Snapshot and Continuous Data Collection in Probabilistic Wireless Sensor Networks

Shouling Ji, *Student Member*, *IEEE*, Raheem Beyah, *Senior Member*, *IEEE*, and Zhipeng Cai, *Member*, *IEEE*

Abstract—*Data collection* is a common operation of Wireless Sensor Networks (WSNs), of which the performance can be measured by its achievable *network capacity*. Most existing works studying the network capacity issue are based on the unpractical model called *deterministic network model*. In this paper, a more reasonable model, *probabilistic network model*, is considered. For snapshot data collection, we propose a novel *Cell-based Path Scheduling* (CPS) algorithm that achieves capacity of $\Omega(1/5\omega \ln n \cdot W)$ in the sense of the worst case and order-optimal capacity in the sense of expectation, where n is the number of sensor nodes, ω is a constant, and W is the data transmitting rate. For continuous data collection, we propose a *Zone-based Pipeline Scheduling* (ZPS) algorithm. ZPS significantly speeds up the continuous data collection process by forming a data transmission pipeline, and achieves a capacity gain of $N\sqrt{n}/\sqrt{\log n}\ln n$ or $n/\log n \ln n$ times better than the optimal capacity of the snapshot data collection scenario in order in the sense of the worst case, where N is the number of snapshots in a continuous data collection task. The simulation results also validate that the proposed algorithms significantly improve network capacity compared with the existing works.

Index Terms—Probabilistic wireless sensor networks, data collection, probabilistic network model, lossy links

1 Introduction

Wireless Sensor Networks (WSNs) are mainly used for gathering data from the physical world [1], [2], [3], [4]. Generally, data gathering can be categorized as *data aggregation* [5], [6], [7], [8], which obtains aggregated values, for example, the maximum, minimum, or average values of all the data, and *data collection* [10], [37], [43], [47], which gathers all the data from the network without any data aggregation. For data collection, the union of all the values from all the nodes at a particular time instant is called a *snapshot* [10]. The problem of collecting one snapshot is called *snapshot data collection* (SDC). On the other hand, the problem of collecting multiple continuous snapshots is called *Continuous Data Collection* (CDC). To evaluate network performance, *network capacity*, which can reflect the achievable data transmission/collection rate, is usually used [29], [21], [10], [28], [43], [11], [19], [13]. Particularly, for unicast, multicast, and broadcast, we use *unicast capacity*, *multicast capacity*, and *broadcast capacity* to denote the network capacity, respectively. For data collection, we use the data receiving rate at the sink, referred to as *data collection capacity*, to measure its achievable network capacity, i.e., data collection capacity reflects how fast data have been collected at the sink.[1]

After the seminal work [36], many works emerged to study the network capacity issue under the *Protocol Interference Model* (PrIM) [10], [43][2] or the *Physical Interference Model* (PhIM) [30][3] for a variety of network scenarios, for example, multicast capacity [11], unicast capacity [32], broadcast capacity [35], and SDC capacity [10], [37]. All of the above-mentioned works are based on the *deterministic network model*, where any pair of nodes in a network is either *connected* or *disconnected*. If two nodes are connected, i.e., there is a deterministic link between them, then a successful data transmission can be guaranteed as long as there is no collision. For the wireless networks considered under the deterministic network model, we call them *deterministic wireless networks*. However, in real applications, this deterministic network model assumption is not practical due to the "*transitional region phenomenon*" [55], [56] (beyond the always connected region, there is a transitional region where wireless links are opportunistically connected). With the transitional region phenomenon, a large number of network links (more than 90 percent [55]) become unreliable links, named *lossy links* [55]. Even without collisions, data transmission over a lossy link is successfully conducted with a certain probability, rather than being completely

- *S. Ji is with the Department of Computer Science, Georgia State University, 34 Peachtree Street, Suite 1428, Atlanta, GA 30303, and the School of Electrical and Computer Engineering, Georgia Institute of Technology, 266 Ferst Drive Atlanta, GA 30332-0765. E-mail: sji@gatech.edu.*
- *R. Beyah is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, 266 Ferst Drive, Klaus Advanced Computing Building, Rm. 2308, Atlanta, GA 30332-0765. E-mail: rbeyah@ece.gatech.edu.*
- *Z. Cai is with the Department of Computer Science, Georgia State University, 34 Peachtree Street, Suite 1413, Atlanta, GA 30303. E-mail: zcai@cs.gsu.edu.*

Manuscript received 5 Jan. 2012; revised 16 Sept. 2012; accepted 18 Jan. 2013; published online 5 Mar. 2013.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2012-01-0007.
Digital Object Identifier no. 10.1109/TMC.2013.30.

1. Without confusion, we use data collection capacity and network capacity interchangeably in the following of this paper.
2. Under the PrIM, only local wireless interference is considered, and a data transmission can be successfully conducted only if the receiver is within the transmission range of the transmitter and meanwhile out of the interference range of any other ongoing transmitter.
3. Under the PhIM, the aggregated wireless interference from the entire network is considered, and a data transmission can be successfully conducted only if the SINR at the receiver associated with the transmitter is no less than a predefined threshold value.

guaranteed. Therefore, a more practical network model for wireless networks is the *probabilistic network model* [55], in which data communication over a link is successful with a certain probability rather than always being successful or failed. For convenience, the wireless networks considered under the probabilistic network model are called *probabilistic wireless networks*.

Recently, many efforts have been spent on the data collection issue. In [9], [10], [43], [44], [45], [46], some tree-based data collection algorithms are proposed under the deterministic network model. The authors of [9] designed a family of path scheduling algorithms for SDC. Later on, the authors in [10], [43], [44] improved the path scheduling algorithms in [9] and implemented order-optimal data collection methods with higher achievable capacity. Unlike [9], [10], [43], [44], the authors in [45], [46] studied the distributed data collection issue and designed an order-optimal distributed data collection algorithm. In [14], [15], [16], [17], [27], some data collection schemes are designed based on the cell-partition idea. Furthermore, by exploiting the geometrical properties of network distribution, the achievable data collection capacity are also analyzed in [14], [15], [16], [17], [27]. In [37], taking the advantage of the compressive data gathering (CDG) technique, the authors in [37] designed a tree-based data collection algorithm. By analysis, they showed that the designed algorithm is order-optimal under both the PrIM and the PhIM. Unfortunately, for the data collection capacity issue, all the above-mentioned existing works are based on the ideal deterministic network model rather than the more realistic probabilistic network model. Actually, lossy links may degrade the achievable network capacity of data collection because retransmissions may happen when transmit data, and thus more interference and congestion may be induced, followed by lower data transmission concurrency and efficiency. On the other hand, how these lossy links and retransmissions affect the snapshot and continuous data collection capacities is still an open problem. This motivates us to investigate the achievable network capacity of WSNs under the probabilistic network model.

Specifically, in this paper, we study the achievable SDC and CDC capacity for probabilistic WSNs. Inspired by existing network partition methods [41], [42], we first investigate how to partition a probabilistic WSN into *cells* and *zones* to improve the concurrency of the data collection process. Subsequently, we propose two data collection schemes, the *Cell-based Path Scheduling* (CPS) algorithm and the *Zone-based Pipeline Scheduling* (ZPS) algorithm for SDC and CDC, respectively. This work is dedicated to the data collection capacity issue for probabilistic WSNs and the main contributions are as follows:

- For a probabilistic WSN deployed in a square area, we first partition the network into small *cells*. Then, we abstract each cell to a *super node* in the *data collection tree* built for data collection. Based on the data collection tree, we design a novel *CPS* algorithm for SDC. Theoretical analysis shows that the achievable network capacity of CPS is $\Omega(\frac{1}{5\omega \ln n} \cdot W)$ in the sense of the worst case, and $\Omega(\frac{p_o}{2\omega} \cdot W)$ in the sense of expectation, where p_o is the *promising transmission threshold probability* defined in Section 2, ω is a constant defined in Section 3, and W is the data transmitting rate over a wireless channel, i.e., the channel bandwidth. Since the upper bound of the SDC capacity is shown to be W [10], [43], CPS successfully achieves the order-optimal network capacity in the sense of expectation.
- For the CDC problem in a probabilistic WSN, an intuitive idea is to employ a SDC method in a pipeline manner. However, this idea can only improve network capacity within a constant factor even in a deterministic WSN [43]. Therefore, by combining the CDG technique (a data gathering technique by exploiting the compressive sampling theory) [37] (see details in Section 5.2) and the pipeline technique, we propose a novel *ZPS* algorithm for CDC in probabilistic WSNs. Taking the benefits brought by CDG and pipeline, ZPS improves the achievable network capacity significantly. For collecting N continuous snapshots, we theoretically prove that the asymptotic achievable network capacity of ZPS is 1)
$$\Omega\left(\frac{N\sqrt{n}}{10\omega M\sqrt{\log n}\ln n}\cdot W\right)$$
if $N = O(\sqrt{n/\log n})$ or $\Omega(\frac{n}{20\omega^2 M \log n \ln n} \cdot W)$ if $N = \Omega(\sqrt{n/\log n})$ in the sense of the worst case; and 2)
$$\Omega\left(\frac{p_o N\sqrt{n/\log n}}{4\omega M}\cdot W\right)$$
if $N = O(\sqrt{n/\log n})$ or $\Omega(\frac{p_o n}{8\omega^2 M \log n} \cdot W)$ if $N = \Omega(\sqrt{n/\log n})$ in the sense of expectation, where n is the number of nodes in a WSN and M is a parameter used in CDG and usually $M \ll n$ in large-scale WSNs. Considering that the upper bound capacity is also W for CDC, this implies that the achievable network capacity of ZPS is
$$\frac{N\sqrt{n}}{\sqrt{\log n}\ln n}$$
or $\frac{n}{\log n \ln n}$ times better than the optimal capacity of the snapshot data collection scenario in order in the sense of the worst case, and $\sqrt{n/\log n}$ or $n/\log n$ times better than the optimal capacity of the snapshot data collection scenario in order in the sense of expectation, which are significant improvements.
- The simulation results also indicate that the proposed algorithms significantly improve the network capacity compared with the existing works for probabilistic and deterministic WSNs.

