This paper was presented as part of the main technical program at IEEE INFOCOM 2011

# SMITE: A Stochastic Compressive Data Collection Protocol for Mobile Wireless Sensor Networks

Longjiang Guo[1,2], Raheem Beyah[2] and Yingshu Li[2†]
[1]School of Computer Science and Technology, Heilongjiang University, Harbin, China 150080
[2]Department of Computer Science, Georgia State University, Atlanta, Georgia 30303
Email: {lguo, rbeyah, yli}@cs.gsu.edu

***Abstract*—Wireless sensors are attached to all kinds of mobile devices/entities such as mobile phones, PDAs, vehicles, robots and animals. This generates Mobile Wireless Sensor Networks (MWSNs) with very dynamic topologies and loose connectivity that depend on mobility of the mobile devices. Data collection from these mobile sensors has become a great challenge considering volatile topologies, loose connectivity and limited buffer storage. This paper proposes a stochastic compressive data collection protocol for MWSNs named SMITE. SMITE consists of three parts: random collector election, stochastic direct transmission from common nodes to collectors when common nodes are in the collectors' transmission range, and angle transmission from collectors to the mobile sink when collectors gather enough data using a predictive method. The collectors use bloom filters to compress the received data. The protocol's performance is theoretically analyzed. The analytic results show that data from the common nodes can be gathered to the collectors with a high probability and gathered data on the collectors can also be forwarded to the mobile sink with a high probability. Simulations are carried out for performance evaluation. The simulation results show that SMITE significantly outperforms the state-of-the-art solutions such as DFT-MSN, SCAR and Sidewinder on the aspects of delivery ratio, transmission overhead, and time delay.**

## I. INTRODUCTION

Recently, researchers have attached wireless sensors to all kinds of mobile devices/entities such as micro-air vehicles [1], bikes [2], ground-based vehicles [3] and animals [4]. This generates Mobile Wireless Sensor Networks (MWSNs) with very dynamic topologies and loose connectivity. Data collection from these mobile sensors becomes a challenging and critical issue in many applications. One typical example of a MWSN is ZebraNet [4], where collars with wireless sensors are attached to zebras' necks to study their habitat and to collect environmental information. This paper focuses on the problem of data collection in MWSNs.

Gathering data from mobile sensors has many challenges. This is because a MWSN is similar to a static Wireless Sensor Network (WSN) with traditional constraints such as limited energy, narrow bandwidth and limited computing ability. Also, it distinguishes itself from conventional static WSNs by the following characteristics:

- **Volatile topology**: The sensors and the sink are attached to all sorts of mobile devices with various types of mobility. The connectivity of the network constantly changes. A mobile sensor is connected to other sensors only randomly and occasionally. The topology is dynamic and volatile at all times. Moreover, network topology of MWSNs changes more acutely than that of mobile ad-hoc networks because sensor radios have a much smaller radio range (*e.g.*, outdoor range $75m \sim 100m$ for TelosB TPR2400CA) compared with mobile ad-hoc networks (*e.g.*, $150m \sim 250m$ in 802.11).
- **Loose connectivity**: The connectivity of MWSNs is low. A sensor node is connected to other sensor nodes only randomly and occasionally due to the nodes' mobility.
- **High message overhead**: Each node needs to monitor a large set of parameters (*multi-dimensional data*), which makes the data size large. A MWSN could consist of hundreds to thousands of sensors. Thus, tremendous amounts of data would be generated and delivered to the sink.
- **Limited buffer space**: Each sensor node has a limited buffer space. This constraint has a significant impact on MWSNs, because sensor nodes store the received data locally prior to relaying the data. Limited buffer space makes it impossible to store a large amount of data locally. Consequently, some useful data is lost and more subsequent communications are incurred.

Conventional routing protocols designed for static WSNs are not suitable for MWSNs because of the aforementioned issues. Even though tree-based routing protocols may be effective for slight topology changes caused by nodes' sleeping and failures [5] [6] [7], these protocols are still not robust enough to conquer the challenge of a volatile topology with mobile nodes. Geographic forwarding-based routing protocols [8] [9] [10] are also not applicable to MWSNs because intermediate nodes do not know where a mobile sink is during the process of forwarding messages.

Several classical routing protocols [12] [13] have been designed for mobile ad-hoc networks. Through extensive experiments, M.Keally *et al.* [14] have validated that these protocols cannot be applied to MWSNs because these protocols need to establish a routing path from a source node to the sink before data is sent. This strategy may work well for slight topology changes, however it is not a good choice for volatile topology.

Some existing works [15] [16] utilize mobile sinks to collect data in static WSNs. However, these works only consider the case where sinks are mobile and sensor nodes are static. They

[1]†To whom correspondences should be addressed.

cannot be employed in MWSNs due to the mobility of both the sinks and the sensor nodes.

Several protocols have been proposed to solve the problem of data collection in MWSNs. In [17], a cluster-based protocol is presented. Each node dynamically chooses its cluster-head based on its current velocity to facilitate data transmission to the nearest cluster-head. A cluster-head is responsible for gathering data and forwarding data to the static sink through a single hop. This assumption is not reasonable, especially for large-scale greatly-dynamic MWSNs where a cluster-head has to send its data to the base station through multiple hops given that the radio range in MWSNs is small.

Three approaches tailored for data gathering in MWSNs are proposed in [18]. One is directed transmission. The basic strategy of directed transmission is to allow data delivery only when sensors are in direct proximity of the sinks. The data delivery ratio of directed transmission depends on the number of sinks. Thus, this technique will not work well if the sinks are sparse. For example, in ZebraNet [4] too many mobile sinks may disturb wildlife habitats and obtain anamorphic environmental data. Another is optimized flooding. This approach has high message overhead. The third one is the DFT-MSN data delivery scheme. However, our results show that the proposed SMITE is better than DFT-MSN on the aspects of delivery ratio, message overhead, and time delay.

In [19], a routing approach named SCAR is presented. SCAR uses prediction techniques over the context of a mobile sensor node (such as previously encountered neighbors, energy level and locations of neighboring nodes) to forecast which of the neighbors are the best carriers for data collection. However, this approach requires that each mobile sensor node periodically broadcasts its context information to its neighbors. This incurs high message overhead causing significant energy consumption. In [14], a routing protocol named Sidewinder is presented. Sidewinder forwards data packets based on sink location prediction and estimation techniques. The prediction module of Sidewinder requires the sink's neighbors' average velocity. Similar to SCAR, this mechanism also invokes high message overhead since the sink is mobile and each sensor node must send its velocity to the sink when the sensor node is in direct proximity of the sink. The situation is even worse for multi-dimensional data, causing short network lifetime.

