2011 Eighth IEEE International Conference on Mobile Ad-Hoc and Sensor Systems

Continuous Data Collection Capacity of Wireless Sensor Networks Under Physical Interference Model

Shouling Ji
Department of Computer Science
Georgia State University
Atlanta, Georgia 30303, USA
Email: sji@cs.gsu.edu

Raheem Beyah
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30308, USA
Email: rbeyah@ece.gatech.edu

Yingshu Li
Department of Computer Science
Georgia State University
Atlanta, Georgia 30303, USA
Email: yli@cs.gsu.edu

***Abstract*—*Data collection* is a common operation of Wireless Sensor Networks (WSNs). The performance of data collection can be measured by its achievable *network capacity*. However, most existing works focus on the network capacity of *unicast*, *multicast* or/and *broadcast*, which are different communication modes from data collection, especially continuous data collection. In this paper, we study the *Snapshot/Continuous Data Collection* (SDC/CDC) problem under the Physical Interference Model (PhIM) for randomly deployed dense WSNs. For SDC, we propose a *Cell-Based Path Scheduling* (CBPS) algorithm based on network partitioning. Theoretical analysis shows that its achievable network capacity is $\Omega(W)$ (W is the data transmitting rate, *i.e.* bandwidth, over a channel), which is order-optimal. For CDC, we propose a novel *Segment-Based Pipeline Scheduling* (SBPS) algorithm that significantly speeds up the CDC process, and achieves a surprising network capacity, which is at least $\sqrt{\frac{n}{\log n}}$ or $\frac{n}{\log n}$ times better than the current best result.**

I. Introduction

Wireless Sensor Networks (WSNs) are mainly used for collecting data from the physical world. Data gathering can be categorized as *data aggregation* [1], which obtains aggregated values from WSNs, *e.g.* maximum, minimum or/and average value of all the data, and *data collection* [2][17][30], which gathers all the data from a network without any data aggregation. For data collection, the union of all the sensing values from all the sensors at a particular time instance is called a *snapshot* [2][4]. The problem of collecting one snapshot is called *snapshot data collection* (SDC). The problem of collecting multiple continuous snapshots is called *continuous data collection* (CDC). To evaluate the network performance, *network capacity*, which can reflect the achievable data transmission rate, is usually used [2]-[32]. For unicast, multicast and broadcast, we use *unicast capacity*, *multicast capacity*, and *broadcast capacity* to denote the network capacity, respectively. Particularly, for a data collection WSN, 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 is collected by the sink[1] [2][4].

After the first work [28] in this area, many works emerged to study the network capacity issue for a variety of network scenarios, *e.g.* multicast capacity [9][20], unicast capacity [23], broadcast capacity [25][26][27], SDC capacity [2][17][30]. When researchers studied the network capacity of wireless networks, one of two interference models is usually used, *i.e.* the *Protocol Interference Model* (PrIM) [2]-[16] or the *Physical Interference Model* (PhIM) [17]-[24]. Under the PrIM, two nodes can successfully communicate if and only if the receiver is within the transmission range of the sender, and the receiver is out of the interference range of other simultaneous senders. Under the PhIM, a receiver can successfully receive data from the sender if and only if the Signal-to-Interference-and-Noise-Ratio (SINR) of the sender at the receiver is no less than a threshold (denoted by η). The PrIM simplifies the communication model of wireless networks and is therefore more convenient for analysis. By contrast, the PhIM takes the received physical signal strength as a criterion, which is more accurate and reliable. In this paper, we consider the achievable data collection capacity for WSNs under the PhIM.

For most of the existing works, they studied the multicast capacity [9][20], the unicast capacity [23], and/or the broadcast capacity [26] of wireless networks. In contrast, we study the SDC capacity and the CDC capacity of large-scale WSNs. Mulitcast/unicast/broadcast and data collection are different communication modes. Furthermore, the data collection communication mode introduces more communication traffic and wireless interference. Recently, the authors in [3][4] considered CDC under the PrIM. In [3], the authors proposed a method that combines the SDC scheduling and the pipeline technology to carry out CDC. However, the authors also proved that their CDC method cannot achieve a better network capacity compared with their SDC method. In our previous work [4], we studied the CDC problem in dual-radio multi-channel WSNs under the PrIM. To further study

[1]Without confusion, we use data collection capacity and network capacity interchangeably in the following of this paper.

978-0-7695-4469-4/11 $26.00 © 2011 IEEE
DOI 10.1109/MASS.2011.29

how to significantly improve the CDC capacity under the PhIM, we investigate the restriction that limits the pipeline technology to achieve a higher CDC capacity. By combining the *Compressive Data Gathering* (CDG) technique [30] and the pipeline technology, we propose a new transmission and scheduling method for CDC which achieves a surprising network capacity. Particularly, the main contributions of this work are as follows:

- For a WSN deployed in a square area, we first partition the network into small *cells*, and then abstract every cell as a super-node in a data collection tree rooted at the sink. Based on the data collection tree, we propose a scheduling algorithm, called *Cell-Based Path Scheduling* (CBPS), for the SDC problem in WSNs. Theoretical analysis of CBPS shows that the achievable network capacity is $\Omega(W)$, where W is the data transmission rate over a wireless channel, *i.e.* the channel bandwidth. Since the upper bound of the SDC capacity is shown to be W [2][4], CBPS successfully achieves the order-optimal network capacity.
- We propose a novel *Segment-Based Pipeline Scheduling* (SBPS) method for CDC in WSNs. SBPS combines the CDG [30] technology and the pipeline technology and can significantly improve achievable network capacity. We theoretically prove that the asymptotic achievable network capacity of SBPS to collect N continuous snapshots is $\Omega(\sqrt{\frac{n}{\log n}}W)$ when $N \leq \sqrt{\frac{n}{\log n}}$ or $\Omega(\frac{n}{\log n}W)$ when $N > \sqrt{\frac{n}{\log n}}$, where n is the number of sensors in a WSN. Since the current best result is $\Omega(W)$ [2][17], our result is at least $\sqrt{\frac{n}{\log n}}$ or $\frac{n}{\log n}$ times better than the best result, which is a very significant improvement.

