Ubiquitous Data Collection for Mobile Users in Wireless Sensor Networks

Zhenjiang Li[†§], Mo Li[§], Jiliang Wang[†] and Zhichao Cao[†]
[†]Department of Computer Science and Engineering, Hong Kong University of Science and Technology
[§]School of Computer Engineering, Nanyang Technological University
Email: lzjiang@cse.ust.hk, limo@ntu.edu.sg, {aliang, caozc}@cse.ust.hk

***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 local 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.**

I. 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 environmental monitoring [1], security surveillance [2], localization [3]–[5] and 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 [6], 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, and 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 [7]–[12]. 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 [13], building the data collection tree introduces a large volume of communication overheads. Besides, the routing transitions between different data collection trees contain a non-negligible 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 [13]–[15]. None of those works focus on essentially optimizing the routing transitions, with reduced transition 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 local 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

Fig. 1. Ubiquitous data collection in Tianmu mountain

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 towards 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 of this paper is presented in Section II. We introduce our ubiquitous data collection protocol and related properties in Section III. In Section IV, we implement the proposed protocol in a 49-node testbed and further examine its performance by simulation in Section V. In the end, we conclude this paper in Section VII.

II. Preliminary

In this section, we will formally formulate the problem considered in this paper. 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 in the network and accesses network-wide data at anytime according to his needs. In this work, we aim to develop efficient routing transitions within the network such that the data collection tree will be carefully maintained and updated according to the movement of the mobile user. Assumptions and requirements of the system performance will be presented in the rest of this section.

A. Assumptions

To facilitate our discussion, we make 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 firstly 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 ETX to build the data collection tree.

B. 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 *local* 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 towards the mobile user even though there exist routing transitions due to the user's mobility.

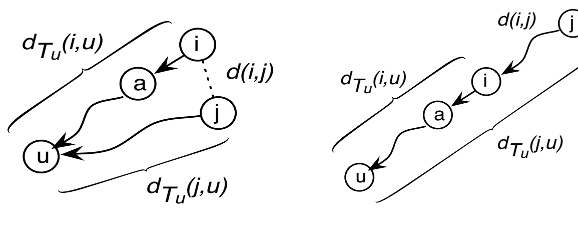
III. System Design

We elaborate the design of our protocol in this section. 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 subsections, 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*.

A. 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 [7], [8], [10]–[12]. Similar to these existing schemes, *Data*



(a) Scenario one (b) Scenario two

Fig. 2. Illustrations of $d(i,j)$ and $d_{\mathcal{T}_u}(i,j)$

Collection Tree Initialization in our protocol is realized by the flooding control in an iterative manner, like [7]. 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 successfully, 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 Fig. 2.

B. 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 re-launched to form $\mathcal{T}_v$. Some alternative updating methods are proposed as well [9]. 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 sensor network.

[1]The communication cost can be the hop-count distance on the path, the path ETX aggregates [7], the information potential [9], the ICTP [8], and etc.

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 the 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, all the sensor nodes can reach the new virtual sink through the routing paths on the original 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 local 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 suboptimal, 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 $EST_{\mathcal{T}_v}(i,v) \leftarrow EST_{\mathcal{T}_v}(j,v) + d(i,j)$.

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 $EST_{\mathcal{T}_v}(i,v)$

Algorithm 1 Local Updating Algorithm at Sensor i

```
1:  while Receiving a flooding message from sensor $j$ do
2:      if $EST_{\mathcal{T}_v}(j,v) + d(i,j) < EST_{\mathcal{T}_v}(i,v)$ then
3:          if $\frac{d_{\mathcal{T}_u}(v,u)+d_{\mathcal{T}_u}(i,u)}{EST_{\mathcal{T}_v}(j,v)+d(i,j)} > \lambda$ then
4:              $EST_{\mathcal{T}_v}(i,v) \leftarrow EST_{\mathcal{T}_v}(j,v) + d(i,j)$
5:              $H_i \leftarrow j$
6:              Flood $M_i(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(i,v))$ to its neighbors
7:          else
8:              Discard $M_j(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(j,v))$
9:          end if
10:     else
11:         Discard $M_j(d_{\mathcal{T}_u}(v,u), EST_{\mathcal{T}_v}(j,v))$
12:     end if
13: end while
```

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 local 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 $\mathcal{U}$ is optimal in terms of delay if λ is carefully chosen; (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 subsection, we focus on the first four properties and the last one will be discussed in the next subsection.