Motivated by the aforementioned challenges, the paper proposes SMITE, a stochastic compressive data collection protocol for MWSNs with only one mobile sink. SMITE consists of three parts: random collector election among all the common nodes, stochastic direct transmission from the common nodes to the collectors when the common nodes are in the collectors' transmission range, and angle transmission from the collectors to the sink using a predictive method through multi-hops after the collectors gather enough data. The collectors use a bloom filter to compress the data into $L$ bits before its transmission to the sink. The main contributions are the following:

- SMITE transforms the traditional data collection problem in MWSNs into in-network data aggregation through a data compression technique using bloom filters. Data collected by the collectors can be aggregated into a packet of $L$ bits and the aggregated packet is then forwarded to the mobile sink through multiple hops instead of forwarding each collected data packet directly to the mobile sink. The sink can restore the collected data from the aggregated packet with a low false positive rate. With this design, the proposed scheme is also able to reduce the total communication overhead.
- SMITE is completely different from the traditional idea where every node periodically broadcasts its context information to its neighbors to maintain real context. In SMITE, only collectors periodically broadcast their context information. Further, only a small subset of the sensor nodes are collectors at any time. Each common node only sends its data to a collector and receives context information from a collector when the common node can communicate with a collector. Common nodes do not need to periodically broadcast their context information. Moreover, SMITE does not require the mobile sink to periodically completely flood its context information to the whole network. Only partial flooding is necessary in SMITE, *i.e.*, the mobile sink's context information is only flooded to $h_{max}$ ($h_{max}$ is a small number relative to the total number of nodes) hops away from the mobile sink. This scheme can greatly reduce message overhead and guarantee the delivery ratio with a high probability.
- The performance of SMITE is analyzed through probability and Markov Chains theories. The analytic results show that data on the common nodes can be transmitted to collectors with a high probability and complete data gathered on collectors can be further forwarded to the mobile sink with a high probability.
- Extensive simulations with NS2 are carried out for performance evaluation. The simulation results demonstrate that SMITE significantly outperforms the state-of-the-art solutions such as SCAR and Sidewinder on the aspects of delivery ratio, transmission overhead, and time delay.

The rest of the paper is organized as follows: Section II introduces the main idea of SMITE. Section III explains the detailed design of SMITE. Section IV shows the theoretical analysis of SMITE's performance. Simulation results are presented and discussed in Section V. Finally, Section VI concludes the paper.

## II. SMITE OVERVIEW

We consider a MWSN consisting of one mobile sink and $N$ mobile sensor nodes randomly moving in an area of $Z \times Z$, where $Z$ is the width of the monitored area. The sensor nodes and the sink set their radio transmission range as $r$. We assume that nodes are synchronized and each node knows its velocity as well as its position using GPS. Time is divided into slots of fixed length. Several continuous time slots form a round. Data should be forwarded to the sink during each round. The length of each round is the same and how the length of each round is determined will be discussed in Section IV.

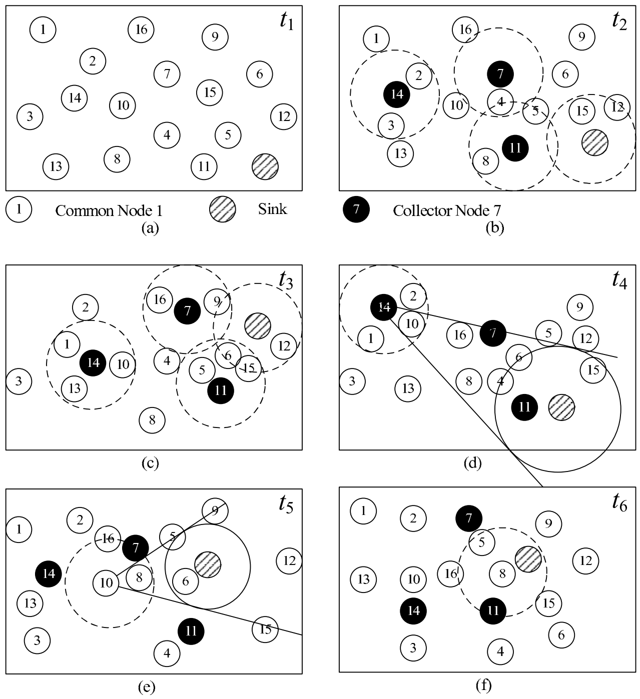


Fig. 1. SMITE overview.

SMITE has three phases as shown in Fig.1. The mobile sink always periodically partially floods its context information (such as position, velocity and time stamp) in the network to reduce message overhead (partial flooding is discussed in Section III-B and Section IV). Each node builds a predictive model for the sink's velocity according to the received sink's context information. In Fig.1(a), we assume a round consists of 4 time slots.

The first phase is collector election. After the deployment of sensor nodes in time slot $t_1$, each node locally executes the collector election algorithm without any communication. Each node determines with a probability $p$ whether it will become a collector or not in the current round. It is obvious that the mobile sink is always a collector.

The second phase is direct data transmission from non-collectors to collectors while non-collectors are in the communication range of collectors. In Fig.1(b), black nodes 7, 11 and 14 become collectors and they periodically broadcast a message to announce this fact. In time slot $t_2$, nodes 2, 3, 4, 8 and 15 are in the communication range of the collectors. When they receive the message from the collectors respectively, they send their data to the collectors. Similarly, in time slot $t_3$ as shown in Fig.1(c), nodes 1, 10, 13, 9, 16, 5, 6, 15 and 12 are in the communication range of the collectors. Thus, these nodes send their data to their respective collectors. Notice that node 15 does nothing, since its data has already been collected in time slot $t_2$. Collectors 7, 11 and 14 adopt the compressive strategy bloom filters to hash the received data into a bloom filter of $L$ bits which is no larger than the size of a packet. Thus, the collected data can be aggregated.

The third phase is data forwarding from the collectors to the mobile sink. Once a bloom filter on a collector is full (*i.e.*, if one more data element is hashed into the bloom filter, the false positive probability will exceed the predefined threshold) or it is at the end of a round, then a collector forwards the aggregated compressed data packet to the mobile sink. In Fig.1(d), assume time slot $t_4$ is the last time slot of a round and collector 14 is full, then collector 14 employs angle transmission (discussed in Section III-B) to forward its collected aggregated data packet to the sink. Collector 14 makes use of the predictive model to predict the sink's movement territory which is a circle centered at the most recently received sink's position. The radius of the circle is determined by the predictive model (details are presented in Section III-B). Two tangents are elicited from 14 to the circle. Then, there are two points of tangency and an angle at collector 14 towards the sink. Collector 14 broadcasts the compressed data packet, the sink's recent position and its position to its neighbors. Each of 14's neighbors determines whether it is in the angle. If so, a neighbor does the same as 14 does. Otherwise, a neighbor discards the message received from 14. For example, in Fig.1(d), node 10 is in the angle, then node 10 does the same as 14 and forwards the received message to its neighbor node 8 at time slot $t_5$ as shown in Fig.1(e). Nodes 1 and 2 are not in the angle, so they simply discard the message received from node 14 as shown in Fig.1(d). Node 8 finally sends the data to the sink at time slot $t_6$ as shown in Fig.1(f).

## III. DETAILED SMITE DESIGN

This section presents the detailed design of SMITE. The three phases of SMITE are discussed in Section III-A and Section III-B. In Section III-C, we explain the data compression and buffer management techniques. Section III-D introduces the rotation mechanism of the collectors.

### A. Collector Election and Direct Transmission to Collectors

Initially, each mobile node sets its initial role as **COMMON**. For each common node $u$ with a probability $p$ to be a collector, where $p$ is a system parameter such that $0 < p < 1$ (Proposition 3 discusses how to set $p$), $u$ generates a random number $x$ between 0 and 1. If $0 \le x \le p$, then $u$ will be a collector and $u$ updates its role as **COLLECTOR**. Otherwise, if $p < x \le 1$, $u$ is still a common node.

*Proposition 1:* Denote a random variable $X$ to be the number of the collectors, then $X$ obeys binomial distribution with parameters $N$ and $p$, where $N$ is the total number of the nodes. $\Pr(X = 0) = (1 - p)^N$. $\Pr(X = 0)$ closes to 0 if $N$ is large.