II. Related Works

Following the first work [28] by Gupta and Kumar, many works emerged to study the network capacity issue.

A. Network Capacity under the PrIM

The authors of [2][3] derived the upper and lower bounds of data collection capacity under the PrIM for arbitrary WSNs. In [9], the authors investigated the multicast capacity for large scale wireless ad hoc networks. They showed that the network multicast capacity is $\Theta(\sqrt{\frac{n}{\log n}} \cdot \frac{W}{k})$ when $k = O(\frac{n}{\log n})$ and is $\Theta(W)$ when $k = \Omega(\frac{n}{\log n})$. A more general (n, m, k)-casting capacity problem under the PrIM was investigated in [10], where n, m and k denote the total number of the nodes in the network, the number of destinations of each communication group, and the actual number of communication-group members that receive information, respectively. In [10], the upper and lower bounds for the (n, m, k)-casting capacity were obtained for random wireless networks.

A general framework to characterize the network capacity of wireless ad hoc networks with arbitrary mobility patterns was studied in [11]. By relaxing the "homogeneous mixing" assumption in most existing works, the network capacity of a heterogeneous network was analyzed. The authors in [12] studied the relationship between the network capacity and the delay of mobile wireless ad hoc networks. They derived how much delay must be tolerated under a certain mobile pattern to achieve an improvement of the network capacity. In another similar work [13], the authors investigated the network capacity scaling in mobile wireless ad hoc networks under the PrIM with infrastructure support.

In [14], the authors studied the connectivity and network capacity problems of multi-channel wireless networks under the PrIM. They considered a multi-channel wireless network with constraints on channel switching, proposed some routing and channel assignment strategies for multiple unicast communications, and derived the per-flow capacity. In [15], the authors first proposed a multi-channel network architecture, called MC-MDA, and then obtained the capacity of multiple unicast communications under the PrIM for arbitrary and random wireless networks. In a similar work [16], the authors studied the network capacity of hybrid wireless networks with directional antenna and delay constraints. Unlike previous works, the authors in [29] studied the capacity of multi-unicast for wireless networks from the algorithmic aspects, and they designed provably good algorithms for arbitrary instances. The broadcast capacity of wireless networks under the PrIM is investigated in [25], where the authors derived the upper and lower bounds of the broadcast capacity in arbitrary connected networks.

When the authors in [5] studied the data gathering capacity of wireless networks under the PRIM, they concerned the per source node throughput in a network where a subset of nodes send data to some designated destinations while other nodes serve as relays. To gather data from WSNs, a multi-query processing technology is proposed in [6]. In that work, the authors considered how to obtain data efficiently with data aggregation and query scheduling. Under different communication organizations, the authors in [7] derived the many-to-one capacity bound under the PrIM. Another work studied the many-to-one capacity issue for WSNs is [8], where the authors considered to use data compression to improve the data gathering efficiency. They also studied the relation between a data compression scheme and the data gathering quality.

B. Network Capacity under the PhIM

In [17], the authors investigated the data collection capacity for WSNs under the PhIM. They proposed a grid partition method which divides the network into small grids to collect data and then derived the network capacity. In [18], the authors considered the scheduling problem, where all the communication requests are single-hop and all the

nodes transmit at a fixed power level. They proposed an algorithm to maximize the number of links in one time-slot. Unlike [18], the authors in [19] considered the power-control problem. A family of approximation algorithms were presented to maximize the network capacity of arbitrary wireless networks. In [20], the authors showed that when $k \leq \theta_1 \frac{n}{(\log n)^{2\alpha+6}}$ and $n_s \geq \theta_2 n^{1/2+\beta}$, the capacity that each multicast session can achieve is at least $c_8 \frac{\sqrt{n}}{n_s\sqrt{k}}$. In [21], the authors studied the scaling laws of WSNs based on an antenna sharing idea. In that work, the author derived the many-to-one capacity bounds under different power constraints.

In [23], the authors studied the balanced unicast and multicast capacity of a wireless network consisting of n randomly placed nodes, and obtained the characterization of the scaling of the n^2-dimensional balanced unicast and $n2^n$-dimensional balanced multicast capacity regions under the Gaussian fading channel model. In [24], the authors studied the multicast capacity of MANETs under the PhIM, called motioncast. They considered the network capacity of MANETs in two particular situations, which are the LSRM (local-based speed-restricted) model and the GSRM (global-based speed-restricted) model. In [26] and [27], the authors studied the broadcast capacity of wireless networks, where they obtained the broadcast capacity bounds under the (general) PhIM. The multi-unicast capacity of wireless networks is studied in [22] via percolation theory. By applying percolation theory, the authors obtained a tighter capacity bound for arbitrary wireless networks.

The authors of [30]-[32] considered both the PrIM and the PhIM when they studied the network capacity for wireless networks. The work in [30] studies how to distribute the data collection task to the entire network for WSNs to achieve load balancing. In this work, all the sensors transmit the same number of data packets during the data collection process. The authors in [31] studied the network capacity of CSMA wireless networks. They formulated the models of a series of CSMA protocols and studied the network capacity of CSMA scheduling versus TDMA scheduling. In [32], a scheduling partition method for large-scale wireless networks was proposed. This method decomposes a large network into many small zones, and then localized scheduling algorithms which can achieve the order optimal network capacity as a global scheduling strategy are executed in each zone independently.