The rest of this paper is organized as follows: Sections 2 and 3 introduce the probabilistic network model and the network partition strategy, respectively. The CPS algorithm for SDC is proposed and analyzed in Section 4. Section 5 presents a novel ZPS scheme for CDC and its achievable asymptotic network capacity is shown. We conclude this paper in Section 6. Related works and the simulation results validating the proposed algorithms are presented in a

TABLE 1
Notations in This Paper

Parameter	Value
n	the number of sensor nodes
s_i	a sensor node
A	the size of the network deploying area
c, c_i, N_0	constants
η_i, μ_i^z	constants
B	the size of a data packet
W	the bandwidth of the wireless channel
t_o	a time slot
P	the working power of sensor nodes
$\Lambda(s_i, s_j)$	the SINR value at s_j associated with s_i
α	the path-loss exponent
$\Pr(s_i, s_j)$	the data transmission success probability from s_i to s_j
p_o	the promising transmission threshold probability
$\hbar$	the upper bound of retransmissions for a data packet over a lossy link
l	the length of a cell
m	the number of cells in a row/column
$\kappa_{i,j}$	a cell
$\chi_{i,j}$	the number of sensor nodes within cell $\kappa_{i,j}$
$\mathbb{S}_{i,j}$	the Compatible Transmission Cell Set (CTCS) containing cell $\kappa_{i,j}$
ω, r	constants, see Theorem 1
$o_{i,j}$	a compatible zone
$R = \omega l$	the length of a compatible zone
$s_{i,j}^u$	the super node corresponding to cell $\kappa_{i,j}$
$\mathbb{T}$	the data collection tree
S_i	a segment
L_j^i	the j-th level in segment S_i
t_p	the maximum super time slots consumed by a segment for transmitting one snapshot

supplemental file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2013.30.

2 Network Model

In this section, we describe the network model and assumptions. For the frequently used notations, we list them in Table 1 for convenience of referencing.

In this paper, we consider a WSN consisting of n nodes, denoted by $s_1, s_2, \ldots, s_n$, respectively, and one sink (base station) deployed in a square plane with area $A = cn$ (i.e., the node density of the network is $1/c$), where c is a constant. Furthermore, we assume the distribution of all the nodes is independent and identically distributed (i.i.d.) and without loss of generality, the sink is located at the top-right corner of the square.[4] At each time interval, every node generates a data packet with size B bits, and transmits its data to the sink via a multihop way over a common wireless channel with bandwidth W bits/second, i.e., the data transmitting rate of the common channel is W. We further assume the time is slotted into time slots with each of length $t_o = B/W$ seconds.

During the data collection process, all the nodes in the network transmit data with an identical power P. Therefore, when node s_i transmits a packet to node s_j, the signal-to-interference-and-noise-ratio (SINR) associated with s_i at s_j is defined as

$$\Lambda(s_i, s_j) = \frac{P \cdot \|s_i - s_j\|^{-\alpha}}{N_0 + \sum_{s_k \in \mathcal{S}, s_k \neq s_i} P \cdot \|s_k - s_j\|^{-\alpha}}, \tag{1}$$

where $\|s_i - s_j\|$ is the euclidean distance between s_i and s_j, α is the path-loss exponent and usually $\alpha \in (2, 4)$, $N_0 > 0$ is a constant representing the background noise, and $\mathcal{S}$ is the set of all the transmitters that transmit data simultaneously with s_i. Traditionally, in a *deterministic network model*, people assumed that if the SINR value at a node is greater than or equal to a threshold value, the packet can be received successfully. However, in real-application environments, due to the existence of plenty of *lossy links*, this deterministic network model is too ideal. To be more practical and realistic, instead of taking the deterministic network model, we define a *probabilistic network model*, where each link is associated with a *success probability* that indicates the probability that a successful data transmission is conducted over this link. Based on the empirical literatures [56], we define the success probability associated with s_i and s_j as

$$\Pr(s_i, s_j) = \left(1 - \eta_1 \cdot e^{-\eta_2 \cdot \Lambda(s_i, s_j)}\right)^{\eta_3}, \tag{2}$$

where η_1, η_2, and $\eta_3 > 1$ are positive constants. Clearly, when s_i transmits a data packet to s_j, until a successful transmission (i.e., s_j successfully received the whole data packet), the number of transmissions satisfies the *geometric distribution* with parameter $\Pr(s_i, s_j)$. Therefore, the expected transmission times from s_i to s_j is $1/\Pr(s_i, s_j)$, i.e., this transmission will cost $1/\Pr(s_i, s_j)$ time slots on average.

Actually, we do not want the success probability to be too low, which implies too many transmission times, too much energy consumption and induced interference, as well as low transmission concurrency. Therefore, we introduce a *promising transmission threshold probability* p_o here. For any promising transmission, we require its success probability is no less than the promising transmission threshold probability p_o, i.e., for any node pair s_i and s_j, the transmission between s_i and s_j can be conducted only if $\Pr(s_i, s_j) \geq p_o$. Now, for any qualified communication pair to transmit one data packet, the expected transmission time is no more than t_o/p_o. For convenience, *in the sense of expectation*, we define a *modified time slot* $t_m = t_o/p_o$. Furthermore, we have Lemma 1 as follows, which indicates the upper bound of consumed time slots by any qualified communication to successfully transmit a data packet.

Lemma 1. *In an interference-free communication environment, it is almost sure that the number of consumed time slots of any qualified communication pair is upper bounded by* $\hbar = \arg\min_{1<z<1/(1-p_o)} 2\mu_1^z \ln n + \mu_2^z = O(\ln n)$,[5] *where* $\mu_1^z = -\frac{1}{\ln z(1-p_o)}$ *and* $\mu_2^z = -\log_{z(1-p_o)} \frac{p_o}{(1-p_o)(z-1)}$ *are some adjustable constant values depending on z.*

Proof. Please refer to the online supplemental material. □

4. Note that it is same with the situation when the sink is located at somewhere of the network, and we divide the network into four parts by a vertical line and a horizontal line, and consider each part individually.

5. Here, n is a notation that represents a large number. We exploit n to represent the upper bound of consumed time slots is mainly for the convenience of following derivations.

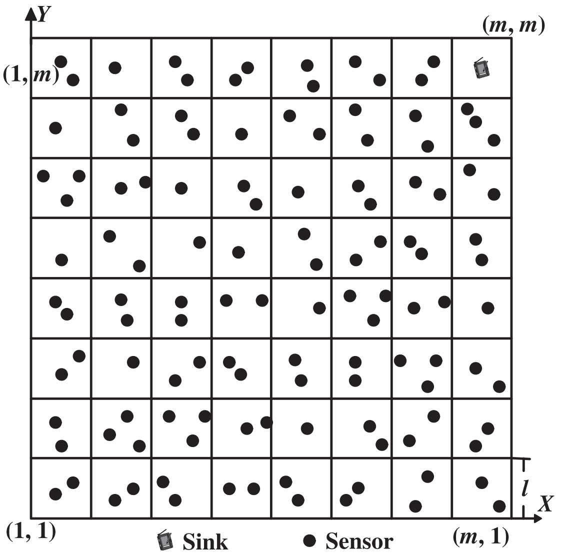


Fig. 1. Network partition.

Similarly, *in the sense of the worst case*, we define another *modified time slot* $t_w = \hbar \cdot t_o$ according to Lemma 1. In this paper, we analyze the achievable snapshot and continuous data collection capacities in the sense of expectation and the worst case, respectively.

We further formally define the achievable *data collection capacity* as the ratio between the amount of data successfully collected by the sink and the time Γ used to collect these data. For instance, in our probabilistic WSN model, to collect N continuous snapshots, the achievable data collection capacity is defined as NnB/Γ, which is actually the data receiving rate at the sink. Particularly, when $N = 1$, nB/Γ is the SDC capacity.

3 Network Partition

In this section, we explain the network partition method, which is essential for our following data collection algorithm.

3.1 Cell-Based Network Partition

In the previous section, we assume the network is distributed over a square with area size $A = cn$. Now, we partition the network into small square *cells* with edge length $l = \sqrt{4c \log n}$ by a group of horizontal and vertical lines. The resulting network is shown in Fig. 1. For convenience, we use $m = \sqrt{cn}/\sqrt{4c\log n} = \sqrt{n/4\log n}$ to denote the number of cells in each column/row and further define $m' = m - 1$. For each cell shown in Fig. 1, we assign each cell a pair of integer coordinates $(i, j)(1 \leq i,\ j \leq m)$, and a cell with coordinates (i, j) is denoted by $\kappa_{i,j}$. Clearly, the sink is located at the cell $\kappa_{m,m}$. Based on the network partition method, and considering that the sink is located at the top-right corner cell, we decide the possible communication modes for each cell (actually, for the nodes in each cell)[6] are *upward transmission*, *rightward transmission*, and *up-rightward transmission*. Take cell $\kappa_{i,j}$ as an example, when $\kappa_{i,j}$ works on the upward (respectively, rightward, up-rightward) transmission mode, it transmits its data to cell $\kappa_{i,j+1}$ (respectively, $\kappa_{i+1,j}$, $\kappa_{i+1,j+1}$). For cell $\kappa_{i,j}(1 \leq i, j \leq m)$, let the random variable $\chi_{i,j}$ denote the number of nodes within it. Then, based on the above network partition, the following three lemmas can be derived.

6. Without confusion, we use cell and the nodes within this cell interchangeably.

Lemma 2. *The expected number of nodes* $E[\chi_{i,j}]$, *i.e., the average number of nodes, in* $\kappa_{i,j}(1 \leq i, j \leq m)$ *is* $4 \log n$.

Proof. Please refer to the online supplemental material. □

Lemma 3. *It is almost surely that no cell is empty, i.e., it is almost surely that* Pr *(there exists at least one cell with no nodes)* $\cong 0$ *for large* n.

Proof. Please refer to the online supplemental material. □

Lemma 4. *It is almost surely that no cell contains more than* $10 \log n$ *nodes.*

Proof. Please refer to the online supplemental material. □

From Lemma 2, we know that the expected number of nodes within a cell is $4 \log n$. Lemma 3 implies that for large WSN, i.e., large n, every cell will have some nodes within it. Furthermore, from Lemma 4, the probability that a cell contains more than $10 \log n$ is zero when $n \rightarrow \infty$. Hence, in the following discussion, we assume a cell contains $4 \log n$ nodes *in the sense of expectation* and $10 \log n$ nodes *in the sense of the worst case*.