*Lemma 2:* If $N \ge \frac{3}{p \cdot \varepsilon^2} ln(2/\delta)$, then $N \cdot p$ provides $(\varepsilon, \delta)$-approximation for the number of the collectors. That is, for any $0 < \varepsilon, \delta < 1$, $\Pr(|X - N \cdot p| \ge \varepsilon N \cdot p) \le \delta$.

**Proof**. Let random variable $X_u$ denote the role of node $u$. $X_u$=0 denotes that $u$ is a common node; $X_u$=1 denotes that $u$ is a collector. $\Pr(X_u = 0) = 1 - p$; $\Pr(X_u = 1) = p$. Thus, $X = \sum_{u=1}^{N} X_u$ and $\mathbf{E}(X)$= $\sum_{u=1}^{N} \mathbf{E}(X_u)$=$N \cdot p$. $X_1, X_2, \cdots, X_N$ is a sequence of independent Poisson trials with $\Pr(X_u = 1) = p$

($u = 1, 2, \cdots, N$) since each node is independently likely to be a collector with probability $p$. According to Chernoff bounds estimation for the sum of Poisson trials [22], for any $0 < \varepsilon < 1$, $\Pr(|X - N \cdot p| \geq \varepsilon N \cdot p) \leq 2e^{-\frac{N \cdot p \cdot \varepsilon^2}{3}}$. For any $0 < \delta < 1$, if $N \geq \frac{3}{p \cdot \varepsilon^2} ln(2/\delta)$, *i.e.* $2e^{-\frac{N \cdot p \cdot \varepsilon^2}{3}} \leq \delta$. Thus $N \cdot p$ gives $(\varepsilon, \delta)$-approximation for the number of the collectors. ■

Proposition 1 shows that having no collector is almost impossible. Based on Lemma 2, $N \cdot p$ can be $X$'s estimation. The probability of the case where the absolute error between $N \cdot p$ and $X$ is larger than a small number is very small.

Once a common node $u$ has become a collector, $u$ periodically broadcasts a message **COLLECT**($u$) to its neighbors, where the information in the brackets denotes the message's contents and $u$ denotes $u$'s ID. If a collector receives a **COLLECT** message, it discards it. If a common node $v$ receives **COLLECT**($u$) from $u$, then $v$ sends its local compressed multi-dimensional data **DATA**($m_v$) to $u$, where $m_v$ is $v$'s local compressed data. How to compute $m_v$ will be discussed in Section III-C. Once a common node $v$ has already sent its data to a collector, $v$ sets its state as **COL**. If $v$'s state is **COL**, then $v$ discards the **COLLECT** message. A node's radio coverage range is a circle, when common node $v$ sends **DATA**($m_v$) to collector $u$, all of $v$'s neighbors in its coverage range can receive this packet. For any node $w$, upon receiving compressed data packet **DATA**($m_v$) from $v$, if $w$ is a collector, then $w$ stores $m_v$ in its local buffer (buffer management is discussed in Section III-C); if $w$ is a common node, and $w$ has already sent its data to a collector, then $w$ discards **DATA**($m_v$); if $w$ is a common node, and $w$ does not send its data to a collector, then $w$ stores $m_v$ in its local buffer as a replica.

For example, in Fig.1(b), when common node 3 sends its data $m_3$ to collector 14, node 3's neighbor 13 can receive $m_3$. Since node 13 has not sent its data to a collector, node 13 stores $m_3$ in its local buffer as a replica. In Fig.1(c), when node 13 is in the coverage range of collector 14, node 13 sends its data $m_{13}$ and 3's replica $m_3$ together in one packet to 14 (if the buffer at node 13 is full, it hashes data into a bloom filter and does not receive any replica as explained in Section III-C). This replication mechanism provides compensation in case of transmission failure of node 3. Because in Fig.1(b), if nodes 2 and node 3 send data in the same time slot, $m_3$ may not arrive at collector 14 successfully due to a transmission collision. Though in Fig.1(c), node 13 sends $m_{13}$ and $m_3$ together in one packet to collector 14, this does not incur extra communication cost. In Fig.1(c), when node 6 is in the coverage range of collector 11, it sends $m_6$ to 11. Then 6's neighbor 15 can receive $m_6$, and 15 discards $m_6$. This is because in Fig.1(b), $m_{15}$ has already been collected by the sink and node 15 already set its state to **COL** in time slot $t_2$.

### *B. Angle Transmission From Collectors To Mobile Sink*

The general idea of angle transmission was presented in Section II. This section further clarifies four issues: 1) How does the sink partially flood its context information in the network; 2) How does the node predict the sink's movement territory according to the sink's historical context information; 3) How does a node estimate the forwarding angle; 4) How does a node determine whether it is in a transmission angle?

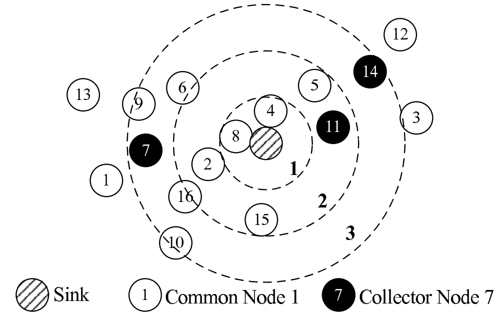


Fig. 2. The location corelation between a relayed node and the sink.

The purpose of partially flooding the sink's context information is to help all the nodes estimate the sink's movement territory. The mobile sink periodically floods its message **VP**($v_t$, $x_t$, $y_t$, $h_s$) only to the nodes which are at most $h_{max}$ hops away from the sink rather than fully flooding the message in the entire network. Here, $v_t$ and ($x_t$, $y_t$) are the sink's velocity and its position at time slot $t$ respectively. $h_s$ is a variable and it indicates the location corelation between a relayed node and the sink. $h_s$'s value is the serial number of the homocentric circles centered at the sink where the relayed node locates. Its initial value is 0 since the sink is the first relayed node and the sink is the center. The distance between the boundaries of two adjacent homocentric circles equals the communication range $r$ (see Fig.2). For any relayed node $w$, it decides the value of $h_w$, $w$'s location corelation to the sink as follows: $h_w \leftarrow \lceil \frac{\sqrt{(x_w - x_t)^2 + (y_w - y_t)^2}}{r} \rceil$, where $(x_w, y_w)$ denotes $w$'s position and $r$ is the communication range. For example, in Fig.2, node 5 locates in the second homocentric circle centered at the sink, and $h_5$=2. Similarly, $h_8$=1, $h_7$=3.

For any relayed node $w$, it maintains the recent $k$ sinks' velocity information $v_{t_1}$, $v_{t_2}$, ..., $v_{t_k}$, and the most recent sink's position ($x_{t_k}$, $y_{t_k}$), where $t_k$ is the most recent time slot. Upon receiving **VP**($v_t$, $x_t$, $y_t$, $h_u$) from node $u$, if $t \neq t_j$ for any $j \in [1, k]$, it means the received **VP** is a new one, then: (i). Node $w$ deletes the oldest one and inserts $v_t$ into local velocity information queue in an increasing order of time stamps; (ii). If $t > t_k$, then $w$ replaces ($x_{t_k}$, $y_{t_k}$) with ($x_t$, $y_t$); (iii). If $\exists i \in [1, k]$ such that $t = t_i$, then it means node $w$ has already received the **VP**, and node $w$ drops the received **VP**. After updating the sink's information, node $w$ decides whether to forward the received **VP**. First, $w$ calculates $h_w$, the location coelation to the sink. If $h_u < h_w \leq h_{max}$ and received **VP** is a new message, then $w$ continues to broadcast **VP**($v_t$, $x_t$, $y_t$, $h_w$) to its neighbors. Otherwise, $w$ drops the received packet. For example, in Fig.2, when nodes 2 and 4 receive **VP**($v_t$, $x_t$, $y_t$, 1) from node 8, node 2 continues to forward **VP**($v_t$, $x_t$, $y_t$, 2) and node 4 drops the packet. When node 8 receives **VP**($v_t$, $x_t$, $y_t$, 2) from node 2, node 8 discards the packet. This flooding mechanism prevents message transmission expansion through