III. Network Partition

A. Network Model

In this paper, we consider a WSN consisting of n sensors, denoted by $s_1, s_2, \ldots, s_n$, and one sink deployed in a square with area $A = cn$, where c is a constant. Furthermore, we assume the distribution of all the sensors is independent and identically distributed (i.i.d.) and loosing only a constant factor, the sink is located at the top-right corner of the square[2]. The communication radius of a sensor is r. In each time interval, every sensor generates a data packet with size b bits, and transmits its data to the sink in a multi-hop fashion over a single common wireless channel with bandwidth W bits/second, *i.e.* the data transmission rate of the common channel is W. We further assume the network time is slotted into time slots with each of length $t = b/W$ seconds. To accurately represent the wireless interference in a WSN, we consider the data collection problem under the PhIM, where a receiver s_j successfully receives the transmitted data from the sender s_i if and only if the SINR of s_i at s_j is no less than a constant $\eta > 0$, *i.e.*

$$SINR = \frac{P_i \cdot (\|s_i - s_j\|)^{-\alpha}}{N_0 + \sum_{k \neq i} P_k (\|s_k - s_j\|)^{-\alpha}} \geq \eta,$$

where P_i is the transmission power of s_i, $\|s_i - s_j\|$ is the Euclidean distance between s_i and s_j, α is the path-loss exponent and usually $\alpha \in [3, 5]$, $N_0 > 0$ is a constant representing the background noise, and P_k is the transmission power of concurrent sender $s_k (1 \leq k \leq n, k \neq i)$. In this paper, we assume all the sensors have the same transmission power P when they transmit data and therefore we have the communication radius r of a sensor satisfying $r \leq (\frac{P}{\eta N_0})^{1/\alpha}$.

We further formally define the achievable data collection capacity C as the ratio between the amount of data successfully collected by the sink and the time τ used to collect these data. For instance, in our WSN model, the achievable C to collect N continuous snapshots of data is defined by $C = Nnb/\tau$, which is actually the data receiving rate at the sink. Particulary, when $N = 1$, $C = nb/\tau$ is the SDC capacity.

B. Network Partition

In the previous subsection, we assume the network is distributed in a square with area size $A = cn$. We partition the network into small square cells with edge length $\sqrt{2c \log n}$, denoted by l, by a group of horizontal and vertical lines. The resulting network is shown in Figure 1. In order for the sensors in a cell to transmit their data to the sensors in the neighboring cells, we set a sensor's communication range $r = 2\sqrt{2}l$. Furthermore, we use $\lambda = \sqrt{cn}/\sqrt{2c \log n} = \sqrt{n/2 \log n}$ to denote the number of cells in each row/column and define $\lambda' = \lambda - 1$. For the cells shown in Figure 1, we assign each cell positive integer coordinates $(i, j)(1 \leq i, j \leq \lambda)$, and a cell with coordinates (i, j) is denoted by $\kappa_{i,j}$. Hence, the coordinates of the bottom-left corner cell are $(1, 1)$ and the coordinates of the upper-right corner cell are (λ, λ).

[2]Note that when the sink is in the middle of the network, one achieves a $1/4$ of data collection capacity of the sink in the corner.

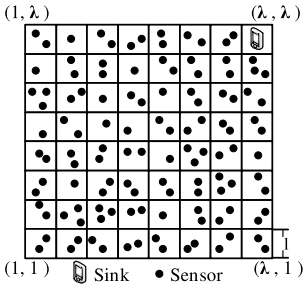

Figure 1. Network partition.

Based on the above network partition, the following two lemmas can be derived by similar techniques as those used in [33]. We omit the proofs of Lemma 1 and Lemma 2 due to space limitation.

Lemma 1: For any cell $\kappa_{i,j}$, let e_{ij} denote the random event that cell $\kappa_{i,j}$ is empty, *i.e.* no sensor is located at cell $\kappa_{i,j}$. Then, the probability that at least one cell is empty is no more than $\frac{1}{2n\log n}$, *i.e.* $\Pr[\bigcup_{1\leq i,j\leq\lambda} e_{ij}] \leq \frac{1}{2n\log n}$.

Lemma 1 implies that when $n \to \infty$, the probability that one cell is empty is zero. Therefore, when n is a large value, we can assume that there is at least one sensor located in every cell.

Lemma 2: For any cell $\kappa_{i,j}$, let the random variable Z_{ij} denote the number of sensors in it. Then, the probability that cell $\kappa_{i,j}$ contains more than $8\log n$ sensors is no more than $\frac{1}{n^2}$, *i.e.* $\Pr[Z_{ij} > 8\log n] \leq \frac{1}{n^2}$.

From Lemma 2, the probability that a cell contains more than $8\log n$ sensors is zero when $n \to \infty$. Hence, for a large n, we use $8\log n$ as the upper bound of the number of sensors located in a cell.

C. Interference Zone

Since we assume the sink is located at the upper-right corner cell $\kappa_{\lambda,\lambda}$, for the sensors in cell $\kappa_{i,j}$, they will forward their data to the sensors located at cells $\kappa_{i+1,j}, \kappa_{i,j+1}$ or/and $\kappa_{i+1,j+1}$ as shown in Figure 2, *i.e.* the sensors in each cell will forward their data to sensors in subsequent cells horizontally, vertically, or/and diagonally. Finally, the data generated by all the sensors will be forwarded to the sink via this multi-hop fashion.

Figure 2. Data transmission mode.

When data transmission is initialized between two neighboring cells, they may incur interference caused by other concurrent data transmissions. To make all the concurrent data transmissions successful, we further partition the network into larger square zones with side length $R = \omega \cdot l$ (to avoid radio conflicts, $\omega > 2$, *i.e.* $R \geq 3l$) by another group of horizontal and vertical lines and we call these square zones *interference zones* as shown in Figure 3. We also assign each interference zone integer coordinates $(i,j)(1 \leq i,j \leq \lceil\sqrt{cn}/R\rceil)$ and interference zone (i,j) is denoted by $o_{i,j}$. For a cell $\kappa_{i',j'}$ in an interference zone $o_{i,j}$, the *relative position* of $\kappa_{i',j'}$ in $o_{i,j}$ is defined as $(i' \cdot l - (i-1)\cdot R, j' \cdot l - (j-1)\cdot R)$. We call the cells having the same relative positions in different interference zones *compatible cells*. In Figure 3, compatible cells having relative position (l,l) are highlighted. If two sensors are in different cells which are compatible cells, then they can transmit data simultaneously without incurring any interference.

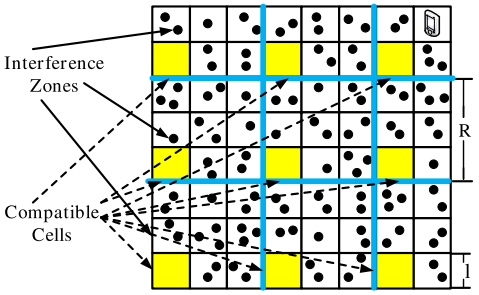

Figure 3. Interference zones and compatible cells.