3.2 Zone-Based Network Partition

After partitioning the network into cells, we want to find which cells can carry out transmissions concurrently. Further, for these cells that can conduct transmissions concurrently, we define them as a *Compatible Transmission Cell Set* (CTCS), denoted by $\mathbb{S}$. Formally, we define $\mathbb{S} = \{\kappa_{i1,j1}, \kappa_{i2,j2}, \ldots, \kappa_{ig,jg} |$ 1) $1 \leq ik, jk \leq m$ for $1 \leq k \leq g$; 2) $\kappa_{ik,jk}(1 \leq k \leq g)$ can conduct transmissions concurrently; 3) For $\kappa_{ik,jk}(1 \leq k \leq g)$, suppose $\kappa'_{ik,jk}$ is its destination, i.e., $\kappa_{ik,jk}$ transmits data to $\kappa'_{ik,jk}$, then when $\kappa_{ik,jk}(1 \leq k \leq g)$ conduct transmissions simultaneously, $\min_{1 \leq k \leq g} \Pr(\kappa_{ik,jk}, \kappa'_{ik,jk}) = \min_{1 \leq k \leq g}\{\min\{\Pr(s_u, s'_u) | s_u$ is a node in $\kappa_{ik,jk}$, and s'_u is a node/sink in $\kappa'_{ik,jk}\} \geq p_o\}$. Clearly, the CTCS is an *equivalence relation* defined on the cells (i.e., CTCS is reflexive, symmetric, and transitive). Hence, a CTCS can be viewed as an *equivalence class*.

To partition the cells of a WSN into equivalence classes, i.e., CTCSs, we assign each cell $\kappa_{i,j}(1 \leq i, j \leq m)$ a vector representation $\vec{\kappa_{i,j}} = ((i-1) \cdot l, (j-1) \cdot l) = \kappa_{i,j}$. We further introduce two vectors $\vec{X} = (R, 0)$ and $\vec{Y} = (0, R)$, where $R = \omega \cdot l$, $\omega \in \mathbb{Z}$. Then, for any cell $\kappa_{i,j}(1 \leq i,\ j \leq m)$, we define the equivalence class, i.e., the CTCS, containing $\kappa_{i,j}$ as the set $\mathbb{S}_{i,j} = \{\vec{\kappa_{i,j}} + a \cdot \vec{X} + b \cdot \vec{Y} \mid a,\ b \in \mathbb{Z}\}$, i.e., $\mathbb{S}_{i,j} = \{\kappa_{i+a\cdot\omega, j+b\cdot\omega} \mid a, b \in \mathbb{Z}, 1 \leq i + a \cdot \omega, j + b \cdot \omega \leq m\}$ (Here, we suppose $\mathbb{S}_{i,j}$ is a CTCS. Later, we will show how to choose R to make it actually a CTCS.). Taking the WSN shown in Fig. 1 as an example, if we set $\omega = 3$, i.e., $R = 3l$, then the network can be partitioned into 9 equivalence classes, i.e., CTCSs, $\mathbb{S}_{i,j}(1 \leq i, j \leq 3)$ as shown in Fig. 2. In Fig. 2, the CTCS containing $\kappa_{1,1}$ is $\mathbb{S}_{1,1} = \{\kappa_{1,1}, \kappa_{4,1}, \kappa_{7,1}, \kappa_{1,4}, \kappa_{4,4}, \kappa_{7,4}, \kappa_{1,7}, \kappa_{4,7}, \kappa_{7,7}\}$. Now, we start from the bottom-left corner of

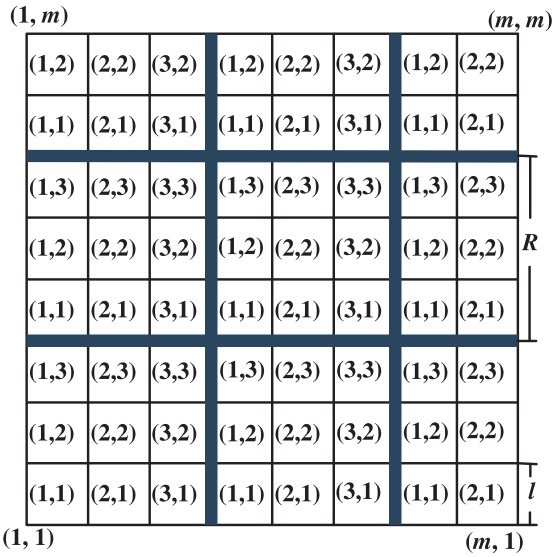


Fig. 2. Equivalence classes (CTCS) and zones.

the WSN and partition the network into square zones, named *compatible zones*, with edge length $R = \omega \cdot l$ as shown in Fig. 2 (where $\omega = 3$). Similar to denote a cell, for each compatible zone, we use $o_{i,j}(1 \leq i, j \leq \lceil m/\omega \rceil)$ to denote it, and the bottom-left zone with the smallest i and j, i.e., $o_{1,1}$. Clearly, $o_{i,j} = \{\kappa_{i',j'} \mid (i-1)\cdot\omega + 1 \leq i' \leq i \cdot \omega, (j-1)\cdot\omega + 1 \leq j' \leq j \cdot \omega\}$, i.e., within a compatible zone, none of the cells belong to the same equivalence class. Furthermore, all the cells with the same relative position in different compatible zones belong to the same equivalence, i.e., the same CTCS.

Now, to make any $\vec{X},\vec{Y}$-based cell set $\mathbb{S}_{i,j}$ actually a CTCS, we need to decide the value of R. For large WSNs, the value of R is determined by the following Theorem 1.

Theorem 1. *Let* $R = \omega \cdot l, \omega = \Theta(\frac{r+o(1)}{l}), r = 2\sqrt{2}l,$[7] $\vec{X} = (R, 0)$, $\vec{Y} = (0, R)$, *then the set* $\mathbb{S}_{i,j} = \{\vec{\kappa_{i,j}} + a \cdot \vec{X} + b \cdot \vec{Y} \mid a, b \in \mathbb{Z}\} = \{\kappa_{i+a\cdot\omega, j+b\cdot\omega} \mid a, b \in \mathbb{Z}, 1 \leq i + a\cdot\omega, j + b\cdot\omega \leq m\}$ *is a CTCS.*

Before proving Theorem 1, we prove Lemma 5 and Lemma 6 first. In the following proof, assume all the cells in a CTCS $\mathbb{S}_{i,j}$ conduct transmissions concurrently, and all other cells keep quiet or receive data from some cells in $\mathbb{S}_{i,j}$.

Lemma 5. *For each* $\mathbb{S}_{i,j}, \forall \kappa_{i,j} \in \mathbb{S}_{i,j}$, *suppose* $\kappa'_{i,j}$ *is the destination cell of* $\kappa_{i,j}$, *then* $\Lambda(\kappa_{i,j}, \kappa'_{i,j}) = \min\{\Lambda(s_u, s_v) \mid 1 \leq u, v \leq n, s_u \in \kappa_{i,j}, s_v \in \kappa'_{i,j}\} \geq \frac{P \cdot r^{-\alpha}}{N_0 + P \cdot \beta \cdot R^{-\alpha}}$, *where* $r = 2\sqrt{2}l$ *and* β *is a positive constant.*

Proof. Please refer to the online supplemental material. □

Lemma 6. $\mathbb{S}_{i,j}$ *is a CTCS when* $R \geq (c_9 \cdot r^{-\alpha} + c_{10})^{-1/\alpha}$, *where* $c_9 = \frac{\eta_2}{\beta \cdot \ln \eta_1 (1-\eta_3\sqrt{p_o})^{-1}}$ *and* $c_{10} = -\frac{N_0}{P \cdot \beta}$.

Proof. Please refer to the online supplemental material. □

Now, we are ready to prove Theorem 1.

Proof of Theorem 1. From Lemma 6, we know that when $R \geq (c_9 \cdot r^{-\alpha} + c_{10})^{-1/\alpha}$, $\mathbb{S}_{i,j}$ is a CTCS. Since large $|\mathbb{S}_{i,j}|$ implies more concurrent data transmissions, we prefer small R. Thus, let $R = (c_9 \cdot r^{-\alpha} + c_{10})^{-1/\alpha}$. Define $\omega = \lceil R/l \rceil$. For large n, i.e., large-scale WSNs, $R \sim \Theta(r + o(1))$, which implies $\omega = \Theta(\frac{r+o(1)}{l})$. Thus, the conclusion of Theorem 1 holds. □

From Theorem 1, we know that if we set $R = \omega \cdot l$, then, all the CTCSs can conduct data transmissions simultaneously in an interference-free manner. Based on the conclusion of Theorem 1, the following corollary can be obtained.

Corollary 1. *By* $\vec{X}$ *and* $\vec{Y}$, *the cells* $\kappa_{i,j}(1 \leq i, j \leq m)$ *can be partitioned into at most* ω^2 *CTCSs (equivalence classes).*

Proof. Please refer to the online supplemental material. □

4 Snapshot Data Collection

In this section, we study the achievable network capacity of SDC. First, we propose a novel CPS algorithm for SDC. Subsequently, we analyze the achievable network capacity of CPS. Finally, we make some further discussion about the extension from SDC to CDC.

4.1 Cell-Based Path Scheduling

Before giving the CPS algorithm, we construct a *data collection tree*, which serves as the routing structure, for the data collection process. For each cell $\kappa_{i,j}(1 \leq i, j \leq m)$, we abstract it to a *super node*, denoted by $s^u_{i,j}$.[8] Following the discussion in Section 3.1, a cell contains $4 \log n$ nodes in the sense of expectation and $10 \log n$ nodes in the sense of the worst case. Thus, we abstract the data packets of nodes within a cell as a *super data packet*, whose size is $4 \log n \cdot B$ bits in the sense of expectation and $10 \log n \cdot B$ bits in the sense of the worst case. Accordingly, to send out a super data packet, we define a *super time slot* t_s as $4 \log n \cdot t_m = 4 t_o \log n / p_o$ *in the sense of expectation* and $10 \log n \cdot t_w = 10 \hbar \log n \cdot t_o$ *in the sense of the worst case*. Afterwards, considering the communication modes defined in Section 3.1, we construct a data collection tree, denoted by $\mathbb{T}$, rooted at the sink to connect all the super nodes according to the following rules: 1) For super nodes $s^u_{i,j}(1 \leq i, j \leq m')$ (note that $m' = m - 1$), $s^u_{i,j}$ transmits its data to $s^u_{i+1,j+1}$, i.e., create a link from $s^u_{i,j}$ to $s^u_{i+1,j+1}$. 2) For super nodes $s^u_{m,j}(1 \leq j \leq m')$, $s^u_{m,j}$ transmits its data to $s^u_{m,j+1}$, i.e., create a link from $s^u_{m,j}$ to $s^u_{m,j+1}$. 3) For super nodes $s^u_{i,m}(1 \leq i \leq m')$, $s^u_{i,m}$ transmits its data to $s^u_{i+1,m}$, i.e., create a link from $s^u_{i,m}$ to $s^u_{i+1,m}$. After applying the above rules to all the super nodes except for $s^u_{m,m}$, the data collection tree is built. Taking the WSN shown in Fig. 1 as an example, the obtained data collection tree is shown in Fig. 3. For a data transmission route from a leaf super node to the root in $\mathbb{T}$, we call it a *path*. The path starting from $s^u_{i,1}(1 \leq i \leq m)$ is denoted by P_i and the path from $s^u_{1,j}(2 \leq j \leq m)$ is denoted by P'_j, as shown in Fig. 3.