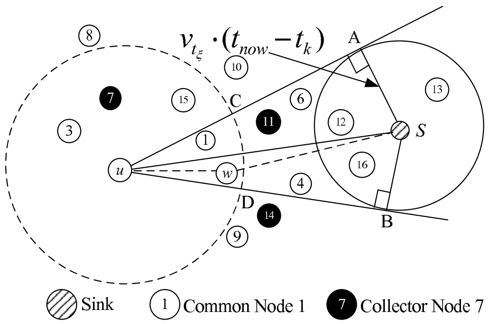


Fig. 3. Prediction of the sink's movement territory.

eliminating redundant inverted message transmissions back to the sink. Corollary 8 in Section IV shows that any node which is less than $h_{max}$ hops away from the sink can receive message **VP** with a high probability.

Node $w$ can predict the sink's movement territory according to the recent $k$ sink's velocities $v_{t_1}, v_{t_2}, \ldots, v_{t_k}$ and the most recent position ($x_{t_k}$, $y_{t_k}$). Suppose the sink's velocity is a continuous function over time $t$ denoted by $v(t)$. Then, from $t_k$ to current time slot $t_{now}$, the sink's movement distance can be calculated according to Integral Mean Value Theorem as follows:

$$\int_{t_k}^{t_{now}} v(t)dt = v(t_\xi)(t_{now} - t_k), t_\xi \in [t_k, t_{now}]$$

Thus, the sink's movement territory is a disk centered at ($x_{t_k}$, $y_{t_k}$) with radius $v(t_\xi)(t_{now} - t_k)$(see Fig. 3). Without loss of generality, we use $v_{t_\xi}$ to substitute $v(t_\xi)$. The remaining task is to predict $v_{t_\xi}$ using $v_{t_1}, v_{t_2}, \ldots, v_{t_k}$.

First, node $w$ transforms the time series $v_{t_1}, v_{t_2}, \ldots, v_{t_k}$ into stationary time series by taking the second difference transformation as the latter formulation. Experience indicates that using the second difference transformation will usually produce a stationary time series [20].

$$\begin{cases} W_{t_i} = (v_{t_i} - v_{t_{i-1}}) - (v_{t_{i-1}} - v_{t_{i-2}}); (i = 1, 2, ..., k) \\ W_{t_1} = v_{t_1}; W_{t_2} = v_{t_2} \end{cases}$$

Next, for time series $W_{t_1}, W_{t_2}, \ldots, W_{t_k}$, node $w$ uses the weighted moving average [20] to predict $\hat{W}_{t_i}$, where $\hat{W}_{t_i}$ is the estimated value for $W_{t_i}(i{=}k{+}1, k{+}2,...,\xi)$. A weighted moving average is simply a moving average that is weighted so that more recent values are more heavily weighted than values further in the past. The most common type of weighted moving average is exponential smoothing [20] as follows:

$$\begin{cases} \hat{W}_{t_1} = W_{t_1}; \hat{W}_{t_2} = W_{t_2}; ...; \hat{W}_{t_k} = W_{t_k} \\ \hat{W}_{t_i} = (1-\lambda)\lambda^{k-1}\hat{W}_{t_{i-k}} + ... + (1-\lambda)\lambda^{k-k}\hat{W}_{t_{i-1}} \\ i{=}k{+}1, k{+}2, ..., \xi. \end{cases}$$

Here, $\lambda$ is a system parameter and its initial value is randomly drawn from [0, 1]. $\xi$ is a random number from [$k$+1, $now$], $now$ is the current time slot. Thus, node $w$ can convert $\hat{W}_{t_i}(i{=}k{+}1, k{+}2,...,\xi)$ to $\hat{v}_{t_i}(i{=}k{+}1, k{+}2,...,\xi)$ based on the equation: $\hat{v}_{t_i} = \hat{W}_{t_i} - \hat{v}_{t_{i-2}} + 2\hat{v}_{t_{i-1}}(i{=}k{+}1, k{+}2,...,\xi)$. Here $\hat{v}_{t_i}$ is the estimated value for $v_{t_i}$. If $i \leq k$, $\hat{v}_{t_i} = v_{t_i}$. Otherwise, $\hat{v}_{t_i} = \hat{v}_{t_i}$. When node $w$ receives the real $v_{t_i}$, node $w$ can adjust $\lambda$ such that $\hat{v}_{t_i}$ approaches to $v_{t_i}$ as close as possible. Thus, node $w$ can get an optimal parameter $\lambda$ through this incessant self-leaning mechanism.

During the process of angle transmission from a collector to the mobile sink, if a node $w$ receives a collected data packet from node $u$, then $w$ should determine whether it is in the angle shaped by $u$ and two tangents elicited from $u$ to the sink's movement territory estimated by $u$. In Fig. 3, an example of such an angle is $\angle AuB$. The angle can determine the scope of the sink's movements from $u$'s point of view. If $u$ forwards a collected data packet to all of its neighbors then any of $u$'s neighbors can determine whether it is in the angle $\angle AuB$. Nodes 1 and $w$ are in $\angle AuB$, then nodes 1 and $w$ continue to forward the collected data packet. Nodes 3, 7 and 15 are not in $\angle AuB$, then they will discard the received data packet.

The main issue is that how can node $w$ determine whether it is in the angle $\angle AuB$ while $w$ receives the packet from node $u$. To do so, besides the collected data packet, node $u$ also broadcasts its current position ($ux_{t_{now}}$, $uy_{t_{now}}$), its recent received sink's position ($sx_{t_k}$, $sy_{t_k}$) and the estimated radius of the sink's movement territory centered at ($sx_{t_k}$, $sy_{t_k}$), $\hat{r}_u{=}v(t_\xi)(t_{now} - t_k)$. Suppose $w$'s current position is ($wx_{t_{now}}$, $wy_{t_{now}}$), if $w$ receives ($ux_{t_{now}}$, $uy_{t_{now}}$), ($sx_{t_k}$, $sy_{t_k}$) and $\hat{r}_u$ from $u$, then $w$ can perform the following computations to judge whether it is in $\angle AuB$:

$$\begin{cases} |uw| \leq r \\ \arccos(\frac{|uw|^2+|us|^2-|ws|^2}{2|uw|\cdot|us|}) \leq \arcsin(\frac{\hat{r}_u}{|us|}) \end{cases}$$

where $r$ is the communication range; $|uw|$, $|us|$ and $|ws|$ denote the Euclidian distances between each pair of the nodes $u$, $w$ and $s$. They can be easily computed since their positions are already known for node $w$. If both of the above two equations hold, then $w$ is in $\angle AuB$.