At any time, we select one sensor in each compatible cell to transmit data. The selected sensors transmit data simultaneously. These transmitting sensors will not incur interference, since they are spread in different compatible cells. To this end, the next task is to decide the value of R as shown in Theorem 1.

Theorem 1: If we partition the network into interference zones with edge length $R = \omega \cdot l$, where $\omega = 2\sqrt{2}\cdot\sqrt[\alpha]{c_1\eta}$ is a constant value, it can be guaranteed that the sensors in compatible cells can simultaneously and successfully transmit data without interference, with each transmitting sensor residing in a unique compatible cell.

Proof: Let $\mathcal{C}$ be a *compatible cell set* that contains any cell $\kappa_{i,j}$ and all of its compatible cells, *i.e.* $\mathcal{C} = \{\kappa_{i,j}\} \cup \{\kappa_{i',j'} | \kappa_{i',j'} \neq \kappa_{i,j}$, and $\kappa_{i',j'}$ is a compatible cell of $\kappa_{i,j}\}$. To make every cell in $\mathcal{C}$ can initiate a data transmission concurrently without interference[3], it is sufficient to have

$$\frac{P \cdot \|\kappa_{i,j} - \kappa'_{i,j}\|^{-\alpha}}{N_0 + \sum_{\kappa_{i',j'}\in\mathcal{C}, \kappa_{i',j'}\neq\kappa_{i,j}} P \cdot \|\kappa_{i',j'} - \kappa'_{i,j}\|^{-\alpha}} \geq \eta, \quad (1)$$

where $\kappa_{i,j}$ is any transmitting cell, $\kappa'_{i,j}$ is the receiving cell of $\kappa_{i,j}$, and $\kappa_{i',j'}$ is a compatible cell of $\kappa_{i,j}$ ($\kappa_{i',j'}$ is also transmitting some data packets simultaneously with $\kappa_{i,j}$).

[3]Here, we actually mean "to make every cell in $\mathcal{C}$ having a sensor can initiate a data transmission concurrently without interference". Without confusion, we use cell and sensor interchangeably.

Since P, N_0, and η are constant values, we derive the bounds of $\|\kappa_{i,j}-\kappa'_{i,j}\|^{-\alpha}$ and $\sum_{\kappa_{i',j'}\in\mathcal{C},\kappa_{i',j'}\neq\kappa_{i,j}}\|\kappa_{i',j'}-\kappa'_{i,j}\|^{-\alpha}$ in the following.

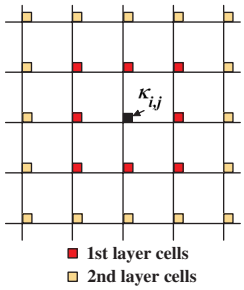

Figure 4. Computation of R.

First, we have $\|\kappa_{i,j}-\kappa'_{i,j}\|^{-\alpha}\geq r^{-\alpha}$ since r is the transmission radius of a sensor (defined in Section III.A) and every communication pair must within the communication range of each other. Subsequently, all the compatible cells of $\kappa_{i,j}$ in $\mathcal{C}$ can be layered with respect to $\kappa_{i,j}$ as shown in Figure 4, with the ψ-th ($\psi\geq 1$) layer having at most 8ψ cells[4]. Furthermore, the distance between the receiving cell of $\kappa_{i,j}$, *i.e.* $\kappa'_{i,j}$, and any compatible cell at the ψ-th layer is no less than $\psi\cdot R-2l$. Thus, we have

$$\sum_{\kappa_{i',j'}\in\mathcal{C},\kappa_{i',j'}\neq\kappa_{i,j}}\|\kappa_{i',j'}-\kappa'_{i,j}\|^{-\alpha} \tag{2}$$

$$\leq\sum_{\psi\geq 1}8\psi\cdot(\psi\cdot R-2l)^{-\alpha} \tag{3}$$

$$=8R^{-\alpha}\cdot\sum_{\psi\geq 1}\psi(\psi-\frac{2l}{R})^{-\alpha} \tag{4}$$

$$=8R^{-\alpha}\cdot[(1-\frac{2l}{R})^{-\alpha}+\sum_{\psi\geq 2}\psi(\psi-\frac{2l}{R})^{-\alpha}] \tag{5}$$

$$\leq 8R^{-\alpha}\cdot[(1-\frac{2}{3})^{-\alpha}+\sum_{\psi\geq 2}\psi(\psi-\frac{2}{3})^{-\alpha}] \tag{6}$$

$$\leq 8R^{-\alpha}\cdot[3^{\alpha}+\sum_{\psi\geq 2}\psi(\psi-1)^{-\alpha}] \tag{7}$$

$$=8R^{-\alpha}\cdot[3^{\alpha}+\sum_{\varpi\geq 1}\varpi^{-\alpha}(\varpi+1)] \tag{8}$$

$$=8R^{-\alpha}\cdot[3^{\alpha}+\sum_{\varpi\geq 1}\varpi^{-\alpha+1}+\sum_{\varpi\geq 1}\varpi^{-\alpha}] \tag{9}$$

$$\leq 8R^{-\alpha}\cdot[3^{\alpha}+\sum_{\varpi\geq 1}\varpi^{-2}+\sum_{\varpi\geq 1}\varpi^{-3}] \tag{10}$$

$$=8R^{-\alpha}\cdot(3^{\alpha}+\zeta(2)+\zeta(3)). \tag{11}$$

The reason from (4) to (5) is because $R\geq 3l$ (explained in Section III.C). The reason from (7) to (8) is because we use ϖ to substitute $\psi-1$, *i.e.* $\varpi=\psi-1$. The reason from (9) to (10) is because $\alpha\geq 3$. In (11), $\zeta(\cdot)$ is the *Riemann zeta function*, and with $\zeta(2)=\frac{\pi^2}{6}\approx 1.645$ and $\zeta(3)\approx 1.202$, respectively. Let $c_1=8(3^{\alpha}+2.847)$. We have

$$\sum_{\kappa_{i',j'}\in\mathcal{C},\kappa_{i',j'}\neq\kappa_{i,j}}\|\kappa_{i',j'}-\kappa'_{i,j}\|^{-\alpha}\leq c_1\cdot R^{-\alpha}. \tag{12}$$