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

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 equation:

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

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 Eq. (1) as

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

After solving Eq. (2), we get

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

Based on the obtained expressions of x and y, it is not difficult to verify that both x and y are bounded. We omit the verification due to the page limitation. Therefore, the region $\mathcal{U}$ is a bounded area and the influence of Algorithm 1 is local. ■

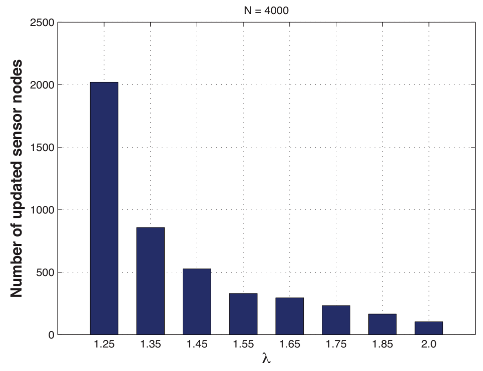


Fig. 3. Local data collection tree updating as suggested by simulations

To verify the conclusion made in Theorem 1, we conduct a set of simulations and illustrate the results in Fig. 3. 4000 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. 3 shows that there are 51% 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 V, we will demonstrate the local influence of our protocol by testbed as well. According to Theorem 1, the local 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 V.

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

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

Due to the space limitation, the detailed proof of Theorem 2 is omitted here. The importance of Theorem 2 is that an efficient local routing tree has been formed in region $\mathcal{U}$ by Algorithm 1, in which any sensor can reach virtual sink v through an optimal routing path if λ is carefully chosen, i.e. $\lambda > 1.25$. As shown in Theorem 3 then, the unchanged portion in the original collection tree $\mathcal{T}_u$ and $\mathcal{U}$ 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 non-modified 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. ■

C. Data Routing

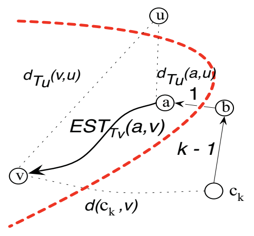
After *Data Collection Tree Updating* completes, a new routing structure is built. If sensor i has data to send or helps other sensors to relay data, it simply transmits data to the neighbor indicated by H_i. Data are guaranteed to be delivered towards the mobile user by Theorem 3. In this subsection, we examine the routing efficiency for the sensors outside $\mathcal{U}$. 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 $\mathcal{U}$ to the virtual sink v is bounded and controlled by λ compared to its optimal routing delay.**

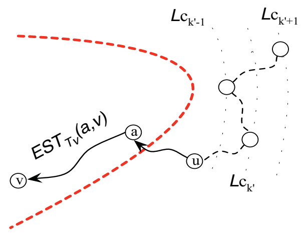
Proof: We first define the *accessing point* of one sensor outside $\mathcal{U}$. The accessing point of sensor i is defined as the *first* encountered sensor in $\mathcal{U}$ if we trace the routing path from sensor i to v in $\mathcal{T}_v$. Then, to show the correctness of Theorem 4, we need to check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

Case I 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. 4(a), where sensor c_k means that this sensor is $k \geq 1$ hops away from its *accessing point*. At a result, the routing path length distortion at sensor c_k can be defined as $\frac{k+EST_{\mathcal{T}_v}(a,v)}{d(c_k,v)}$. Then, we have

$$\begin{aligned}\frac{k+EST_{\mathcal{T}_v}(a,v)}{d(c_k,v)} &\leq \frac{k+EST_{\mathcal{T}_v}(a,v)}{d_{\mathcal{T}_u}(a,u)+k-d_{\mathcal{T}_u}(v,u)}\\ &\leq \frac{k+EST_{\mathcal{T}_v}(a,v)}{\lambda EST_{\mathcal{T}_v}(a,v)+k-2d_{\mathcal{T}_u}(v,u)} \qquad (3)\end{aligned}$$



(a) Case I



(b) Case II

Fig. 4. Two cases in Theorem 4

After a simplification of Eq. (3), we can further derive

$$\text{Eq. (3)} \quad \leq \quad 1+\frac{(k-1)+2d_{\mathcal{T}_u}(v,u)}{d(a,v)+k-2d_{\mathcal{T}_u}(v,u)}. \qquad (4)$$