7. r is a parameter in the derivation, which can be viewed as the maximum transmission range of a node.

8. Without confusion, we use cell and super node exchangeably.

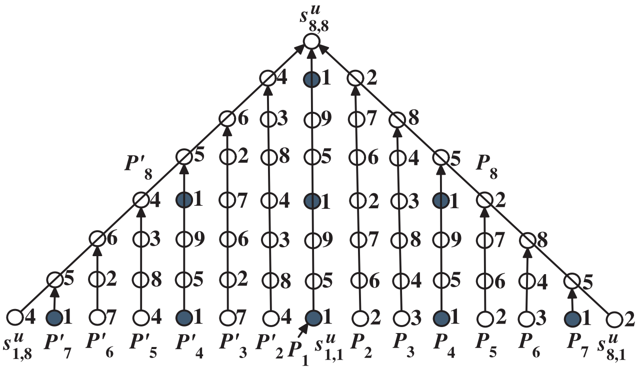


Fig. 3. Data collection tree.

According to Corollary 1, all the cells of a WSN can be partitioned into ω^2 CTCSs (equivalence classes). For each CTCS $\mathbb{S}_{i,j}(1 \le i, j \le \omega)$, we map it to an integer $(i-1)\cdot\omega + j$. In Fig. 3, the number next to each super node indicates the CTCS it belongs to. For convenience, we also use $\mathbb{S}_{(i-1)\cdot\omega+j}$ to represent the CTCS $\mathbb{S}_{i,j}(1 \le i, j \le \omega)$.

Based on the abstracted data collection tree $\mathbb{T}$, we propose a novel *CPS* algorithm, which has two phases. In Phase I of CPS, we schedule the ω^2 CTCSs one by one, until all the data packets of cells $\kappa_{i,j}(1 \le i, j \le m')$ have been collected to the cells on path P_m, path P'_m, or the sink. In Phase II of CPS, we schedule the cells of P_m and P'_m until all the data packets have been collected to the sink. We use the example shown in Fig. 3 to present the main idea of CPS as follows: The formal description of CPS is shown in Algorithm 1.[9]

Algorithm 1: The CPS Algorithm

input : a data collection tree, CTCSs
output: a data collection plan

```
while the sink has not collected all the data do
    for i = 1; i ≤ ω; i++ do
        for j = 1; j ≤ ω; j++ do
            if all the cells in CTCS S_{i,j} have no data for
            transmission then
                continue;
            Assign CTCS S_{i,j} a super time slot;
            During the assigned super time slot,
            schedule all the super nodes (cells) in S_{i,j}
            simultaneously:  for ∀κ_{u,v} ∈ S_{i,j}, schedule
            all the nodes with data for transmission
            in κ_{u,v} sequentially according to some
            order, e.g. the ID order, each with one
            modified time slot;
```

Phase I: Inner-Tree Scheduling. Since the cells within a CTCS can be scheduled to transmit data concurrently, schedule CTCSs $\mathbb{S}_1, \mathbb{S}_1, \ldots, \mathbb{S}_{\omega^2}$ orderly, each for a super time slot. Repeat Phase I until there is no packet remaining at the super node $s^u_{i,j}(1 \le i, j \le m')$, i.e., all the data packets at $s^u_{i,j}(1 \le i, j \le m')$ have been collected to the sink or $s^u_{i,j}(i = m$ or $j = m)$. For the specific nodes within a cell, schedule them sequentially according to any order at the available super time slots for this cell.[10] Taking the data collection tree $\mathbb{T}$ shown in Fig. 3 as an example, the cells in $\mathbb{T}$ can be partitioned into nine CTCSs. For the nine CTCSs $\mathbb{S}_1, \mathbb{S}_1, \ldots, \mathbb{S}_9$, we schedule them orderly each for one super time slot. At the end of Phase I, all the data packets of $s^u_{i,j}(1 \le i, j \le 7)$ have been collected to the sink, or the cells on path P_8 and P'_8.

Phase II: Scheduling of P_m and P'_m. For the super nodes $s^u_{i,j}(i = m$ or $j = m)$ which have data packets waiting for collection, partition them into λ CTCSs (Actually, $\lambda \le 2\omega - 1$ which is proven in Lemma 10.). Then, schedule these λ CTCSs sequentially each for one super time slot. Repeat Phase II until all the packets have been collected to the sink. Taking P_8 and P'_8 shown in Fig. 3 as an example, the cells on P_8 and P'_8 can be partitioned into five CTCSs. Then, we schedule these five CTCSs sequentially until all the data packets been collected to the sink.

From the description of CPS, we know it can collect all the data packets to the sink after Phase I and Phase II. In the following section, we will analyze the achievable network capacity of CPS.

4.2 Capacity Analysis of CPS

In this section, we investigate the achievable network capacity of CPS. The upper bound of SDC is W even under the deterministic network model [10], [43].[11] Therefore, the upper bound of SDC under the probabilistic network model is W too. Consequently, we focus on the lower bound of CPS in the following analysis.

For convenience, we introduce the concept of *scheduling round*. A scheduling round for Phase I (respectively, Phase II) of CPS is the time used to run Phase I (respectively, Phase II) once. For the data collection tree $\mathbb{T}$ shown in Fig. 3, a scheduling round is $9t_s$ (respectively, $5t_s$) in Phase I (respectively, Phase II), since there are nine (respectively, five) CTCSs need to schedule in each running of Phase I (respectively, Phase II). Now, we can obtain the number of super time slots used in Phase I of CPS as shown in Lemma 7.

Lemma 7. *For SDC, it takes CPS $\omega^2 m'$ super time slots to finish Phase I.*

Proof. According to the scheduling in Phase I, every CTCS is scheduled once in a scheduling round. This implies every super node in the network is scheduled once in every scheduling round. Therefore, for each super node $s^u_{i,j}(1 \le i, j \le m')$, it can receive one super data packet at most from its child and send out one super data packet at most to its parent during every scheduling round. Thus,

9. Note that, although the two phases are not shown explicitly in Algorithm 1, the data collection process can be viewed consisting of two phases as discussed.

10. Suppose the parent node of super node $s^u_{i,j}$ is $s^u_{i',j'}$, i.e., all the nodes in cell $\kappa_{i,j}$ will transmit their data to the nodes in cell $\kappa_{i',j'}$. Then, when a node s_u in cell $\kappa_{i,j}$ is scheduled to transmit data to some node in cell $\kappa_{i',j'}$, s_u will transmit its data to the node s_v in cell $\kappa_{i',j'}$, where s_v satisfies the condition that the success probability of the link from s_u to s_v is the highest among the links from s_u to all the nodes in cell $\kappa_{i',j'}$.

Now, assume the success probability of the link from s_u to s_v is 0.5. Then, when s_u transmits a data packet to s_v, s_v successfully receives this data packet with probability 0.5. If this data transmission fails, s_u will retransmit that data packet until the packet is successfully received by s_v. Evidently, the expected transmission times of that packet is 2 in this case.

In this paper, without specification, for any node s_u, it determines its next hop and transmits data in terms of the aforementioned manner.

11. This is because the sink node can receive at most one data packet during a time slot. Consequently, based on the definition of data collection capacity (which is defined as the average data receiving rate of the sink during a data collection process), W is a trivial upper bound of any data collection algorithm in both deterministic WSNs and probabilistic WSNs.

for each path of $P_i(1 \le i \le m')$ and $P'_j(2 \le j \le m')$, its length will decrease by one after each scheduling round (if we assume the node without any data for transmission will be deleted from the path). It follows that the data packets of $s^u_{i,j}(1 \le i, j \le m')$ will be collected to the sink or $s^u_{i,j}(i = m$ or $j = m)$ in m' scheduling round, i.e., $\omega^2 m'$ super time slots, since the length of the longest path of $P_i(1 \le i \le m')$ and $P'_j(2 \le j \le m')$ is m'. □

Now, we study the time slots used in Phase II of CPS. First, we derive the number super data packets remaining at each of the super nodes $s^u_{i,j}(i = m$ or $j = m)$ waiting for transmission at the beginning of Phase II. Subsequently, we obtain the upper bound of the number of super time slots used in Phase II, and followed by the lower bound of the achievable network capacity of CPS. In the following analysis, we use $\phi_{i,j}(1 \le i, j \le m)$ to denote the number of super data packets transmitted/forwarded by $s^u_{i,j}$ through the entire SDC process. Further, we use $\varphi_{i,j}(1 \le i, j \le m)$ to denote the number of super data packets at $s^u_{i,j}$ waiting for transmission at the beginning of Phase II. Clearly, $\varphi_{i,j} = 0(1 \le i, j \le m')$ after Phase I.

Lemma 8. *For* $1 \le i \le m'$, $\phi_{m,i} = \frac{i(i+1)}{2}$.

Proof. Based on the constructed data collection tree in the previous section, for $s^u_{m,i}(2 \le i \le m')$, it has two children $s^u_{m-1,i-1}$ and $s^u_{m,i-1}$. Hence, during the entire data collection process, the number of super data packets transmitted/forwarded by $s^u_{m,i}(2 \le i \le m')$ is the sum of the number of super data packets transmitted/forwarded by $s^u_{m-1,i-1}$ and $s^u_{m,i-1}$ plus 1 (1 means the super data packet of $s^u_{m,i}$ itself), i.e., $\phi_{m,i} = \phi_{m-1,i-1} + \phi_{m,i-1} + 1$. Considering $\phi_{m-1,i-1}$, it has only one child $\phi_{m-2,i-2}$. Thus, $\phi_{m-1,i-1} = \phi_{m-2,i-2} + 1$. In a sum, we have

$$\begin{cases} \phi_{m,1} = 1, \ \phi_{m-i+1,1} = 1 \\ \phi_{m-1,i-1} = \phi_{m-2,i-2} + 1 \\ \phi_{m,i} = \phi_{m-1,i-1} + \phi_{m,i-1} + 1. \end{cases} \quad (3)$$

Then, it is straightforward for us to obtain the generating functions of $\phi_{m-1,i-1}$, which is $\phi_{m-1,i-1} = i - 1$, and $\phi_{m,i}(1 \le i \le m')$, which is $\phi_{m,i} = \frac{i(i+1)}{2}$. □

From the proof of Lemma 8 and by symmetry, we have the following corollary.