### C. Data Compression and Buffer Management

Data generated by any common node $v$ is usually multi-dimensional in the form of ($c_1$, $c_2$, $\cdots$, $c_k$) where each $c_i$ ($i = 1, \cdots, k$) represents different parameters such as temperature, light, humidity, *etc*. Suppose $c_i$'s domain size is $D_i$, then ($c_1$, $c_2$, $\cdots$, $c_k$) can be mapped into a value $cm$ using the equation: $cm \leftarrow \sum_{i=1}^{k-1}(c_i \cdot \prod_{j=i+1}^{k} D_j) + c_k$. Here, $cm$ could be a large number since $\prod_{j=i+1}^{k} D_j$ may be a large number if the dimension size $k$ is large. To store $cm$ would consume a large amount of space despite the fact that the storage space for $cm$ is less than that for ($c_1$, $c_2$, $\cdots$, $c_k$). We adopt the following transformations to reduce the storage space for $cm$:

$(p.q) \leftarrow \log_2(cm); \quad m_v \leftarrow q \cdot D_I + p;$

where . between $p$ and $q$ is the decimal point and $p$ is the integral part and $q$ is the decimal part with 4 significant digits after the decimal point. $D_I$ is the domain size of the integral part. Usually, $D_I = 64$, thus we can use 2 bytes to store $m_v$. We can easily extract ($c_1$, $c_2$, $\cdots$, $c_k$) from $m_v$ as follows:

$p \leftarrow m_v - \left[\frac{m_v}{D_I}\right] \cdot D_I$; $q \leftarrow \frac{m_v - p}{D_I}$; $cm \leftarrow 2^{(p.q)}$;
$cm_k \leftarrow cm$; $c_k \leftarrow cm_k - \left[\frac{cm_k}{D_k}\right] \cdot D_k$;
$cm_{k-1} \leftarrow \frac{cm_k - c_k}{D_k}$; $c_{k-1} \leftarrow cm_{k-1} - \left[\frac{cm_{k-1}}{D_{k-1}}\right] \cdot D_{k-1}$;
......
$cm_1 \leftarrow \frac{cm_2 - c_2}{D_2}$; $c_1 \leftarrow cm_1$;

During the process of data collection, when a common node $v$ receives a **COLLECT** message from a collector $w$, $v$ sends compressed data message **DATA**($m_v$) to $w$. This can cut down the packet size and reduce total message overhead.

Upon receiving **DATA**($m_v$) from node $v$, collector $w$ stores $m_v$ in its local buffer. The core idea of local buffer management is to first use an array to store the received compressed data without false positives. If the array is full, $w$ uses a bloom filter [21] to store more received data with a tolerant false positive probability $\eta$. If the bloom filter is full, $w$ starts the angle transmission.

The local buffer initially is an array that consists of $x$ elements and each element can store the received compressed data. Here, $x$=$\frac{N-N\cdot p}{N\cdot p}+1$=$\frac{1}{p}$, where $N$ is the number of all the nodes; $p$ is a probability with which each common node is a collector. According to Proposition 1, this means each collector on average collects $x$ data items, and the total number of collected data items is $x \cdot Np$=$N$. If the array is full, then collector $w$ maintains its data buffer as follows: (i) $w$ creates a bloom filter **BF**($L$, $k$, $\eta$, $min$, $max$, $n$) with $L$ bits, $k$ independent hash functions $h_1, h_2, \cdots, h_k$ and a tolerant false positive probability $\eta$. The bloom filter **BF** representing a set of $n$ compressed data $\{m_1, m_2, \cdots, m_n\}$ is described by a vector of $L$ bits. $min$ and $max$ denote the minimum and maximum value among $\{m_1, m_2, \cdots, m_n\}$, $n$ denotes the number of data items hashed into **BF**. (ii) $w$ hashes the $x$ elements into **BF**. (iii) $w$ releases the space of the array.

The above buffer management mechanism is also applicable to the data replication mechanism on common nodes discussed in Section III-A. Here, we set $L$=16$x$=$\frac{16}{p}$, then the **BF** does not requires extra storage space since each compressed data item takes 16 bits. The **BF** uses $k$ independent hash functions to map each data item $m_i$ ($1 \leqslant i \leqslant n$) to a sequence of random numbers $h_1(m_i)$, $h_2(m_i)$, $\cdots$, $h_k(m_i)$ over a range $\{1, \cdots, L\}$ uniformly. For any data item $m_i$, the bits $h_1(m_i)$, $h_2(m_i)$, $\cdots$, $h_k(m_i)$ are set to 1 when inserting $m_i$ into **BF**.

When we extract data from **BF**, we just need to check whether an element $m$ from $min$ to $max$ stepped by 1 is in **BF**. One just needs to judge whether all the $h_i(m)$ ($1 \leqslant i \leqslant k$) bits are set to 1. If so, we report $m$ in the **BF**. However, there is a probability that this could be wrong (*i.e.*, false positive). The false positive probability is:

$$f_{L,k,n}^{BF} = (1-(1-\frac{1}{L})^{n\times k})^k = (1-(1-\frac{p}{16})^{n\times k})^k$$

Given a tolerant false positive probability $\eta$, if there exists a maximum $n_{max}$ such that $f_{L,k,n_{max}}^{BF} \leq \eta$, we call $n_{max}$ as the *capacity* of **BF**. This mechanism is equivalent to aggregating many data items into one data structure. We hope that (a) the size of **BF** is no more than the packet size $S$ (typically 320 bits), *i.e.* $L$=$\frac{16}{p} \leqslant S$ and (b) with the same space size, **BF** can accommodate more data items than an array with lower false positive probability, *i.e.* $x = \frac{1}{p} \leqslant n_{max} \leqslant \frac{\ln(\frac{1}{1-\sqrt[k]{\eta}})}{k\ln(\frac{1}{1-p/16})}$.

*Proposition 3:* Given $\eta$, $k$ and packet size $S$, a proper $p$ can be found such that requirements (a) and (b) can be satisfied.

For example, if $k$=5, $\eta$=2%, $S$=320, then $p$=5% $n_{max}$=39≈2·$x$ > $x$=$\frac{1}{p}$=20.

Upon receiving **DATA**($m_v$) from common node $v$, node $w$ stores $m_v$ in its local data array. If its array is full, $w$ hashes $m_v$ into its local **BF** and increases $n$ by 1. If $n > n_{max}$, *i.e.* the **BF** is full, then (i) if $w$ is a collector, $w$ starts angle transmission; (ii) if $w$ is a common node, $w$ discards **DATA**($m_v$) since its data buffer is full.

### *D. Rotation of The Collectors*

SMITE adopts randomized rotation of the collectors to evenly distribute the energy consumption load among the sensor nodes. Once a collector $u$ finishes the angle transmission, it turns its role from **COLLECTOR** to **COMMON**. If the sink receives $y$ data packets from $y$ collectors during a round, then there are $y$ collectors which turn their roles to **COMMON**. To maintain the data delivery ratio, $y$ collectors should be randomly re-elected from $N$-$N \cdot p$+$y$ common nodes. Before the sink broadcasts the **VP** message, it adds $y$ into **VP**. When a common node $w$ receives **VP**, $w$ turns into **COLLECTOR** with a probability $g$=$\frac{y}{N-N\cdot p+y}$+$\frac{e_w}{N\cdot e}$, where $e_w$ is $w$'s residual energy and $e$ is $w$'s initial energy. If $w$ has more residual energy, then $w$ becomes a collector with a high probability.

*Proposition 4:* Let random variable $Y$ denote the number of the re-elected collectors, then $\mathbf{E}(Y)$=$y$.