In Eq. (4), $d_{\mathcal{T}_u}(v,u)$ is a constant. Additionally, as we will show in the next section, $\mathcal{U}$ 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 II captures the scenario in which sensors reach the virtual sink v passing the original virtual sink u as shown in Fig. 4(b). To examine Case II, we first **logically** reorganize all the sensors outside $\mathcal{U}$ by levels. All level k' sensors form a cluster, denoted as $L_{k'}$, satisfying:

$$L_{k'} \quad = \quad \{i \mid i \notin \mathcal{U} \wedge d_{i,\mathcal{U}} = k'\}, \qquad (5)$$

where $k' \geq 1$. $L_{k'}$ contains all the sensors whose minimum distance to $\mathcal{U}$ is k'. In this scenario, the delay distortion at sensor $c_{k'}$ can be expressed as $\frac{d_{\mathcal{T}_u}(c_{k'},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)}$. To prove the correctness of Case II, we apply the constructive mathematical induction on the **level** of each sensor outside region $\mathcal{U}$.

Hypothesis: $\frac{d_{\mathcal{T}_u}(c_{k'},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)} \leq \lambda + \sum_{l=d(a,v)+1}^{d(a,v)+k'} \frac{1}{l} - 1$, where a indicates the accessing point of sensor $c_{k'}$.

Basis: when $k' = 1$. According to line 3 in Algorithm 1, $\frac{d_{\mathcal{T}_u}(c_1,u)+d_{\mathcal{T}_u}(v,u)}{d(c_1,v)} \leq \lambda$. Equivalently, $\frac{d_{\mathcal{T}_u}(c_1,u)+d_{\mathcal{T}_u}(v,u)}{d(c_1,v)} \leq \lambda + 1 - 1$.

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

$$\begin{aligned}\frac{d_{\mathcal{T}_u}(c_{k'+1},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'+1},v)} &\leq \frac{d_{\mathcal{T}_u}(c_{k'},u)+1+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)+1}\\ &\leq (\lambda + \sum_{l=d(a,v)+1}^{d(a,v)+k'} \frac{1}{l} - 1) + \frac{1}{d(c_{k'},v)+1}\\ &= \lambda + \sum_{l=d(a,v)+1}^{d(a,v)+k'+1} \frac{1}{l} - 1 \qquad (6)\end{aligned}$$

Eq. (6) implies that at sensor $c_{k'}$:

$$\frac{d_{\mathcal{T}_u}(c_{k'},u)+d_{\mathcal{T}_u}(v,u)}{d(c_{k'},v)} \quad \leq \quad \lambda + \ln\frac{d(a,v)+k'}{d(a,v)}. \qquad (7)$$

Similar to Case I, 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 form $\mathcal{U}$. ■

According to Theorem 4, the routing path length distortion from any sensor outside region $\mathcal{U}$ 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), through which the mobile user is able to achieve a balance between the routing efficiency and the cost of building the routing structure.

D. 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 towards 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 subsection, we examine the data streaming property of the approach proposed in this paper.

Lemma 1: **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 towards v_k.**

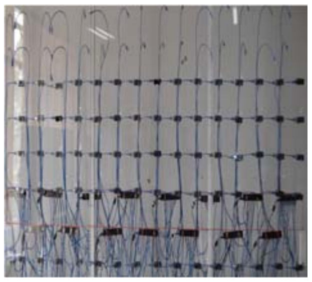
Proof: Before the execution of our protocol, following H_i of each sensor i, uncollected data flow towards v_{k-1}. During the execution of Algorithm 1, the modification of H_i will change the routing destination of sensor i from v_{k-1} to v_k. As a result, the routing destinations of all the ancestors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

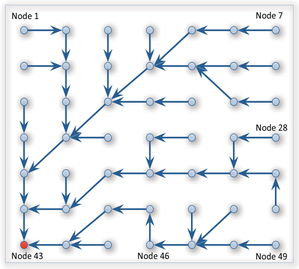
Lemma 1 reveals one critical feature of our proposed protocol. By executing Algorithm 1, region $\mathcal{U}$ behaves like an potential source that attracts data. The data flows always move towards the mobile user. As the mobile user passes by the senor nodes and updates the data collection tree along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