Corollary 2. *For* $1 \le i \le m'$, $\phi_{i,m} = \frac{i(i+1)}{2}$.

Based on Lemma 8, we obtain the number of super data packets at $s^u_{m,i}$ waiting for transmission at the beginning of Phase II as shown in Lemma 9.

Lemma 9. *Let* $\theta = \lceil \frac{\sqrt{1+8m'}-1}{2} \rceil$, *then*

$$\varphi_{m,i} = \begin{cases} 0, \quad 1 \le i < \theta \\ \phi_{m,i} - m' = \dfrac{i(i+1)}{2} - m' \le i, \quad i = \theta \\ i, \quad \theta < i \le m'. \end{cases} \quad (4)$$

Proof. We prove this lemma by cases.

Case 1. $1 \le i < \theta$. From Lemma 8, $s^u_{m,i}$ transmits/forwards $\phi_{m,i} = \frac{i(i+1)}{2}$ super data packets to its parent through the entire SDC process. In Phase I, we schedule every CTCS for m' times by the proof of Lemma 7, which implies $s^u_{m,i}$ has been scheduled for m' times. It follows that $s^u_{m,i}$ can transmit/forward m' super data packets to its parent during its available super time slots in Phase I. Considering that $1 \le i < \theta$, we have $\phi_{m,i} = \frac{i(i+1)}{2} \le \frac{1}{2}(\theta^2 + \theta) = \frac{1}{2} \cdot 2m' = m'$. Thus, we conclude that $s^u_{m,i}(1 \le i < \theta)$ has already finished its data transmission task in Phase I, i.e., $\varphi_{m,i}(1 \le i < \theta) = 0$ at the beginning of Phase II.

Case 2. $i = \theta$. According to the proof of the previous case and the scheduling of Phase I, for super node $s^u_{m,\theta}$, its two children $s^u_{m,i-1}$ and $s^u_{m-1,i-1}$ have no data packet waiting for transmission at the beginning of Phase II. Furthermore, as explained in the previous case, $s^u_{m,\theta}$ has been scheduled for m' times in Phase I, which implies that $s^u_{m,i}$ transmitted m' super data packets to its parent. It follows that the number of data packets waiting at $s^u_{m,\theta}$ for transmission is $\varphi_{m,i} = \phi_{m,i} - m' = \frac{i(i+1)}{2} - m' \le i$ at the beginning of Phase II.

Case 3. $\theta < i \le m'$. For the child $s^u_{m-1,i-1}$ of $s^u_{m,i}$, it transmitted $i - 1$ super data packets to $s^u_{m,i}$ in Phase I by the proof of Lemma 8. For another child $s^u_{m,i-1}$ of $s^u_{m,i}$, it transmitted m' super data packets to $s^u_{m,i}$ in Phase I by the proof of Lemma 7. Furthermore, $s^u_{m,i}$ also transmitted m' super data packets to its parent $s^u_{m,i+1}$ by the proof of Lemma 7. This implies the number of super data packets waiting at $s^u_{m,i}(\theta < i \le m')$ for transmission at the beginning of Phase II is $\varphi_{m,i} = (i - 1) + 1 = i$. □

According to Corollary 2 and Lemma 9, it is straightforward to obtain the following corollary.

Corollary 3.

$$\varphi_{i,m} = \begin{cases} 0, \quad 1 \le i < \theta \\ \phi_{m,i} - m' = \dfrac{i(i+1)}{2} - m' \le i, \quad i = \theta \\ i, \quad \theta < i \le m'. \end{cases} \quad (5)$$

Lemma 10. *For super nodes* $s^u_{m,i}(\theta \le i \le m')$ *and* $s^u_{j,m}(\theta \le j \le m')$, *they can be partitioned into at most* $2\omega - 1$ *CTCSs, i.e.,* $\lambda \le 2\omega - 1$, *where* λ *is the one in Phase II of CPS.*

Proof. According to the vector-based CTCS partition method in Section 3.2, the super nodes $s^u_{m,i}(\theta \le i \le m)$ can be partitioned into at most ω CTCSs. Similarly, $s^u_{j,m}(\theta \le j \le m)$ can be partitioned into at most ω CTCSs too. Furthermore, $s^u_{m,m}$ lies in the same CTCS no matter how to partition these cells, which implies $s^u_{m,i}(\theta \le i \le m')$ and $s^u_{j,m}(\theta \le j \le m')$ can be partitioned into at most $2\omega - 1$ CTCSs. □

Lemma 11. *In Phase II of the CPS algorithm, it costs at most* $\frac{1}{2}(2\omega - 1)(m' + \theta)(m' - \theta + 1)$ *super time slots to transmit all the data packets to the sink.*

Proof. During each schedule round of Phase II, every super node of $s^u_{m,i}(\theta \le i \le m')$ and $s^u_{j,m}(\theta \le j \le m')$ is scheduled once to transmit a super data packet to its parent. Hence, the sink will receive two super data packets during every scheduling round. From Lemma 9 and Corollary 3, we know that the total number of super data packets waiting at $s^u_{m,i}(\theta \le i \le m')$ and $s^u_{j,m}(\theta \le j \le m')$

for transmission at the beginning of Phase II is at most $2\sum_{i=\theta}^{m'} = (m'+\theta)(m'-\theta+1)$. It turns out that the sink can collect all the super data packets at $s_{m,i}^u$ and $s_{j,m}^u$ within $\frac{1}{2}(2\omega-1)(m'+\theta)(m'-\theta+1)$. □

Now, we are ready to derive the achievable network capacity of CPS in the sense of the worst case and in the sense of expectation as shown in Theorem 2.

Theorem 2. *For the achievable data collection capacity of CPS for SDC, it is $\Omega(\frac{1}{5\omega \ln n}\cdot W)$ in sense of the worst case, which is a degradation of $O(\ln n)$ of the optimum capacity, and $\Omega(\frac{p_o}{2\omega}\cdot W)$ in the sense of expectation, which is order-optimal.*

Proof. From Lemma 7 and Lemma 11, the total number of super time slots used by CPS is at most

$$\omega^2 m' + \frac{1}{2}(2\omega-1)(m'+\theta)(m'-\theta+1) \tag{6}$$

$$\leq \omega^2 m + \frac{1}{2}\cdot 2\omega(m+\theta)(m-\theta) \tag{7}$$

$$\leq \omega^2 m + \omega m^2 \tag{8}$$

$$= \omega^2\sqrt{\frac{n}{2\log n}} + \frac{\omega n}{2\log n} \tag{9}$$

$$\leq O\left(\frac{\omega n}{2\log n}\right). \tag{10}$$

The total amount of data received by the sink is $n\cdot b$. Thus, in the sense of the worst case, the achievable network capacity of CPS is

$$\frac{n\cdot B}{O\left(\frac{\omega n}{2\log n}\right)\cdot t_s} = \frac{n\cdot B}{O\left(\frac{\omega n}{2\log n}\right)\cdot 10\log n\cdot t_w} \tag{11}$$

$$= \frac{n\cdot B}{O\left(\frac{\omega n}{2\log n}\right)\cdot 10\log n\cdot \hbar t_o} \tag{12}$$

$$= \Omega\left(\frac{1}{5\omega\hbar}\cdot W\right) \tag{13}$$

$$= \Omega\left(\frac{1}{5\omega\ln n}\cdot W\right). \tag{14}$$

Similarly, in the sense of expectation, the achievable network capacity of CPS is

$$\frac{n\cdot B}{O\left(\frac{\omega n}{2\log n}\right)\cdot t_s} = \frac{n\cdot B}{O\left(\frac{\omega n}{2\log n}\right)\cdot 4\log n\cdot \frac{t_o}{p_o}} \tag{15}$$

$$= \Omega\left(\frac{p_o}{2\omega}\cdot W\right). \tag{16}$$

Since the upper bound of SDC is W under deterministic/probabilistic network model, and p_o, ω are constants, the achievable network capacity of CPS in the sense of expectation is order-optimal. However, the data collection capacity of CPS has a degradation of $O(\ln n)$ in the sense of the worst case. □

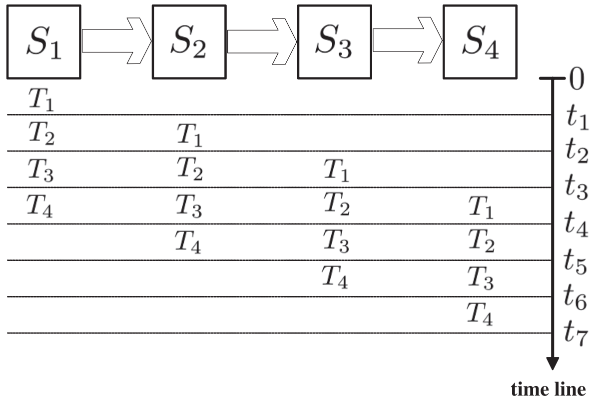


Fig. 4. A pipeline system.

When addressing the CDC problem, an intuitive idea is to combine the existing SDC methods with the *pipeline* technique. Nevertheless, such an idea cannot induce a significant improvement on the network capacity. Taking the CPS as an example, it has already achieved the order-optimal data collection capacity. By pipelining the CPS algorithm, data transmissions at the nodes far from the sink can definitely be accelerated. However, the fact that the sink can receive at most one packet during each time slot makes the data accumulated at the nodes near the sink. As a result, the network capacity still cannot be improved even with pipeline [43].

5 Continuous Data Collection

Intuitively, CDC has much more traffic load than SDC. Therefore, it is easier for the data to accumulate at the nodes near the sink, which makes the data transmission schedule very complicated and inefficient. Consequently, new elegant techniques are required to address this situation. On the other hand, the combination of a SDC method and the *pipeline* technique cannot improve network capacity effectively. Therefore, we propose a novel *ZPS* algorithm based on the technology used in *CDG* [37] in this section. Theoretical analysis shows that ZPS can improve data collection capacity significantly.