It follows, we have

$$\frac{P\cdot\|\kappa_{i,j}-\kappa'_{i,j}\|^{-\alpha}}{N_0+\sum_{\kappa_{i',j'}\in\mathcal{C},\kappa_{i',j'}\neq\kappa_{i,j}}P\cdot\|\kappa_{i',j'}-\kappa'_{i,j}\|^{-\alpha}} \tag{13}$$

$$\geq\frac{P\cdot r^{-\alpha}}{N_0+c_1P\cdot R^{-\alpha}}. \tag{14}$$

Now, to make the inequality in (1) valid, it is sufficient to have

$$\frac{P\cdot r^{-\alpha}}{N_0+c_1P\cdot R^{-\alpha}}\geq\eta\Leftrightarrow R^{-\alpha}\leq\frac{r^{-\alpha}}{c_1\eta}-\frac{N_0}{c_1P} \tag{15}$$

$$\Leftrightarrow R\geq(\frac{r^{-\alpha}}{c_1\eta}-\frac{N_0}{c_1P})^{-1/\alpha} \tag{16}$$

Since N_0, α, c_1, η, and P are constant values, we have $R\geq(\frac{r^{-\alpha}}{c_1\eta}-\frac{N_0}{c_1P})^{-1/\alpha}\sim\Theta(r)$. Furthermore, considering that N_0 is negligible compared with the interference brought by concurrent transmitters [31], we can ignore N_0, *i.e.* let $N_0=0$. Thus, we have $R\geq\sqrt[\alpha]{c_1\eta}\cdot r=2\sqrt{2}\cdot\sqrt[\alpha]{c_1\eta}\cdot l$, since $r=2\sqrt{2}l$. Based on the definition of interference zones, a small R implies more compatible cells, which further implies more sensors can conduct transmissions concurrently, we set $R=2\sqrt{2}\cdot\sqrt[\alpha]{c_1\eta}\cdot l$, which is sufficient to make every cell in $\mathcal{C}$ can initiate a data transmission concurrently without interference. Now, let $\omega=2\sqrt{2}\cdot\sqrt[\alpha]{c_1\eta}$, *i.e.* $R=\omega\cdot l$. Evidently, ω is a constant value. It follows that we have proved Theorem 1. □

IV. Network Capacity of SDC

A. Cell-Based Path Scheduling (CBPS)

For each cell $\kappa_{i,j}$ in a WSN, we abstract it to a *super-node*, denoted by $\nu_{i,j}$. Note that $\nu_{i,j}$ may actually contain at most $8\log n$ sensors by Lemma 2, and we use this $\nu_{i,j}$ to represent the sensors in $\kappa_{i,j}$. We further define a *super time slot* $t_s=8\log n\cdot t$, which implies any super-node can transmit all its data to the next-hop, super-node or sink, within a super time slot t_s. Afterwards, we construct a *data collection tree* rooted at the sink to connect all the super-nodes according to the following rules:

- For super-node $\nu_{\lambda,j}(1\leq j\leq\lambda')$ (note that $\lambda'=\lambda-1$), $\nu_{\lambda,j}$ transmits its data to $\nu_{\lambda,j+1}$, *i.e.*, create a directed edge from $\nu_{\lambda,j}$ to $\nu_{\lambda,j+1}$;
- For super-node $\nu_{i,\lambda}(1\leq i\leq\lambda')$, $\nu_{i,\lambda}$ transmits its data to $\nu_{i+1,\lambda}$, *i.e.*, create a directed edge from $\nu_{i,\lambda}$ to $\nu_{i+1,\lambda}$;
- For super-node $\nu_{i,j}(1\leq i,j\leq\lambda')$, $\nu_{i,j}$ transmits its data to $\nu_{i+1,j+1}$, *i.e.*, create a directed edge from $\nu_{i,j}$ to $\nu_{i+1,j+1}$.

[4]This can be easily proven by *mathematical induction*.

The abstraction and data collection tree construction process are shown in Figure 5. In Figure 5, we also assign every path a name: p_1, the principal diagonal path; $p_i (2 \leq i \leq \lambda')$, the i-th path below p_1; $p'_i (2 \leq i \leq \lambda')$, the i-th path above p_1; p_v, which consists of super-nodes located at the same column with the sink; and p_h, which consists of super-nodes located at the same row with the sink. Furthermore, if the associated cells of some super-nodes are compatible cells, then these super-nodes are called *compatible nodes*. The paths that contain compatible nodes are called *compatible paths*, *e.g.* in Figure 5, p_1, p_4 and p_7 are compatible paths. From the analysis in Section III-C, we know that the compatible nodes on compatible paths can transmit data concurrently.

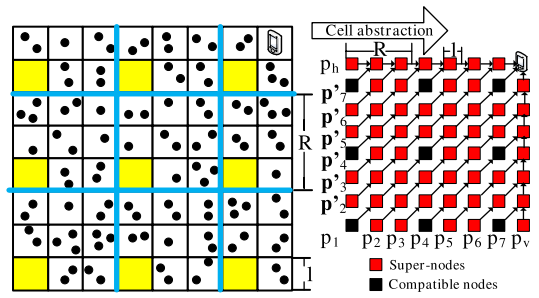

Figure 5. Construction of a data collection tree.

We now present the idea of our CBPS algorithm according to the example shown in Figure 5. CBPS has the following four steps.