**Proof**. Let random variable $Y_w$ denote the role of node $w$ during the process of collectors re-election. $Y_w$=0 denotes that $w$ is a common node; $Y_w$=1 denotes that $u$ is a collector. $\Pr(Y_w$=0)=1-$g$;$\Pr(Y_w$=1)=$g$. Thus, $Y$=$\sum_{w=1}^{N-N\cdot p+y} Y_w$ and $\mathbf{E}(Y)$=$\sum_{w=1}^{N-N\cdot p+y} \mathbf{E}(Y_w)$=$y$+$\sum_{w=1}^{N-N\cdot p+y} \frac{e_w}{N\cdot e}$< $y$+1. Since $\mathbf{E}(Y)$ is an integer, thus $\mathbf{E}(Y)$=$y$. ■

## IV. Theoretical Analysis and Optimization

This section proposes a probability model using Markov Chains to depict common nodes' movement and presents some analytic results to show data on common nodes can be transmitted to collectors with a high probability and all the received data on collectors can be forwarded to the mobile sink also with a high probability. Based on the analytical results, the length of a round can be determined.

### *A. From Common Nodes To Collector Nodes*

For any common node $w$ at time slot $t$, $w$ has two possible states: one is that $w$ receives a **COLLECT** message from a collector $u$ and it sends its data to $u$, this state is called **COL**; another is that $w$ never receives any **COLLECT** message from any collector, we call this state **UNCOL**. Random variable

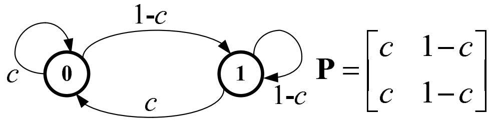


Fig. 4. A Markov chain for a common node movement.

$X_{w,t}$ denotes common node $w$'s state at time slot $t$. $X_{w,t}$=0 denotes $w$ stays in **UNCOL** state. $X_{w,t}$=1 denotes $w$ stays in **COL** state. $\{X_{w,t}, t \in [1, \infty)\}$ is a discrete Markov chain, because $X_{w,t}$ satisfies the following equation, where $a_t \in \{0, 1\}$: $\Pr(X_{w,t} = a_t | X_{w,t-1} = a_{t-1}, X_{w,t-2} = a_{t-2}, \cdots, X_{w,1} = a_1) = \Pr(X_{w,t} = a_t | X_{w,t-1} = a_{t-1}) = \Pr(X_{w,t} = a_t)$. This is obvious since every common node moves randomly, and its current state is independent of the history during which $X_{w,t-1}$ is arrived.

Now we compute $\Pr(X_{w,t} = a_t), a_t \in \{0, 1\}$. For a collector $u$, the probability of the case that any common node $w$ stays in $u$'s coverage area is $b = \frac{\pi r^2}{Z^2}$, where $r$ is the communication range and $Z$ is the width of the monitored area. According to Proposition 1, there are on average $N \cdot p$ collectors. On average, the probability of the case that any common node $w$ stays in **UNCOL** state, *i.e.*, the probability of the case that $w$ does not stay in any collector's coverage area is $\Pr(X_{w,t} = 0)$=$(1 - b)^{(N \cdot p)}$=$c$; the probability of that any common node $w$ stays in **COL** state, *i.e.*, the probability of that $w$ stays in at least one collector's radio coverage area is $\Pr(X_{w,t} = 1)$=$1 - (1 - b)^{(N \cdot p)}$=$1 - c$. Thus, $\Pr(X_{w,t} = 0 | X_{w,t-1} = 0)$=$\Pr(X_{w,t} = 0 | X_{w,t-1} = 1)$=$c$; $\Pr(X_{w,t} = 1 | X_{w,t-1} = 0)$=$\Pr(X_{w,t} = 1 | X_{w,t-1} = 1)$=$1 - c$. Fig. 4 shows this Markov chain and its transition matrix. For simplicity, in Fig. 4, 0 denotes **UNCOL** and 1 denotes **COL**. Proposition 5 estimates how many common nodes stay in **COL** state in one time slot.

*Proposition 5:* If $N \geq \frac{3}{(1-p)\cdot(1-c)\cdot\varepsilon^2} ln(2/\delta)$, then $N \cdot (1 - p) \cdot (1 - c)$ gives $(\varepsilon, \delta)$-approximation for $X_t$, the number of the common nodes which stay in **COL** state during time slot $t$. That is, for any $0 < \varepsilon, \delta < 1$, $\Pr(|X_t - N \cdot (1 - p) \cdot (1 - c)| \geq \varepsilon N \cdot (1 - p) \cdot (1 - c)) \leq \delta$.

**Proof**. According to Lemma 2, there are $N - N \cdot p$ common nodes. $X_t = \sum_{w=1}^{N-N\cdot p} X_{w,t}$ and $\mathbf{E}(X_t)$=$\sum_{w=1}^{N-N\cdot p} \mathbf{E}(X_{w,t})$=$N \cdot (1 - p)(1 - c)$. Since $X_{1,t}$, $X_{2,t}$, $\cdots$, $X_{N-N\cdot p,t}$ is a sequence of independent Poisson trials with $\Pr(X_{w,t} = 1) = 1 - c$, according to Chernoff bounds estimation for the sum of Poisson trials in [22], for any $0 < \varepsilon < 1$, and $\mathbf{E}(X_t)$=$N \cdot (1-p)(1-c)$, $\Pr(|X_t - \mathbf{E}(X_t)| \geq \varepsilon \mathbf{E}(X_t)) \leq 2e^{-\frac{\mathbf{E}(X_t)\cdot\varepsilon^2}{3}}$. For any $0 < \delta < 1$, if $N \geq \frac{3}{(1-p)\cdot(1-c)\cdot\varepsilon^2} ln(2/\delta)$, *i.e.* $2e^{-\frac{\mathbf{E}(X_t)\cdot\varepsilon^2}{3}} \leq \delta$. Thus $N \cdot (1 - p) \cdot (1 - c)$ gives $(\varepsilon, \delta)$-approximation for the number of common nodes stayed in **COL** during time slot $t$. ∎

From Proposition 5 we know that in one time slot, there are $N \cdot (1 - p)(1 - c)$ data on common nodes that can be collected, *i.e.*, not all $N$ data items can be collected in one time slot. Lemma 6 shows how many time slots a common node $w$ stays in **UNCOL** state. After those time slots, $w$'s data can be collected.

*Lemma 6:* For any common node $w$, if it stays in **UNCOL** state, on average for $\frac{1}{1-c}$ time slots, $w$ can change its state from **UNCOL** to **COL**.

**Proof**. From Fig. 4 we know that $w$'s state is a Markov chain. For simplicity, in Fig. 4, 0 denotes **UNCOL** and 1 denotes **COL**. Let $r_{0,1}^t$ denote the probability that, starting at state 0, the first transition to state 1 occurs at time slot $t$; that is, $r_{0,1}^t = c^{t-1} \cdot (1 - c)$, *i.e.*, in the previous $t - 1$ time slots, $w$ stays in state 0, and at time slot $t$, $w$ stays in state 1. We denote by $h_{0,1} = \sum_{t=1}^{\infty} t \cdot r_{0,1}^t$ the expected time to first reach state 1 from state 0. Then $h_{0,1} = \sum_{t=1}^{\infty} t \cdot c^{t-1} \cdot (1 - c)$=$\frac{1}{1-c}$. ∎

Based on Lemma 6, the length of a round can be set as $\frac{1}{1-c}$ and at the beginning of each round, if a node generates new data and its state is **COL**, then its state recovers from **COL** to **UNCOL**. The probability of the case that after $\frac{1}{1-c}$ time slots, any common node becomes **COL**, is given by the Theorem 7.

*Theorem 7:* For any common node $w$, the probability that its data can be collected by collectors during $\frac{1}{1-c}$ time slots approaches to 1.