IV. 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.


(a) Testbed



(b) Data collection tree at node 43

Fig. 5. Testbed based experiments

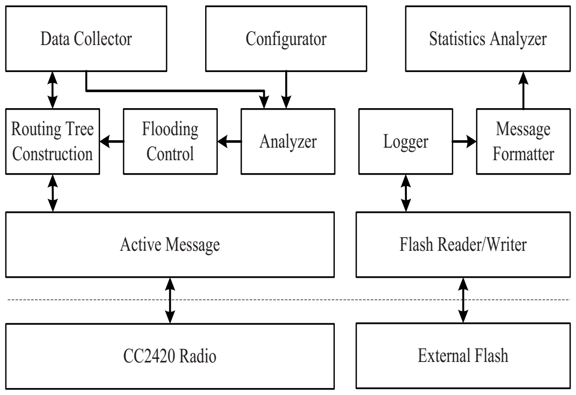


Fig. 6. The diagram of software modules

A. Experiment Setting

We implement our protocol on TelosB motes and use a 49-node testbed to examine its performance, as shown in Fig. 5(a). 49 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 6. 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. 6 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 the sensors, network, and flash, according to the preconfigured message

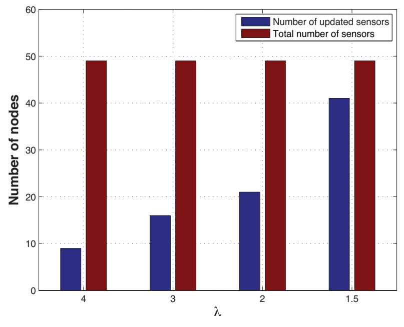


Fig. 7. Updated nodes vs. λ

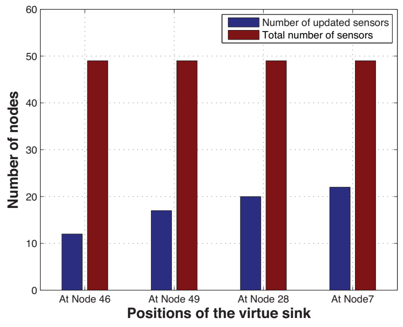


Fig. 8. Updated nodes vs. Moving position

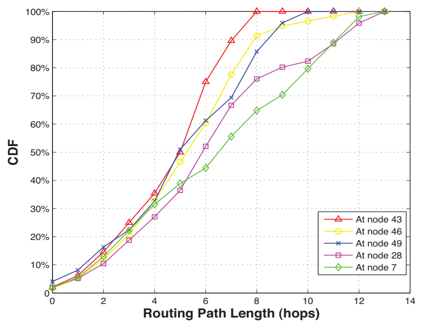


Fig. 9. CDF of routing path lengths

formats.

During the experiment, the mobile user enters the sensor network and builds the first routing tree at node 43, i.e. the left-bottom node. The resulting data collection tree is depicted in Fig. 5(b). In addition, the moving path of the user in our experiment is: node 43 $\Rightarrow$ node 46 $\Rightarrow$ node 49 $\Rightarrow$ node 28 $\Rightarrow$ node 7. During the process, λ is set to be 3.

B. Experiment Results

Investigation on System Parameter λ: Fig. 7 depicts the number of sensor nodes that are updated during updating the data collection tree at node 49. 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. 7, 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%).

Investigation on Updated Areas at Different Positions: Fig. 8 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. 8 provides another good indication that the influence of our protocol is only local.

Investigation on the CDF of Routing Path Lengths: in Fig. 9, we depict the cumulative distribution function (CDF) of the routing path lengths of the sensor nodes in the data collection trees built at different places (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 Fig. 9. According to statistics, the worst routing path length distortion is only 20%. Nevertheless, the cost in building the routing tree can be reduced up to 54%.

V. 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 [7] at each data collection point.

A. 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 $1m/s$. 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 subsections. For each network setting, we perform 20 runs and demonstrate the average performance.

B. Simulation Results

Investigation on System Parameter λ: we first examine the effect of parameter λ of our λ-Flooding approach in a 4000-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. 10. 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. 10 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. 10, 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.