Step 1: Schedule paths $p_1, p_2, \ldots, p_{\lambda'}$ until all the data packets on these paths have been transmitted to the super-nodes on p_v. When schedule $p_1, p_2, \ldots, p_{\lambda'}$, it is obvious that we can divide them into at most ω groups $G_k (0 \leq k \leq \omega - 1)$ with each group consisting of mutual compatible paths. Thereafter, in the i-th super time slot, we schedule paths in group $G_{(i-1)\%\omega}$. Taking the data collection tree shown in Figure 5 as an example, $p_1, p_2, \ldots, p_7$ can be divided into three groups with $G_0 = \{p_1, p_4, p_7\}$, $G_1 = \{p_2, p_5\}$ and $G_2 = \{p_3, p_6\}$. Thereafter, G_0, G_1 and G_2 will be scheduled in a round-robin fashion. Within a group, the super-nodes on all the paths can also be divided into at most ω node-groups $g_k (0 \leq k \leq \omega - 1)$ with each node-group containing mutual compatible nodes. Then, in the j-th available super time slot for a particular node-group, we schedule the super-nodes in $g_{(j-1)\%\omega}$. For group G_0 in the previous example, the super-nodes on paths p_1, p_4 and p_7 can be divided into three node-groups and they can be scheduled in a round-robin manner in the available super time slots for G_0.

Step 2: Schedule paths $p'_2, p'_3, \ldots, p'_{\lambda'}$ until all the data packets on these paths have been transmitted to the super-nodes on p_h. This step can be done in a similar way as in Step 1.

Step 3: Schedule path p_v until all the data packets have been transmitted to the sink. After Step 1, for any super-node $\nu_{\lambda,j} (1 \leq j \leq \lambda')$, it has the data of j super-nodes. Then, we abstract p_v to a *virtual tree* rooted at the sink, having λ' internal disjoint paths (except at the root) with lengths $1, 2, \ldots, \lambda'$ respectively by splitting super-node $\nu_{\lambda,j}$ into j *virtual nodes*. Now, in the virtual tree, every virtual node contains exactly the same data with a super-node as the result of the splitting. For instance, p_v in Figure 5 is abstracted to a virtual tree shown in Figure 6. Afterwards, we schedule each path of the resulting virtual tree by a similar path-scheduling method used in Step 1.

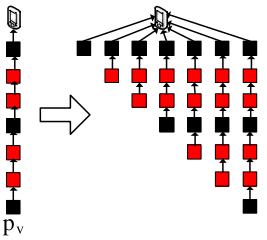

Figure 6. A virtual tree.

Step 4: Schedule path p_h until all the data packets have been transmitted to the sink. This step can be done in a similar way as in Step 3.

After the above four steps of CBPS, the data of a snapshot can be collected by the sink.

B. Network Capacity Analysis of CBPS

In this subsection, we derive the achievable network capacity of CBPS. The upper bound of the SDC capacity is W, which has been explained in [17]. Consequently, we focus on the lower bound of CBPS.

In each of the four steps of CBPS, the basic scheduling blocks are paths. Hence, we show the upper bound of the number of time slots used to schedule a path in the following lemma.

Lemma 3: For a path p (p consists of super-nodes) of length L, it takes $8\omega L \log n$ time slots to collect all the data packets on p by the sink.

Proof: From the description of CBPS, we know that the super-nodes (or virtual nodes) on p can be divided into at most ω compatible node-groups. In each available super time slot for p, we can schedule the super-nodes in a compatible node-group. Therefore, after ω super time slots, all the super-nodes on p have been scheduled once. Hence, all the super-nodes have transmitted their data to their corresponding parent super-nodes except for the sink (or the last end super-node of p), and all the super-nodes except for the leaf super-node have received the data from their corresponding children super-nodes. Therefore, after every ω super time slots, the data transmission path decreases by

one. Finally, the data on p can be collected by the sink within $\omega + (L-1) \cdot \omega$ super time slots, *i.e.* $8\omega L \log n$ time slots. □

From Lemma 3, we have the following corollary which shows the number of time slots used to collect data on p_1 (p_1 is the path corresponding to the cells on the principal diagonal).

Corollary 1: The data on p_1 can be collected to the cells on p_v (*i.e.*, the sink) within $8\omega\lambda' \log n$ time slots.

Lemma 4: Step 1 of CBPS can be finished within $8\omega^2\lambda' \log n$ time slots.

Proof: In Step 1, the paths in a compatible path group can be scheduled simultaneously, and the compatible path groups are scheduled in a round-robin fashion in terms of super time slots. Therefore, the number of time slots used in Step 1 depends on the compatible path group with the longest path. p_1 is the longest path and the compatible path group containing p_1 is scheduled every ω super time slots. Furthermore, by Corollary 1, the number of time slots used to collect the data on p_1 is at most $8\omega\lambda' \log n$. Hence, it follows that the number of time slots used to collect data on $p_1, p_2, \ldots, p_{\lambda'}$ is at most $8\omega^2\lambda' \log n$. □

From Lemma 4, we have the following corollary.

Corollary 2: Step 1 and Step 2 of CBPS can be finished within $16\omega^2\lambda' \log n$ time slots.

Lemma 5: Step 3 of CBPS can be finished within $4\omega\lambda'(\lambda'+1) \log n$ time slots.

Proof: According to CBPS, path p_v is abstracted into a virtual tree having λ' internally disjoint paths (except at the sink). Furthermore, the lengths of these λ' paths are $1, 2, \ldots, \lambda'$, respectively. By Lemma 3, the number of time slots used to collect data on these λ' paths is at most $8\omega \cdot 1 \cdot \log n + 8\omega \cdot 2 \cdot \log n + \ldots + 8\omega \cdot \lambda' \cdot \log n = 8\omega \cdot \log n \cdot (1 + 2 + \ldots + \lambda') = 4\omega\lambda'(\lambda'+1) \log n$. □

From Lemma 5, we have the following corollary.

Corollary 3: Step 3 and Step 4 of CBPS can be finished within $8\omega\lambda'(\lambda'+1) \log n$ time slots.

For the entire CBPS algorithm, the number of time slots is bounded by $O(n)$, which is proved in Theorem 2.

Theorem 2: The number of time slots used by CBPS to collect a snapshot data is bounded by $O(n)$.

Proof: Suppose T is the number of time slots used by CBPS to collect snapshot data by the sink. Then, by Corollary 2 and Corollary 3, we have $T \leq 16\omega^2\lambda' \log n + 8\omega\lambda'(\lambda'+1) \log n \leq 16\omega^2\lambda \log n + 8\omega\lambda^2 \log n = 16\omega^2 \cdot \frac{\sqrt{cn}}{\sqrt{2c \log n}} \cdot \log n + 8\omega \cdot (\frac{\sqrt{cn}}{\sqrt{2c \log n}})^2 \cdot \log n = 16\omega^2 \cdot \sqrt{\frac{n \log n}{2}} + 4\omega n \leq O(n)$. □

Now, we can obtain the lower bound of the achievable network capacity of CBPS which is order-optimal as shown in Theorem 3.