**Proof**. In one time slot $t$, for any common node $w$, the probability that its data is not collected by collectors is $\Pr(X_{w,t} = 0) = c$, and the probability that its data is not collected by collectors during $\frac{1}{1-c}$ time slots is $c^{\frac{1}{1-c}}$. Thus, the probability that its data can be collected by collectors in at least one time slot during $\frac{1}{1-c}$ time slots is $1 - c^{\frac{1}{1-c}}$. According to formula $1 - (1 - x)^n \approx n \cdot x (x \ll 1; n > 1)$, $1 - c^{\frac{1}{1-c}}$= $1 - (1 - b)^{\frac{N\cdot p}{1-(1-b)^{N\cdot p}}} \approx \frac{b\cdot N\cdot p}{1-(1-b)^{N\cdot p}} \approx \frac{b\cdot N\cdot p}{b\cdot N\cdot p} = 1$. ∎

*Corollary 8:* For any node $w$, the probability that $w$ receives the **VP** message from the mobile sink during $\frac{1}{1-c}$ time slots is $\frac{h_{max}^2}{N\cdot p}$, where $h_{max}$ is the number of the flooding hops from the sink.

**Proof**. In one time slot $t$, for any node $w$, the probability that it receives **VP** from the sink is $f = \frac{\pi\cdot r^2\cdot h_{max}^2}{Z^2}$. The probability that $w$ receives **VP** during $\frac{1}{1-c}$ time slots is $1 - (1 - f)^{\frac{1}{1-c}}$. According to $1 - (1 - x)^n \approx n \cdot x (x \ll 1; n > 1)$, $1 - (1 - f)^{\frac{1}{1-c}} \approx \frac{f}{1-c} \approx \frac{f}{b\cdot N\cdot p}$=$\frac{h_{max}^2}{N\cdot p}$. ∎

**Optimization Strategy:** According to Corollary 8, if $h_{max}$=$\sqrt{N \cdot p}$, then for any node $w$, the probability that it receives **VP** from the sink is close to 1. Theorem 7 tells us data on the common nodes can be transmitted to collectors in $\frac{1}{1-c}$ time slots with a high probability. However, there still exists a case with small probability $c^{\frac{1}{1-c}}$ that data on a common node $w$ is not collected by collectors during $\frac{1}{1-c}$ time slots. To avoid such a case and improve data delivery ratio, if a common node $w$ stays in **UNCOL** state in all the previous $\frac{1}{1-c} - 1$ time slots of a round, then $w$ broadcasts it data to all of its neighbors in the last time slot of a round. If $w$'s neighbor is a common node, then the common node stores the received data from $w$ in its local buffer as a replica. In this way, if $w$'s neighbors' data can be collected, then $w$'s data can also be collected.

### B. From Collector Nodes To Sink

This section shows that all gathered data on collectors can also be forwarded to the mobile sink with a high probability.

*Theorem 9:* Suppose there are $k$ hops from a collector $u$ to the sink $S$ along a path **P** from $u$ to $S$, where **P**=$\{u, u_1, u_2, \cdots, u_{k-1}, S\}$. $u_1, u_2, \cdots, u_{k-1}$ are intermediate relay nodes. The probability of a successful transmission along **P** is close to 1 if $N$ is large enough .

**Proof**. First, we present the probability of 1-hop successful transmission from any collector $u$ to its neighbors in the sector (as shown in Fig. 3, the sector $\widehat{CuD}$) with angle $2 \cdot \theta_u$ (see Section III-B, $\theta_u = \arcsin(\frac{\hat{r}_u}{|us|})$). Since each node randomly and independently moves, the probability for any node to go into the sector with the angle $2 \cdot \theta_u$ is $d_u = \frac{r^2 \cdot \theta_u}{Z^2}$. Thus, the probability of the case that there is at least one node in the sector is $e_u = 1-(1-d_u)^N$. If there is a node in the sector, then the 1-hop angle transmission from $u$ to its neighbors in the sector will be successful. Therefore, the probability of that 1-hop successful transmission from $u$ to its neighbors in the sector is $e_u = 1-(1-d_u)^N$. On average, the number of the nodes in the sector is $N \cdot d_u$ (the proof is similar to that of Proposition 1). For any node $v \in$**P**, $v$ has a sector with a transmission angle. On average, the number of the nodes in $v$'s sector is $N \cdot d_v$. Thus, transmission from $u$ to $S$ is a multi-path transmission and there are at least $\min_{v \in \mathbf{P}} N \cdot d_v$ paths. Among these paths, the probability of the case that there is at least one path which is a successful transmission path is $1-(1-\prod_{v \in \mathbf{P}} e_v)^{\min_{v \in \mathbf{P}}\{N \cdot d_v\}}$. This probability is close to 1 if $N$ is large enough. ■

## V. Simulation Evaluation

SMITE is implemented on NS2 with C++. Mobile nodes and a sink are deployed in an area of $1200m \times 1200m$. Each mobile node as well as the sink randomly chooses their movement direction from (0, $2\pi$) and moves with a velocity randomly chosen from (0, $maxv$ $m/s$), where $maxv$ is the node's maximum velocity. Each node has a transmission range of $100m$. The bloom filter with 5 random hash functions on each sensor is 320 bits. The tolerant false positive probability $\eta$ is $5\%$. A time slot duration is $1s$. A round has two time slots according to Lemma 6. The sink partially floods its location and velocity every time slot. In each round, all data on $N$ nodes should be collected. We compare SMITE with Sidewinder [14], DFT-MSN [18] and SCAR [19] on the aspects of data delivery ratio, messages overhead and time delay.

For data delivery ratio, the node's maximum velocity and the number of nodes are varied. Fig. 5 shows the delivery ratio against the maximum velocity of nodes $maxv$. 200 sensor nodes ($N$=200) randomly move in the area. The number of the hops of the sink's partial flooding is 12 ($h_{max}$=12). 20% of the nodes are chosen as collectors ($p$=20%). As the maximum speed increases, the delivery ratio of SMITE increases. SMITE outperforms DFT-MSN, Sidewinder and SCAR when the nodes' maximum speed exceed $5m/s$. The data delivery ratio of SMITE is on average 28.3%, 85.3% and 37.0% higher

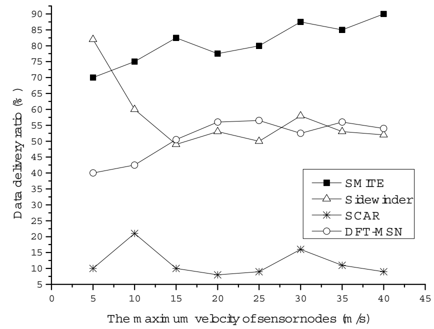


Fig. 5. Delivery ratio vs the maximum velocity of sensor nodes.

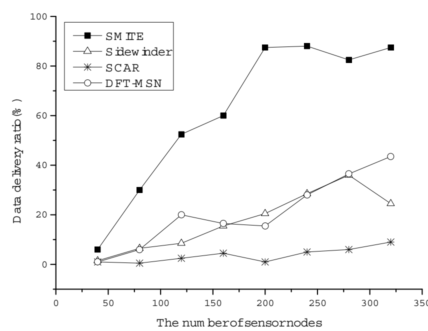


Fig. 6. Delivery ratio vs the number of sensor nodes.

than that of Sidewinder, SCAR and DFT-MSN respectively. In a high-speed scenario, the chance of common nodes meeting with collectors increases, so that it is more likely that the transmissions from common nodes to collectors are successful. The estimated movement territory for the sink in the high-speed scenario also enhances because the sink's estimated territory relies on the sink's speed. Large territory results in a large transmission angle during angle transmission. This leads to more multi-path transmission from collectors to the sink and improves the data delivery ratio. Fig. 6 shows the delivery ratio against the number of sensor nodes $N$. The maximum velocity of nodes is $20m/s$ ($maxv$=$20m/s$). $N$ varies from 40 to 320. As the number of nodes increases, the delivery ratio of SMITE increases. The data delivery ratio of SMITE is on average 72.9%, 93.1% and 69.3% higher than that of Sidewinder, SCAR and DFT-MSN respectively. In a high density scenario, the chance of common nodes meeting with collectors also increases. High node density results in a large number of nodes in the transmission angle during angle transmission. This results in more multi-path transmissions from collectors to the sink and increases the data delivery ratio.