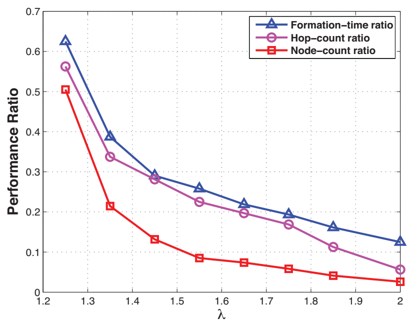


Fig. 10. Updating overhead vs. λ

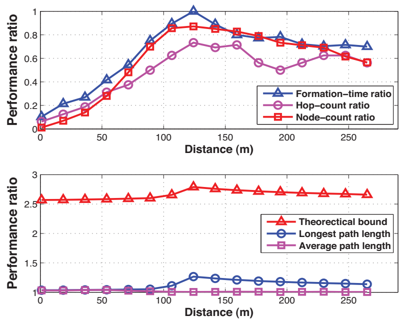


Fig. 11. System performance vs. Distance

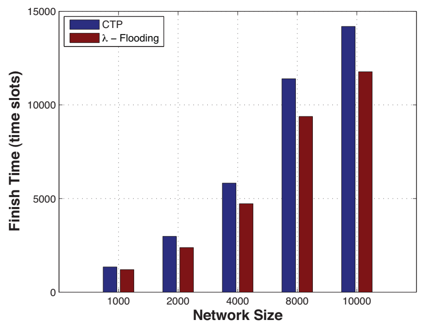


Fig. 12. Finish time of different protocols

System Performance as the Mobile User's Movement: in Fig. 11, 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 filed, to the middle point of the bottom line of the network. There are 5000 sensors deployed in the network and λ is 2.6. Fig. 11 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. 11, the affected area in updating the data collection tree gradually approaches the entire network during the user's movement, i.e., our protocol migrates to building 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. 11. 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. 11. which is loose in this scenario.

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. 12 demonstrates the performance of different protocols with different network sizes. According to the result, λ-Flooding can largely reduce the finish time compared to CTP. As a matter of fact, the streaming property helps shorten the finish time of the λ-Flooding approach.

VI. Related Work

As a basic operation, the data collection in WSNs has been extensively studied. A surge of works study the data accessing and gathering but with static settings. In addition, according to how does each packet transmitted in the network, the data collection can be further divided into two categories: with aggregation or without aggregation. In the former category, in-network aggregating data at various nodes results in a reduction in the amount of bits transmitted over the network, and hence, saves energy. Typical examples include [12], [16], [17]. [12] proposes the first protocol, TAG, such that a simple SQL-like declarative language for expressing aggregation queries in WSNs. In [16], authors study the construction of a data gathering tree to maximize the network lifetime. The problem is shown to be NP-complete. Joo et al. in [17] further study the delay performance of scheduling with data aggregation in WSNs. In the latter category, [11] proposes to collect data through a tree structure with fair rate control. Lin et al. [9] propose to form an information potential based routing structure using harmonic functions. In [8], Challen et al. present Integrated Distributed Energy Awareness (IDEA), a sensor network service enabling effective network-wide data collection framework by introducing a new communication metric called ICTP. Even wireless sensor networks are capable to support large volume data accessing [18], while recent works [13], [14], [19] 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.

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. Xing. et al. [20] propose two algorithms to plan the data collection tours of mobile sinks. Mobile sinks must travel along network routing trees in [20]. [21] further jointly optimizes data routing paths and the data collection tour. Moreover, on the application level, [22] proposes to adopt HST tree to distributed manage resources in sensor networks and [23] 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 [24], 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, [24] is not tailored for the optimization of routing tree transitions. In [13], authors propose to use mobility graphs to predict the future data collection position of the mobile user. [14] utilizes linear programming to optimize the prediction accuracy for mobile users. 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.

VII. Conclusion and Future Work

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.

A possible future work is to further optimize the routing transitions during the movement of the mobile user. In Section V, we can observe that with a fixed λ, the routing efficiency and the updating cost of our approach still has room to be optimized. For instance, when the mobile user roams far away from the first virtue sink, the routing efficiency is approximately optimal as both average path length and longest path length ratios get close to 1. Actually, it is not necessarily needed. In the future, we try to explore an adjusted λ mechanism to break such a barrier. Another possible future work is to apply our approach for low-duty-cycled sensor networks. To prolong the system lifetime, sensors are usually turned off to save energy. In such a scenario, the problem becomes more challenging as the delay increases with the low duty cycle ratio, and our approach in saving routing delay will be of more potential benefits.

Acknowledgement

This work is supported by COE_SUG/RSS_20Aug2010 _13/14 in Nanyang Technological University of Singapore, the NSFC under grants No. 60803152, and National Basic Research Program of China (973 Program) under Grants No. 2011CB302705.

