. Lin, M. Lu, N. Milosavljevic, J. Gao, and L.J. Guibas, "Composable Information Gradients in Wireless Sensor Networks," *Proc. ACM Seventh Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 121-132, 2008.
[13] Y. Mao, F. Wang, L. Qiu, S. Lam, and J. Smith, "S4: Small State and Small Stretch Compact Routing Protocol for Large Static Wireless Networks," *IEEE/ACM Trans. Networking,* vol. 18, no. 3, pp. 761-774, June 2010.
[14] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-Aware Fairerate Control in Wireless Sensor Netowrks," *Proc. ACM SIGCOMM,* pp. 63-74, 2006.
[15] S. Michael, M. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *Proc. Fifth Usenix Symp. Operating Systems Design and Implementation (OSDI),* pp. 131-146, 2002.
[16] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, and L. Guibas, "Predictive QoS Routing to Mobile Sinks in Wireless Sensor Networks," *Proc. ACM Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 109-120, 2009.
[17] H. Lee, M. Wicke, B. Kusy, O. Gnawali, and L. Guibas, "Data Stashing: Energy-Efficient Information Delivery to Mobile Sinks through Trajectory Prediction," *Proc. ACM/IEEE Ninth Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 291-302, 2010.
[18] R. Tan, G. Xing, J. Wang, and H. So, "Exploiting Reactive Mobility for Collaborative Target Detection in Wireless Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 9, no. 3, pp. 317-332, Mar. 2010.
[19] G. Xing, T. Wang, Z. Xie, and W. Jia, "Rendezvous Planning in Wireless Sensor Networks with Mobile Elements," *IEEE Trans. Mobile Computing,* vol. 7, no. 12, pp. 1430-1443, Dec. 2008.
[20] J.W. Lee, B. Kusy, T. Azim, B. Shihada, and P. Levis, "Whirlpool Routing for Mobility," *Proc. ACM Mobihoc,* pp. 131-140, 2010.
[21] Y. Wu, Z. Mao, S. Fahmy, and N. Shroff, "Constructing Maximum-Lifetime Data Gathering Forests in Sensor Networks," *IEEE/ACM Trans. Networking,* vol. 18, no. 5, pp. 1571-1584, Oct. 2010.
[22] O. Durmaz, A. Ghosh, B. Krishnamachari, and K. Chintalapudi, "Fast Data Collection in Tree-Based Wireless Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 11, no. 1, pp. 86-99, Jan. 2012.
[23] K. Wu, H. Tan, Y. Liu, J. Zhang, Q. Zhang, and L. Ni, "Side Channel: Bits over Interference," *IEEE Trans. Mobile Computing,* vol. 11, no. 8, pp. 1317-1330, Aug. 2012.
[24] J. Luo, J. Panchard, M. Piorkowski, M. Grossglauser, and J. Hubaux, "Mobiroute: Routing Towards a Mobile Sink for Improving Lifetime in Sensor Networks," *Proc. IEEE Int'l Conf. Distributed Computing in Sensor Systems (DCOSS),* pp. 480-497, 2006.
[25] X. Liu, H. Zhao, X. Yang, X. Li, and N. Wang, "Trailing Mobile Sinks: A Proactive Data Reporting Protocols for Wireless Sensor Networks," *Proc. IEEE Seventh Int'l Conf. Mobile Ad Hoc and Sensor Systems (MASS),* pp. 214-223, 2010.
[26] K. Tian, B. Zhang, K. Huang, and J. Ma, "Data Gathernig Protocols for Wireless Ensor Networks with Mobile Sinks," *Proc. IEEE GLOBECOM,* pp. 1-6, 2010.
[27] J. Gao, L. Guibas, N. Milosavljevic, and D. Zhou, "Distributed Resource Management and Matching in Sensor Networks," *Proc. IEEE Int'l Conf. Information Processing in Sensor Networks (IPSN),* pp. 97-108, 2009.
[28] J. Luo, D. Wang, and Q. Zhang, "On the Double Mobility Problem for Water Surface Coverage with Mobile Sensor Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 23, no. 1, pp. 146-159, Jan. 2012.
[29] Z. Li, M. Li, J. Wang, and Z. Cao, "Ubiquitous Data Collection for Mobile Users in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* pp. 2246-2254, 2011.

Zhenjiang Li (M'12) received the BE degree from the Department of Computer Science and Technology at Xi'an Jiaotong University, China, in 2007, the Mphil degree from the Department of Electronic and Computer Engineering at Hong Kong University of Science and Technology, in 2009, and the PhD degree from the Department of Computer Science and Engineering at Hong Kong University of Science and Technology, in 2012. His research interests include distributed systems, wireless sensor networks, and wireless and mobile systems. He is a member of IEEE.

Yunhao Liu (M'02-SM'06) received the BS degree from the Automation Department, Tsinghua University, Beijing, China, in 1995, and the MS and PhD degrees in computer science and engineering from Michigan State University, in 2003 and 2004, respectively. He is a member of the Tsinghua National Lab for Information Science and Technology and the director of the Tsinghua National MOE Key Lab for Information Security. He is also a member of the faculty with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His research interests include Distributed Systems and Wireless Sensor Networks/RFID, Cyber Physical Systems and IoT, P2P, Network Management and Diagnosis. He is a senior member of the IEEE.

Mo Li (M'06) received the BS degree in the Department of Computer Science and Technology from Tsinghua University, China, in 2004 and the PhD degree in the Department of Computer Science and Engineering from Hong Kong University of Science and Technology in 2009. He is currently an assistant professor in School of Computer Engineering of Nanyang Technological University, Singapore. His research interest includes wireless sensor networking, pervasive computing, mobile and wireless computing. He won ACM Hong Kong Chapter Prof. Francis Chin Research Award in 2009 and Hong Kong ICT Award C Best Innovation and Research Grand Award in 2007. He is a member of the IEEE and ACM.

Jiliang Wang (M'12) received the BE degree in the Department of Computer Science from University of Science and Technology of China, in 2007 and the PhD degree in the Department of Computer Science and Engineering from Hong Kong University of Science and Technology. His research interest includes wireless sensor networks, network measurement, and pervasive computing. He is a member of the IEEE.

Zhichao Cao (M'12) received the BE degree in the Department of Computer Science and Technology from Tsinghua University, Beijing, China, 2009. He is currently a Third year PhD student of Hong Kong University of Science and Technology. His current research interest is wireless sensor networks. He is a member of the IEEE.