Theorem 3: The achievable network capacity of CBPS is $\Omega(W)$, which is order-optimal.

Proof: By Theorem 2 and the definition of network capacity, we have $C = \frac{n \cdot b}{\tau} \geq \frac{n \cdot b}{O(n) \cdot t} = \Omega(W)$. Since it has been proved that the upper bound of the data collection capacity is W, this implies that CBPS is order-optimal. □

When addressing the CDC problem, an intuitive idea is to pipeline the existing SDC algorithms. However, such an idea cannot improve the achievable network capacity in order as explained in [4]. This is because, data transmissions at the nodes far from the sink can really be accelerated by a pipeline. Nevertheless, the fact that a sink can receive at most one packet during each time slot makes the data accumulated at the nodes near the sink [4].

V. Network Capacity of CDC

Since CBPS and the existing works [2][17] with pipeline technology cannot improve the CDC capacity significantly as we discussed in Section IV, in this section, we propose a novel *Segment-Based Pipeline Scheduling* (SBPS) algorithm based on the the technology used in *Compressive Data Gathering* (CDG) [30] for CDC in WSNs. Theoretical analysis shows that the proposed SBPS algorithm can achieve a surprising network capacity.

A. Segment-Based Pipeline Scheduling (SBPS)

CDG was first proposed in [30] for snapshot data gathering in single-radio single-channel WSNs. The basic idea of CDG is to distribute the data collection load uniformly to all the nodes in the entire network. We take the data collection on a path consisting of L sensors $s_1, s_2, \ldots, s_L$ and one sink s_0 as shown in Figure 7 [30] as an example to explain CDG. In Figure 7, the packet produced at sensor $s_j (1 \leq j \leq L)$ is d_j. For the basic data collection shown in Figure 7(a), s_1 transmits one packet d_1 to s_2, s_2 transmits two packets d_1 and d_2 to s_3, and finally all the packets on the path are transmitted to s_0 by s_L. To balance the transmission load, the authors in [30] proposed the CDG method as shown in Figure 7(b). Instead of transmitting the original data directly, s_1 multiplies its data with a random coefficient $\phi_{i1} (1 \leq i \leq M)$, and sends the M results $\phi_{i1} d_1$ to s_2. Upon receiving $\phi_{i1} d_1 (1 \leq i \leq M)$ from s_1, s_2 multiplies its data d_2 with a random coefficient $\phi_{i2} (1 \leq i \leq M)$, adds it to $\phi_{i1} d_1$, and then sends $\phi_{i1} d_1 + \phi_{i2} d_2$ as one data packet to s_3. Finally, s_L does a similar multiplication and addition and sends the result $\sum_{j=1}^{L} \phi_{ij} d_j (1 \leq i \leq M)$ to s_0. After s_0 receives all M packets, s_0 can restore the original packets based on the compressive sampling theory [30]. The number of the transmitted packets is $O(n^2)$ in Figure 7 (a) and is $O(nM)$ in Figure 7 (b), and usually $M \ll n$ for large scale WSNs. Therefore, CDG reduces the number of transmitted packets.

Thanks to the benefit brought by CDG, we can address the CDC problem with the pipeline technique. Since we partition the network into interference zones $o_{i,j} (1 \leq i, j \leq \lceil \sqrt{cn}/R \rceil)$ in Section III-C, we here define a new term called *segment* based on interference zones. On the basis of interference zone $o_{i,i} (1 \leq i \leq \lceil \sqrt{cn}/R \rceil)$, the area

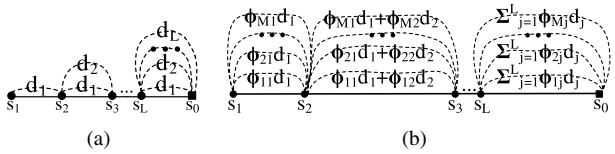

Figure 7. Comparison of (a) basic data collection and (b) CDG [30].

consisting of interference zones $o_{j,i}(i \leq j \leq \lceil\sqrt{cn}/R\rceil)$ and $o_{i,j}(i \leq j \leq \lceil\sqrt{cn}/R\rceil)$ is called a *segment*, denoted by S_i. Taking the network shown in Figure 8 as an example, there are three segments S_1 (consisting of interference zones $\{o_{1,1}, o_{2,1}, o_{3,1}, o_{1,2}, o_{1,3}\}$), S_2 (consisting of interference zones $\{o_{2,2}, o_{3,2}, o_{2,3}\}$), and S_3 (consisting of interference zone $\{o_{3,3}\}$). Within a segment S_i, the area consisting of cells on and below the principal diagonal is denoted by S_{ir}, and the area consisting of the remaining cells is denoted by S_{iu}, *i.e.*, $S_i = (S_{ir}, S_{iu})$. For instance, in the network shown in Figure 8, $S_{1r} = \{\kappa_{i,1}(1 \leq i \leq \lambda), \kappa_{i,2}(2 \leq i \leq \lambda), \kappa_{i,3}(3 \leq i \leq \lambda)\}$, and $S_{1u} = \{\kappa_{1,j}(1 < j \leq \lambda), \kappa_{2,j}(2 < j \leq \lambda), \kappa_{3,j}(3 < j \leq \lambda)\}$.

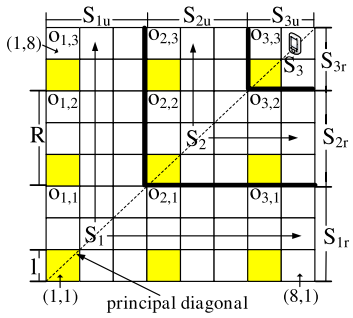

Figure 8. Segments.