References

[1] L. Mo, Y. He, Y. Liu, J. Zhao, S. Tang, X. Li, and G. Dai, “Canopy closure estimates with greenorbs: Sustainable sensing in the forest,” in *Proceedings of ACM SenSys*, 2009.

[2] T. He, S. Krishnamurthy, J. A. Stankovic, and ect., “An energy-efficient surveillance system using wireless sensor networks,” in *Proceedings of ACM MobiSys*, 2004.

[3] Z. Yang and Y. Liu, “Quality of trilateration: confidence-based iterative localization,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 5, pp. 631–640, 2010.

[4] L. Ni, Y. Liu, Y. Lau, and A. Patil, “LANDMARC: indoor location sensing using active RFID,” *Wireless Networks*, vol. 10, no. 6, pp. 701–710, 2004.

[5] Z. Yang and Y. Liu, “Understanding node localizability of wireless ad-hoc networks,” in *Proceedings of IEEE, INFOCOM*, 2010.

[6] “GreenOrbs,” http://greenorbs.org.

[7] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, “Collection tree protocol,” in *Proceedings of ACM SenSys*, 2009.

[8] G. Challen, J. Waterman, and M. Welsh, “IDEA: Integrated distributed energy awareness for wireless sensor networks,” in *Proceedings of ACM Mobisys*, 2010.

[9] H. Lin, M. Lu, N. Milosavljevic, J. Gao, and L. J. Guibas, “Composable information gradients in wireless sensor networks,” in *Proceedings of ACM, IPSN*, 2008.

[10] Y. Mao, F. Wang, L. Qiu, S. Lam, and J. Smith, “S4: Small state and small stretch routing protocol for large wireless sensor networks,” in *Proceedings of NSDI*, 2007.

[11] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, “Interference-aware fairerate control in wireless sensor netowrks,” in *Proceedings of ACM Sigcomm*, 2006.

[12] S. Michael, M. Franklin, J. Hellerstein, and W. Hong, “TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks,” in *Proceedings of OSDI*. Citeseer, 2002.

[13] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, and L. Guibas, “Predictive qos routing to mobile sinks in wireless sensor networks,” in *Proceedings of ACM, IPSN*, 2009.

[14] H. Lee, M. Wicke, B. Kusy, O. Gnawali, and L. Guibas, “Data stashing: Energy-efficient information delivery to mobile sinks through trajectory prediction,” in *Proceedings of ACM, IPSN*, 2010.

[15] Y. Liu, L. Chen, J. Pei, Q. Chen, and Y. Zhao, “Mining Frequent Trajectory Patterns for Activity Monitoring Using Radio Frequency Tag Arrays,” in *Proc. of IEEE, PerCom*, 2007.

[16] Y. Wu, S. Fahmy, and N. Shroff, “On the construction of a maximum-lifetime data gathering tree in sensor networks: NP-completeness and approximation algorithm,” in *Proceedings of IEEE, INFOCOM*, 2008.

[17] C. Joo, J. Choi, and N. Shroff, “Delay performance of scheduling with data aggregation in wireless sensor networks,” in *Proceedings of IEEE, INFOCOM*, 2010.

[18] X. Li, Y. Liu, S. Li, and S. Tang, “Multicast Capacity of Wireless Ad Hoc Networks Under Gaussian Channel Model,” *IEEE/ACM Transactions on Networking*, vol. 18, no. 4, pp. 1145–1157, 2010.

[19] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, “Routing without routes: The backpressure collection protocol,” in *Proceedings of ACM, IPSN*, 2010.

[20] G. Xing, T. Wang, Z. Xie, and W. Jia, “Rendezvous planning in mobility-assisted wireless sensor networks,” in *Proceedings of IEEE RTSS*, 2007.

[21] G. Xing, T. Wang, W. Jia, and M. Li, “Rendezvous design algorithms for wireless sensor networks with a mobile base station,” in *Proceedings of ACM, Mobihoc*, 2008.

[22] J. Gao, L. Guibas, N. Milosavljevic, and D. Zhou, “Distributed resource management and matching in sensor networks,” in *Proceedings of ACM, IPSN*, 2009.

[23] X. Xu, J. Luo, and Q. Zhang, “Delay tolerant event collection in sensor networks with mobile sink,” in *Proceedings of IEEE, Infocom*, 2010.

[24] J. W. Lee, B. Kusy, T. Azim, B. Shihada, and P. Levis, “Whirlpool routing for mobility,” in *Proceedings of ACM, Mobihoc*, 2010.