5.1 Pipelining

In computing, a *pipeline* is a set of data processing elements connected in series, so that the output of one element is the input of the next one and the elements of a pipeline are often executed in parallel. For instance, Fig. 4 shows a pipeline system consisting of four functional element S_1, S_2, S_3, and S_4 to address four tasks T_1, T_2, T_3, and T_4. To finish four tasks by this pipeline, we can input these tasks sequentially for processing. As shown in Fig. 4, we first input task T_1 (at time 0) to the functional element S_1 for processing. After T_1 is processed by S_1 (at time t_1), S_1 outputs the result to S_2 for processing, and meanwhile, T_2 will be input to S_1 (also at time t_1) for processing. Then, at some time slot, it can be achieved that multiple tasks are processed simultaneously at different elements of the pipeline system. For instance, all the four tasks are processed by the pipeline system during time slot (t_3, t_4) in Fig. 4. Evidently, by exploiting the pipeline technique, the efficiency of the entire functional system can be improved, and thus, the time consumption to process multiple tasks can be decreased. Consequently, to improve the efficiency and reduce the induced delay of the data collection process of CDC, we will partition the network into different functional elements to form an efficient data collection pipeline.

5.2 Compressive Data Gathering

CDG is first proposed in [37] for distributing the SDC load uniformly to all the nodes in the entire network. Suppose P is a data collection path consisting of d nodes $s_1, s_2, \ldots, s_d$, where s_1 is the leaf node, s_d is the sink (destination), and the data (packet) produced at $s_i (1 \leq i \leq d-1)$ is D_i. We use the data collection process on P to show the basic idea of CDG. In the traditional data collection way, for node $s_i (1 \leq i \leq d-1)$ on P, it transmits i data packets to its parent (1 is for itself and $i-1$ is for the packets it received), which is unbalanced, i.e., the nodes near the sink transmit more data than the ones far from the sink. By contrast, to collect $D_i (1 \leq i \leq d-1)$ to the sink, every node transmits M packets to its parent in the CDG way, i.e., s_1 multiplies its data with M random coefficients $\psi_{i1} (1 \leq i \leq M)$, respectively, and sends the M new data (packets) $\psi_{i1} D_1 (1 \leq i \leq M)$ to its parent s_2; after s_2 receives these M data (packets) from s_1, s_2 first multiplies its data with M random coefficients $\psi_{i2} (1 \leq i \leq M)$, respectively, adds $\psi_{i2} D_2$ with $\psi_{i1} D_1$, respectively, and subsequently sends M new results $\psi_{i1} D_1 + \psi_{i2} D_2 (1 \leq i \leq M)$ to its parent s_3; for the subsequent nodes $s_i (3 \leq i \leq d-1)$, it does the similar multiplication-addition operations as s_2, and sends the M new results $\sum_{j=1}^{i} \psi_{1j} D_j, \sum_{j=1}^{i} \psi_{2j} D_j, \ldots, \sum_{j=1}^{i} \psi_{Mj} D_j$ to its parent s_{i+1}. Finally, after s_d receives all the M packets $\sum_{j=1}^{d-1} \psi_{1j} D_j, \sum_{j=1}^{d-1} \psi_{2j} D_j, \ldots, \sum_{j=1}^{d-1} \psi_{Mj} D_j$ for s_{d-1}, it can restore the original $D_i (1 \leq i \leq d-1)$ based on the compressive sampling theory [37]. For the used parameter M in CDG, usually $M \ll n$ for large-scale WSNs.

5.3 Zone-Based Pipeline Scheduling

Considering the benefit brought by CDG, we combine it with the pipeline technique to design an efficient CDC algorithm, named the *ZPS* algorithm. Before giving the detailed design of ZPS, we further partition the data collection tree $\mathbb{T}$ constructed in Section 4.1 into *levels* and *segments*, which are sets of cells (super nodes) and compatible zones, respectively. As shown in Section 3.2, a WSN can be partitioned into $(\lceil m/\omega \rceil)^2$ compatible zones. For these zones, we define the set $\{o_{j,i}, o_{i,j} \mid i \leq j \leq \lceil m/\omega \rceil\} (1 \leq i \leq \lceil m/\omega \rceil)$ as a *segment*, denoted by $S_i (1 \leq i \leq \lceil m/\omega \rceil)$. Within segment $S_i (1 \leq i \leq \lceil m/\omega \rceil)$, we define the set $\{s^u_{y,x}, s^u_{x,y} \mid x = (i-1) \cdot \omega + j, x \leq y \leq m\} (1 \leq j \leq \omega)$ as a *level*, denoted by $L^i_j (1 \leq j \leq \omega)$. Taking the $\mathbb{T}$ shown in Fig. 3 as an example, it can be partitioned into three segments as shown in Fig. 5, where $S_1 = \{o_{1,1}, o_{2,1}, o_{3,1}, o_{1,2}, o_{1,3}\}$, $S_2 = \{o_{2,2}, o_{3,2}, o_{2,3}\}$, and $S_3 = \{o_{3,3}\}$. Within a segment, the super nodes can be partitioned into ω levels, for example, in Fig. 5, within S_2, the super nodes can be partitioned into levels $L^2_1 = \{s^u_{4,4}, s^u_{5,4}, s^u_{6,4}, s^u_{7,4}, s^u_{8,4}, s^u_{4,5}, s^u_{4,6}, s^u_{4,7}, s^u_{4,8}\}$, $L^2_2 = \{s^u_{5,5}, s^u_{6,5}, s^u_{7,5}, s^u_{8,5}, s^u_{5,6}, s^u_{5,7}, s^u_{5,8}\}$, and $L^2_3 = \{s^u_{6,6}, s^u_{7,6}, s^u_{8,6}, s^u_{6,7}, s^u_{6,8}\}$.

Based on the definitions of segment, level and CTCS, we observe that 1) for levels $L^i_j (1 \leq i \leq \lceil m/\omega \rceil)$, all their super

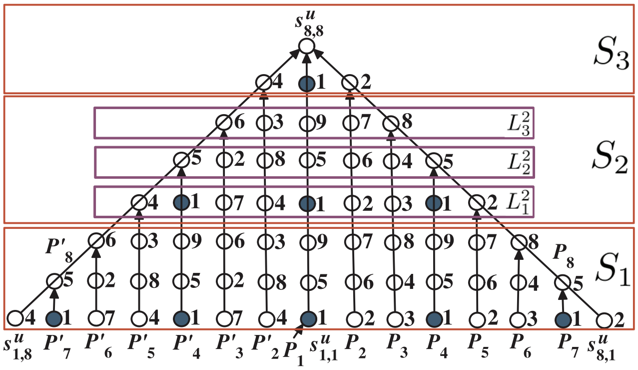


Fig. 5. Levels and segments.

nodes (cells), i.e., $\bigcup_{i=1}^{\lceil m/\omega \rceil} L^i_j$, come from CTCSs $\mathbb{S}_{j,k} \bigcup \mathbb{S}_{j,k} (j \leq k \leq \omega)$, i.e., the super nodes in $\bigcup_{i=1}^{\lceil m/\omega \rceil} L^i_j$ can be partitioned into at most $2\omega - 1$ CTCSs; and 2) on the other hand, for every super node (cell) in $\mathbb{S}_{j,k} \bigcup \mathbb{S}_{j,k} (j \leq k \leq \omega)$, it is located at level L^i_j for some $1 \leq i \leq \lceil m/\omega \rceil$. According to the observations, we design a *ZPS* algorithm for CDC, which consists of *intersegment pipeline scheduling* and *intrasegment scheduling* as follows.

Intersegment Pipeline Scheduling. Since the super nodes in levels $L^i_j (1 \leq i \leq \lceil m/\omega \rceil)$ can be partitioned into $2\omega - 1$ CTCSs, we can take each level as an unit and schedule the jth$(1 \leq j \leq \omega)$ level of all the segments $S_i (1 \leq i \leq \lceil m/\omega \rceil)$ simultaneously. In other words, we can schedule all the segments concurrently as long as we schedule the same j-th$(1 \leq j \leq \omega)$ level within each segment. Therefore, when we collect N continuous snapshots, we can pipeline the data transmission on the segments, i.e., for each segment $S_i (1 \leq i \leq \lceil m/\omega \rceil)$, S_i starts to transmit the data packets of the $(k+1)$th $(k > 0)$ snapshot immediately after it transmits all the data of the kth snapshot to segment S_{i+1}. Suppose $t(S_i)(1 \leq i \leq \lceil m/\omega \rceil)$ is the number of super time slots used by segment S_i to transmit all the data packets of a snapshot to the subsequent segment (or the sink) and let $t_p = \max\{t(S_i) \mid 1 \leq i \leq \lceil m/\omega \rceil\}$. Then, a segment data transmission pipeline on all the segments is formed with each segment works with t_p super time slots for every snapshot (Now, a snapshot is equivalent to an individual task in a traditional pipeline operation). By this data transmission pipeline, the sink can receive the data of a snapshot in every t_p super time slots after it receives the data of the first snapshot.

Intrasegment Scheduling. The intersegment pipeline scheduling provides a scheme to form a data transmission pipeline over all the segments. Clearly, the efficiency of the formed pipeline highly depends on t_p, which is determined by the intrasegment scheduling. Within segment $S_i (1 \leq i \leq \lceil m/\omega \rceil)$ to transmit the kth snapshot, we schedule the super nodes level by level, i.e., schedule $L^i_1, L^i_2, \ldots, L^i_\omega$ sequentially to transmit the kth snapshot. Finally, the data packets of the kth snapshot are transmitted to the next segment by the super nodes in level L^i_ω. When schedule $L^i_j (1 \leq j \leq \omega)$ for the kth snapshot, we first partition the super nodes in L^i_j into at most $2\omega - 1$ CTCSs according to the observations. Subsequently, we schedule these $2\omega - 1$ CTCSs sequentially. When schedule a particular CTCS, we let all the super nodes within this CTCS transmit their data in the CDG way, i.e., for every super node in this CTCS, it first does the similar multiplication-addition operations as in CDG, and then transmits the M new obtained results to its parent in the subsequent level. Thus, to schedule a CTCS in the CDG way takes M super time slots instead of one. However, this way is more suitable for the pipeline operation by avoiding the data accumulation at nodes near the sink.

In summary, for CDC, ZPS pipeline the data transmission of $\lceil m/\omega \rceil$ continuous snapshots over $\lceil m/\omega \rceil$ segments with each segment transmits a snapshot, respectively, and concurrently. For a particular snapshot transmission within a segment, it is transmitted level by level by the CDG way. Finally, the sink can receive the data of a snapshot in every t_p super time slots after it receives the data of the first snapshot.