For message overhead, the number of nodes varies from 40 to 320. Fig. 7 shows the total number of messages against the number of nodes $N$. 20% of the nodes are chosen as collectors ($p$=20%). The maximum speed is $20m/s$. As the number of nodes increases, the total numbers of messages

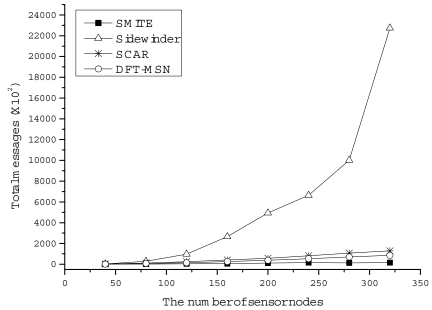


Fig. 7. Total messages vs the number of sensor nodes.

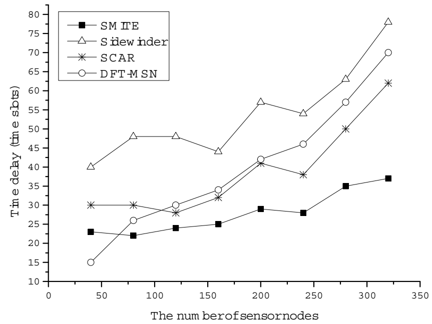


Fig. 8. Time delay vs number of sensor nodes.

from SMITE, Sidewinder, SCAR and DFT-MSN increase. The total number of messages from SMITE is on average 93.4%, 81.1% and 69.2% fewer than that of Sidewinder, SCAR and DFT-MSN respectively. Collected data on collectors can be aggregated together into a bloom filter. With this design, SMITE can reduce message overhead. In SMITE, only a minority of collectors periodically broadcast their context information rather than all nodes. Common nodes do not need to periodically broadcast their context information. To further reduce communication overhead during angle transmission, a node forwards its received packet only once if it stays in an overlapped forwarding angle from two forwarding nodes. Besides this, SMITE only needs mobile sink's partial flooding. This scheme can greatly reduce message overhead.

For time delay, the number of nodes were varied from 40 to 320. Fig. 8 shows the average delay against the number of nodes. The delay of successfully delivered data is defined as the total time slots from the time when the data is generated until the data is collected. $p$ is 20%. The maximum speed is $20m/s$. The delay of SMITE is on average 48.0%, 26.5% and 20.5% shorter than that of Sidewinder, SCAR and DFT-MSN. In SMITE, only aggregated data on collectors should be collected. This design can reduce the amount of collected data.

## VI. Conclusion

The paper proposes SMITE, a novel data collection protocol for MWSNs. The theoretical results show that all data from the common nodes can be transmitted to the collectors with a high probability and gathered data on the collectors can also be forwarded to the mobile sink with a high probability. The collectors use bloom filters to compress received data. The simulation results on NS2 validate the theoretical results and show that SMITE outperforms the state-of-the-art schemes, DFT-MSN, SCAR and Sidewinder, on the aspects of packet delivery ratio, transmission overhead, and time delay.

**Acknowledgements**: This work was supported by NSF under grant No. CAREER-CNS-844144 and CCF-0545667. It is partly supported by NSFC grant No.61033015, 60831160525 and 60803015.

## References

[1] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, and J. Huang. SensorFlock: An Airborne Wireless Sensor Network of Micro-Air Vehicles. In ACM SENSYS, 2007.
[2] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G-S. Ahn, and A. T. Campbell. The BikeNet Mobile Sensing System for Cyclist Experience Mapping. In ACM SENSYS, 2007.
[3] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan. The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. In ACM MOBISYS, 2008.
[4] T. Liu, C. M. Sadler, P. Zhang, and M. R. Martonosi. Implementing Software on Resource-Constrained Mobile Sensors: Experiences with Impala and ZebraNet. In ACM MOBISYS, 2004.
[5] C. Luo, F. Wu, J. Sun, and C. W. Chen. Compressive Data Gathering for Large-Scale Wireless Sensor Networks. In ACM MOBICOM, 2009.
[6] Y. Li, L. Guo, and S. Prasad. An Energy-Efficient Distributed Algorithm for Minimum-Latency Aggregation Scheduling in Wireless Sensor Networks. In IEEE ICDCS, 2010.
[7] A. Woo, T. Tong, and D. Culler. Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks. In ACM SENSYS, 2003.
[8] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In ACM MOBICOM, 2000.
[9] P.-J. Wan, C.-W. Yi, F.F. Yao, X. Jia. Asymptotic Critical Transmission Radius for Greedy Forward Routing in Wireless Ad Hoc Networks. In ACM MOBIHOC,2006.
[10] T. He, J. A. Stankovic, C. Lu, and T. Abdelzaher. SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks. In IEEE ICDCS, 2003.
[11] S. Funke and N. Milosavljevic. Guaranteed-delivery Geographic Routing under Uncertain Node Locations. In IEEE INFOCOM, 2007.
[12] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Mobile Computing, chapter 5, edited by T. Imielinski and H. Korth, Kluwer Academic Publishers, 1996.
[13] Y. B. Ko and N. H. Vaidya. Location-Aided Routing (LAR) in Mobile Ad Hoc Networks. In ACM MOBICOM, 1998.
[14] M. Keally, G. Zhou, and G. Xing. Sidewinder: A Predictive Data Forwarding Protocol for Mobile Wireless Sensor Networks. In IEEE SECON, 2009.
[15] X. Xu, J. Luo, and Q. Zhang. Delay Tolerant Event Collection in Sensor Networks with Mobile Sink. In IEEE INFOCOM, 2010.
[16] M. Ma and Y. Yang. SenCar: An Energy-Efficient Data Gathering Mechanism for Large-Scale Multihop Sensor Networks. IEEE TPDS, 18(10): 1476-1488, 2007.
[17] X. Zhang, H. Wang, A. Khokhar. An Energy-Efficient Data Collection Protocol for Mobile Sensor Networks. In IEEE VTC, 2006.
[18] Y. Wang and H. Wu. DFT-MSN: The Delay/Fault-Tolerant Mobile Sensor Network for Pervasive Information Gathering. In IEEE INFOCOM, 2006.
[19] B. Psztor, M. Musolesi, C. Mascolo. Opportunistic Mobile Sensor Data Collection with SCAR. In IEEE MASS, 2007.
[20] B. L. Bowerman and R. T. O'Connell. *Forecasting and Time Series: An Applied Approach*(3rd ed). Belmont, California: Duxbury Press, 1993.
[21] D. Guo, J. Wu, H. Chen, Y. Yuan, and X. Luo. The Dynamic Bloom Filters. IEEE TKDE, 22(1): 120-133, 2010.
[22] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. New York, NY: Cambridge University Press, 2005.