5.4 Capacity Analysis of ZPS

In this section, we analyze the achievable data collection capacity of ZPS to collect N continuous snapshots. First, we investigate the consumed time slots to collect the first snapshot, which is the foundation of the formed data collection pipeline. Subsequently, we derive the achievable CDC capacity of ZPS in different cases.

Lemma 12. *1) For the t_p in the intersegment pipeline scheduling of ZPS, $t_p \le \omega(2\omega - 1)M$; 2) The number of super time slots used to collect the first snapshot is at most $\lceil \frac{m}{\omega} \rceil \omega(2\omega - 1)M$.*

Proof. 1. According to the intrasegment scheduling, the super nodes in each level of a segment can be partitioned into at most $2\omega - 1$ CTCSs. Moreover, for the super nodes within each CTCS, they transmit their data in the CDG way, i.e., each CTCS can be scheduled within M super time slots. Further, each segment contains at most ω levels, which implies for a single snapshot, a segment can be scheduled within $\omega(2\omega - 1)M$ super time slots, i.e., $t_p \le \omega(2\omega - 1)M$.

2. Based on step 1, the number of super time slots used to collect the first snapshot is at most $\lceil \frac{m}{\omega} \rceil \omega(2\omega - 1)M$, since a WSN can be partitioned into at most $\lceil \frac{m}{\omega} \rceil$ segments. □

Based on Lemma 12, we can derive the achievable CDC capacity of ZPS in different cases as shown in Theorem 3.

Theorem 3. *To collect N continuous snapshots, the achievable network capacity of ZPS is*

$$\begin{cases} \Omega\left(\frac{N\sqrt{n}}{6\sqrt{2}\omega M \sqrt{\log n} \ln n} \cdot W\right), & \text{if } N = O(\sqrt{n/\log n}), \\ \Omega\left(\frac{n}{12\omega^2 M \log n \ln n} \cdot W\right), & \text{if } N = \Omega(\sqrt{n/\log n}), \end{cases}$$

in the sense of the worst case, and

$$\begin{cases} \Omega\left(\frac{p_o N\sqrt{n/\log n}}{2\sqrt{2}\omega M} \cdot W\right), & \text{if } N = O(\sqrt{n/\log n}), \\ \Omega\left(\frac{p_o n}{4\omega^2 M \log n} \cdot W\right), & \text{if } N = \Omega(\sqrt{n/\log n}), \end{cases}$$

in the sense of expectation.

Proof. To collect N continuous snapshots, the data transmission process can be pipelined according to ZPS, which implies the sink can receive the data of a snapshot every t_p super time slots after it receives the first snapshot. Therefore, by Lemma 12, the number of super time slots used to collect N continuous snapshots is at most $\lceil \frac{m}{\omega} \rceil \omega(2\omega - 1)M + (N-1)\omega(2\omega-1)M \le (\frac{m}{\omega} + 1) \cdot 2\omega^2 M + 2\omega^2(N-1)M = O(2\omega m M + 2\omega^2 NM)$. □

Thus, in the sense of the worst case, the achievable network capacity of ZPS is at least

$$\frac{NnB}{O(2\omega m M + 2\omega^2 NM) \cdot 10 \log n \cdot t_w} \tag{17}$$

$$= \frac{NnW}{O(20\omega m M \hbar \log n + 20\omega^2 \hbar NM \log n)} \tag{18}$$

$$= \frac{NnW}{O(10\omega M \hbar \sqrt{n \log n} + 20\omega^2 \hbar NM \log n)} \tag{19}$$

$$= \begin{cases} \Omega\left(\frac{N\sqrt{n}}{10\omega M \sqrt{\log n} \ln n} \cdot W\right), & \text{if } N = O(\sqrt{n/\log n}), \\ \Omega\left(\frac{n}{20\omega^2 M \log n \ln n} \cdot W\right), & \text{if } N = \Omega(\sqrt{n/\log n}). \end{cases} \tag{20}$$

Similarly, in the sense of expectation, the achievable network capacity of ZPS is at least

$$\frac{NnB}{O(2\omega m M + 2\omega^2 NM) \cdot 4 \log n \cdot t_m} \tag{21}$$

$$= \frac{p_o NnW}{O(8\omega m M \log n + 8\omega^2 NM \log n)} \tag{22}$$

$$= \frac{p_o NnW}{O(4\omega M \sqrt{n \log n} + 8\omega^2 NM \log n)} \tag{23}$$

$$= \begin{cases} \Omega\left(\frac{p_o N\sqrt{n/\log n}}{4\omega M} \cdot W\right), & \text{if } N = O(\sqrt{n/\log n}), \\ \Omega\left(\frac{p_o n}{8\omega^2 M \log n} \cdot W\right), & \text{if } N = \Omega(\sqrt{n/\log n}). \end{cases} \tag{24}$$

From Theorem 3, we know that 1) the achievable network capacity of ZPS is $\frac{N\sqrt{n}}{\sqrt{\log n} \ln n}$ or $\frac{n}{\log n \ln n}$ times better than the optimal capacity of the snapshot data collection scenario in order in the sense of the worst case, and $\sqrt{\frac{n}{\log n}}$ or $\frac{n}{\log n}$ times better than the optimal capacity of the snapshot data collection scenario in order in the sense of expectation, which are very significant improvements. By examining ZPS carefully, we find that two main reasons are responsible for this improvement. The primary reason is the use of the CDG technique, which distributes the traffic load evenly over the entire WSN, and then the data accumulation at the nodes near the sink is avoided. Another reason is the *pipeline scheduling*. According to ZPS, the time overlap of the data collection of multiple continuous snapshots in the transmission pipeline conserves a lot of time, which accelerates the network capacity directly and significantly; 2) ZPS will be more effective for large-scale WSNs, since large scale WSNs incur large data collection trees, which are more suitable for pipeline; and

3) ZPS is also more effective for long-term CDC. The longer the CDC process is, the closer for ZPS to its theoretical achievable network capacity.

6 Conclusion

For most existing works studying the network capacity issue, their designs and analysis are based on the deterministic network model. However, in real applications, this deterministic network model assumption is not practical due to the "transitional region phenomenon." Actually, a more practical network model for WSNs is the probabilistic network model, where a transmission over a link is conducted successfully with a probability instead of being determined. Unfortunately, few of the existing works study the data collection capacity issue for WSNs under the probabilistic network model, i.e., for probabilistic WSNs, until now. To fill in this gap, we investigate the achievable snapshot and CDC capacities for probabilistic WSNs in this paper. For SDC, we propose a novel CPS algorithm, which schedules multiple super nodes on multiple paths concurrently. Theoretical analysis of CPS shows that its achievable network capacity is order-optimal in the sense of expectation and has $O(\ln n)$ of degradation in the sense of the worst case. For CDC, we propose a *ZPS* algorithm. ZPS significantly speeds up the CDC process by forming a data transmission pipeline, and achieves a surprising network capacity. The simulation results also validate that the proposed algorithms significantly improve network capacity compared with the existing works.

Acknowledgments

This work was partly supported by the US National Science Foundation under grants CNS-1152001, CNS-1052769, and CNS-1252292.

References

[1] S. Ji, A.S. Uluagac, R. Beyah, and Z. Cai, "Practical Unicast and Convergecast Scheduling Schemes for Cognitive Radio Networks," *J. Combinatorial Optimization,* vol. 26, pp. 161-177, 2012.

[2] J. Li and S. Cheng, "(ϵ, δ)-Approximate Aggregation Algorithms in Dynamic Sensor Networks," *IEEE Trans. Parallel & Distributed Systems,* vol. 23, no. 3, pp. 385-396, Mar. 2012.

[3] S. Cheng, J. Li, and Z. Cai, "$O(\epsilon)$-Approximation to Physical World by Sensor Networks," *Proc. IEEE INFOCOM* 2013.

[4] A. Boukerche, X. Cheng, and J. Linus, "Energy-Aware Data-Centric Routing in Microsensor Networks," *Proc. Sixth ACM Int'l Workshop Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM),* 2003.

[5] P.-J. Wan, S.C.-H. Huang, L. Wang, Z. Wan, and X. Jia, "Minimum-Latency Aggregation Scheduling in Multihop Wireless Networks," *Proc. ACM MobiHoc,* 2009.

[6] J. He, S. Ji, Y. Pan, and Y. Li, "Greedy Construction of Load-Balanced Virtual Backbones in Wireless Sensor Networks," *Wireless Comm. and Mobile Computing,* 2012.

[7] J. He, S. Ji, Y. Pan, and Z. Cai, "Approximation Algorithms for Load-Balanced Virtual Backbone Construction in Wireless Sensor Networks," *Theoretical Computer Science,* 2012.

[8] M. Yan, J. He, S. Ji, and Y. Li, "Multi-Regional Query Scheduling in Wireless Sensor Networks with Minimum Latency," *Wireless Comm. and Mobile Computing,* 2012.

[9] X. Zhu, B. Tang, and H. Gupta, "Delay Efficient Data Gathering in Sensor Networks," *Proc. First Int'l Conf. Mobile Ad-Hoc and Sensor Networks (MSN),* 2005.

[10] S. Chen, S. Tang, M. Huang, and Y. Wang, "Capacity of Data Collection in Arbitrary Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2010.

[11] X.-Y. Li, S.-J. Tang, and O. Frieder, "Multicast Capacity for Large Scale Wireless Ad Hoc Networks," *Proc. ACM MobiCom,* 2007.

[12] X. Mao, X.-Y. Li, and S. Tang, "Multicast Capacity for Hybrid Wireless Networks," *Proc. ACM MobiHoc,* 2008.

[13] Z. Wang, H.R. Sadjadpour, and J.J. Garcia-Luna Aceves, "A Unifying Perspective on the Capacity of Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM,* 2008.

[14] S. Chen, Y. Wang, X.-Y. Li, and X. Shi, "Order-Optimal Data Collection in Wireless Sensor Networks: Delay and Capacity," *Proc. IEEE Comm. Soc. Sixth Ann. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON),* 2009.

[15] S. Ji, R. Beyah, and Y. Li, "Continuous Data Collection Capacity of Wireless Sensor Networks under Physical Interference Model," *Proc. IEEE Eighth Int'l Conf. Mobile Ad-Hoc and Sensor Systems (MASS),* 2011.

[16] S. Ji, J. He, A.S. Uluagac, R. Beyah, and Y. Li, "Cell-Based Snapshot and Continuous Data Collection in Wireless Sensor Networks," *ACM Trans. Sensor Networks,* vol. 9, article 47, 2012.

[17] S. Ji, R. Beyah, and Z. Cai, "Snapshot/Continuous Data Collection Capacity for Large-Scale Probabilistic Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2012.

[18] M. Garetto, P. Giaccone, and E. Leonardi, "On the Capacity of Ad Hoc Wireless Networks under General Node Mobility," *Proc. IEEE INFOCOM,* 2007.

[19] G. Sharma, R. Mazumdar, and N.B. Shroff, "Delay and Capacity Trade-Offs in Mobile Ad Hoc Networks: A Global Perspective," *IEEE/ACM Trans. Networking,* vol. 15, no. 5, pp. 981-992, Oct. 2007.

[20] W. Huang, X. Wang, and Q. Zhang, "Capacity Scaling in Mobile Wireless Ad Hoc Network with Infrastructure Support," *Proc. IEEE 30th Int'l Conf. Distributed Computing Systems (ICDCS),* 2010.

[21] V. Bhandari and N.H. Vaidya, "Connectivity and Capacity of Multi-Channel Wireless Networks with Channel Switching Constraints," *Proc. IEEE INFOCOM,* 2007.

[22] V. Bhandari and N.H. Vaidya, "Capacity of Multi-Channel Wireless Networks with Random (c, f) Assignment," *Proc. ACM MobiHoc,* 2007.

[23] V. Ramamurthi, S.K.C. Vadrevu, A. Chaudhry, and M.R. Bhatnagar, "Multicast Capacity of Multi-Channel Multihop Wireless Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC),* 2009.

[24] H.-N. Dai, K.-W. Ng, R.C.-W. Wong, and M.-Y. Wu, "On the Capacity of Multi-Channel Wireless Networks Using Directional Antennas," *Proc. IEEE INFOCOM,* 2008.

[25] G. Zhang, Y. Xu, X. Wang, and M. Guizani, "Capacity of Hybrid Wireless Networks with Directional Antenna and Delay Constraint," *IEEE Trans. Comm.,* vol. 58, no. 7, pp. 2097-2106, July 2010.

[26] M. Guo, X. Wang, and M. Wu, "On the Capacity of k-MPR Wireless Networks," *IEEE Trans. Wireless Comm.,* vol. 8, no. 7, pp. 3878-3886, July 2009.

[27] S. Chen, Y. Wang, X.-Y. Li, and X. Shi, "Data Collection Capacity of Random-Deployed Wireless Sensor Networks," *Proc. IEEE GlobeCom,* 2009.

[28] O. Goussevskaia, R. Wattenhofer, M.M. Halldorsson, and E. Welzl, "Capacity of Arbitrary Wireless Networks," *Proc. IEEE INFOCOM,* 2009.

[29] M. Andrews and M. Dinitz, "Maximizing Capacity in Arbitrary Wireless Networks in the SINR Model: Complexity and Game Theory," *Proc. IEEE INFOCOM,* 2009.

[30] S. Li, Y. Liu, and X.-Y. Li, "Capacity of Large Scale Wireless Networks under Gaussian Channel Model," *Proc. ACM MobiCom,* 2008.

[31] U. Niesen, P. Gupta, and D. Shah, "On Capacity Scaling in Arbitrary Wireless Networks," *IEEE Trans. Information Theory,* vol. 55, no. 9, pp. 3959-3982, Sept. 2009.

[32] U. Niesen, P. Gupta, and D. Shah, "The Balanced Unicast and Multicast Capacity Regions of Large Wireless Networks," *IEEE Trans. Information Theory,* vol. 56, no. 5, pp. 2249-2271, May 2010.

[33] X. Wang, Y. Bei, Q. Peng, and L. Fu, "Speed Improves Delay-Capacity Tradeoff in MotionCast," *IEEE Trans. Parallel and Distributed Systems,* vol. 22, no. 5, pp. 729-742, May 2011.

[34] Z. Wang, H.R. Sadjadpour, and J.J. Garcia-Luna Aceves, "The Capacity and Energy Efficiency of Wireless Ad Hoc Networks with Multi-Packet Reception," *Proc. ACM MobiHoc,* 2008.

[35] X.-Y. Li, J. Zhao, Y.W. Wu, S.J. Tang, X.H. Xu, and X.F. Mao, "Broadcast Capacity for Wireless Ad Hoc Networks," *Proc. IEEE Fifth Int'l Conf. Mobile Ad Hoc and Sensor Systems (MASS)*, 2008.

[36] P. Gupta and P.R. Kumar, "The Capacity of Wireless Networks," *IEEE Trans. Information Theory*, vol. 46, no. 2, pp. 388-404, Mar. 2000.

[37] C. Luo, F. Wu, J. Sun, and C.W. Chen, "Compressive Data Gathering for Large-Scale Wireless Sensor Networks," *Proc. ACM MobiCom*, 2009.

[38] C.-K. Chau, M. Chen, and S.C. Liew, "Capacity of Large-Scale CSMA Wireless Networks," *Proc. ACM MobiCom*, 2009.

[39] Y. Xu and W. Wang, "Scheduling Partition for Order Optimal Capacity in Large-Scale Wireless Networks," *Proc. ACM MobiCom*, 2009.

[40] T. Moscibroda, "The Worst-Case Capacity of Wireless Sensor Networks," *Proc. Sixth Int'l Conf. Information Processing in Sensor Networks (IPSN)*, 2007.

[41] S.R. Kulkarni and P. Viswanath, "A Deterministic Approach to Throughput Scaling in Wireless Networks," *IEEE Trans. Information Theory*, vol. 50, no. 6, pp. 1041-1049, June 2004.

[42] O. Dousse, D.N.C. Tse, and P. Thiran, "Closing the Gap in the Capacity of Wireless Networks via Percolation Theory," *IEEE Trans. Information Theory*, vol. 53, no. 3, pp. 1009-1018, Mar. 2007.

[43] S. Ji, Y. Li, and X. Jia, "Capacity of Dual-Radio Multi-Channel Wireless Sensor Networks for Continuous Data Collection," *Proc. IEEE INFOCOM*, 2011.

[44] S. Ji, Z. Cai, Y. Li, and X. Jia, "Continuous Data Collection Capacity of Dual-Radio Multi-Channel Wireless Sensor," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 10, pp. 1844-1855, Oct. 2012.

[45] S. Ji and Z. Cai, "Distributed Data Collection and Its Capacity in Asynchronous Wireless Sensor Networks," *Proc. IEEE INFOCOM*, 2012.

[46] S. Ji and Z. Cai, "Distributed Data Collection in Large-Scale Asynchronous Wireless Sensor Networks under the Generalized Physical Interference Model," *IEEE/ACM Trans. Networking*, vol. 21, no. 4, pp. 1270-1283, Aug. 2013.

[47] Z. Cai, S. Ji, J. He, and A.G. Bourgeois, "Optimal Distributed Data Collection for Asynchronous Cognitive Radio Networks," *Proc. IEEE 32nd Int'l Conf. Distributed Computing Systems (ICDCS)*, 2012.

[48] S.C.-H. Huang, S.Y. Chang, H.-C. Wu, and P.-J. Wan, "Analysis and Design of a Novel Randomized Broadcast Algorithm for Scalable Wireless Networks in the Interference Channels," *IEEE Trans. Wireless Comm.*, vol. 9, no. 7, pp. 2206-2215, July 2010.

[49] A. Woo, T. Tong, and D. Estrin, "Taming the Underlying Challenges for Reliable Multihop Routing in Sensor Networks," *Proc. First Int'l Conf. Embedded Networked Sensor Systems (SenSys)*, 2003.

[50] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," *Proc. First Int'l Conf. Embedded Networked Sensor Systems (SenSys)*, 2003.

[51] G. Zhao, T. He, S. Krishnamurthy, and J. Stankovic, "Impact of Radio Irregularity on Wireless Sensor Networks," *Proc. ACM Second Int'l Conf. Mobile Systems, Applications, and Services (MobiSys)*, 2004.

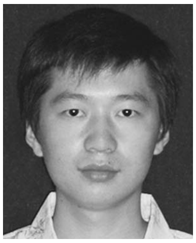
[52] A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin, "Statistical Model of Lossy Links in Wireless Sensor Networks," *Proc. Fourth Int'l Conf. Information Processing in Sensor Networks (IPSN)*, 2005.

[53] A. Cerpa, J.L. Wong, M. Potkonjak, and D. Estrin, "Temporal Properties of Low Power Wireless Links: Modeling and Implications on Multi-Hop Routing," *Proc. ACM MobiHoc*, 2005.

[54] K.-W. Chin, J. Judge, A. Williams, and R. Kermode, "Implementation Experience with Manet Routing Protocols," *ACM SIGCOMM Computer Comm. Rev.*, vol. 11, pp. 49-59, 2002.

[55] Y. Liu, Q. Zhang, and L.M. Ni, "Opportunity-Based Topology Control in Wireless Sensor Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 21, no. 3, pp. 405-416, Mar. 2010.

[56] M. Zuniga and B. Krishnamachari, "Analyzing the Transitional Region in Low Power Wireless Links," *Proc. IEEE Comm. Soc. First Ann. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON)*, 2004.



Shouling Ji received the BS and MS degrees in computer science from Heilongjiang University, China, in 2007 and 2010, respectively. He is currently working toward the PhD degree in the Department of Computer Science at Georgia State University. His research interests include wireless sensor networks, cognitive radio networks, and network evolution analysis of social networks. He is also the membership chair of the IEEE Student Branch at Georgia State. He is currently a student member of the ACM, IEEE, and IEEE Communications Society.

Raheem Beyah received the BS degree in electrical engineering from North Carolina A&T State University in 1998, and the master's and PhD degrees in electrical and computer engineering from Georgia Tech in 1999 and 2003, respectively. He is an associate professor in the School of Electrical and Computer Engineering at Georgia Tech, where he leads the Communications Assurance and Performance Group (CAP) and is a member of the Communications Systems Center (CSC). His research interests include network security, wireless networks, network traffic characterization and performance, and security visualization. He received the US National Science Foundation CAREER Award in 2009. He is a member of the NSBE, ASEE, and a senior member of the ACM and IEEE.

Zhipeng Cai received the BS degree in computer science from the Beijing Institute of Technology, China, and the MS and PhD degrees in computing science from the University of Alberta, Canada. He is currently an assistant professor in the Department of Computer Science at the Georgia State University. His research interests include wireless networks and optimization theory. He has received a number of awards and honors, including a US National Science Foundation CAREER Award and an NSERC Postdoc Fellowship. He is a member of the IEEE.