For CDC, we use a similar routing structure as in the CBPS algorithm (note it does not imply the same scheduling), *i.e.*, we abstract each cell as a super-node and then construct a data collection tree following the same rules as in CBPS (we use cells and super-nodes interchangeably in the subsequent discussion). Unlike in CBPS, a super-node here can compress its currently held data packets of a snapshot into M data packets for transmission.

Based on the defined segments and the constructed data collection tree, we propose a *Segment-Based Pipeline Scheduling* (SBPS) algorithm for CDC. We present the general idea of SBPS in a hierarchy-level fashion as follows.

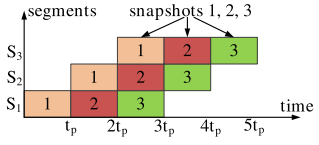

Figure 9. Data collection pipeline of 3 segments.

First, scheduling at the segment-level. At this level, each segment as a whole is considered. Since there is no intersection between any two segments, we can pipeline the data transmission on the segments (it can also be guaranteed that there is no wireless interference among segments in the next step), *i.e.*, for each segment $S_i = (S_{ir}, S_{iu})$, S_i starts the data transmission of the $(k+1)$-th snapshot immediately after it transmits all the data of the k-th snapshot to segment S_{i+1}. Let $t_p = \max\{t(S_i)|1 \leq i \leq \lceil\sqrt{cn}/R\rceil, t(S_i)$ *is the number of time slots used by segment* S_i *to transmit all the data packets of a snapshot*$\}$. Then, a segment data transmission pipeline on all the segments is formed with each segment working with t_p time slots for every snapshot (here, a snapshot is an individual task in a traditional pipeline operation). For instance, Figure 9 shows the data collection process of three snapshots by the segment data transmission pipeline formed from the network shown in Figure 8.

Second, scheduling at the row/column-level,* i.e., *within a segment. For the k-th snapshot, within each segment $S_i = (S_{ir}, S_{iu})$, we first schedule S_{ir} to transmit the data in the cells of S_{ir} to $S_{(i+1)r}$ row by row. Thereafter, we schedule S_{iu} by a similar way to transmit the data in the cells of S_{iu} to $S_{(i+1)u}$ column by column. When we schedule S_{ir}, the first row of cells of S_{ir}, *i.e.* the cells $\kappa_{j,i}(i \leq j \leq \lambda)$, are scheduled first, followed by the second row of cells, *i.e.* the cells $\kappa_{j,i+1}(i+1 \leq j \leq \lambda)$, and so on until the last row of cells of S_{ir}, *i.e.* the cells $\kappa_{j,i+\omega-1}(i+\omega-1 \leq j \leq \lambda)$, are scheduled. When we schedule each row, we can divide the cells on that row into ω compatible cell groups $g_1^i, g_2^i, \ldots, g_\omega^i$ with each group containing mutual compatible cells. Afterwards, $g_1^i, g_2^i, \ldots, g_\omega^i$ are scheduled in sequence. Note that we follow the same approach when we schedule all the segments in the segment data transmission pipeline. Therefore, all the cells in $g_j^i(1 \leq j \leq \omega, 1 \leq i \leq \lceil\sqrt{cn}/R\rceil)$ are also mutual compatible cells according to the discussion in Section III-C. This implies all the segments can be scheduled without wireless interference. Afterwards, we can schedule cells in S_{iu} column by column in a similar way. Finally, S_i transmits all the data packets of the k-th snapshot to its subsequent segment S_{i+1}.

Third, scheduling at the cell-level,* i.e. *within each row/column. For every sensor in cell $\kappa_{i,j}$, it generates one data packet of the k-th snapshot. Furthermore, for the sensors in cells $\kappa_{i,1}(1 \leq i \leq \lambda)$ and $\kappa_{1,j}(1 \leq j \leq \lambda)$, they will not receive any data packets of the k-th snapshot according to the previous segment-level and row/column-level scheduling strategies (actually, this is true for any snapshot). Thus, the sensors in $\kappa_{i,1}(1 \leq i \leq \lambda)$ and $\kappa_{1,j}(1 \leq j \leq \lambda)$ transmit the packets of the k-th snapshot in the CDG way in their available time slots, *i.e.*, for each sensor, it multiplies its data with M random coefficients respectively, and sends the new obtained M products to its parent node. For the sensors in $\kappa_{i,j}(1 < i, j \leq \lambda)$, they will receive some data packets of the k-th snapshot (it is possible that some sensors do not

have any children. In this case, they do the same operation as the sensors in $\kappa_{i,1}(1 \leq i \leq \lambda)$ and $\kappa_{1,j}(1 \leq j \leq \lambda)$.). After they receive all the packets of the k-th snapshot from their children sensors, they combine their data and the received data in the same way as in CDG and transmit the obtained M data packets to their parent sensors, respectively. For the sink, it restores the data of a snapshot in the CDG way after it receives all the packets of that snapshot. Here, it is straightforward that it takes at most $8M \cdot \log n$ time slots for a cell to transmit the data packets of a snapshot to the subsequent cell, since every cell contains at most $8 \log n$ sensors by Lemma 2.

B. Network Capacity Analysis of SBPS

In this subsection, we analyze the achievable network capacity of SBPS. Since the t_p in the SBPS algorithm is essential for our analysis, we give the upper bound of t_p in the following lemma.

Lemma 6: For the t_p in SBPS, $t_p \leq 16\omega^2 M \log n$.

Proof: It has been pointed out in Section V-A that it takes a cell at most $8M \cdot \log n$ time slots to transmit the data of a particular snapshot. Furthermore, when we schedule each row of a segment $S_i = (S_{ir}, S_{iu})$, we divide the cells of that row into ω compatible cell groups and schedule these groups in sequence. Therefore, the data packets of a particular snapshot contained in the cells of a row can be transmitted to the subsequent row within $8\omega M \log n$ time slots. Each segment has at most ω rows. Therefore, the number of time slots used to schedule cells in S_{ir} is at most $8\omega^2 M \log n$. For the same reason, the number of time slots used to schedule the cells in S_{iu} for a particular snapshot is also at most $8\omega^2 M \log n$. In a sum, $t_p \leq 16\omega^2 M \log n$. □

Based on Lemma 6, we obtain the upper bound of the number of time slots used by SBPS to collect N continuous snapshots as follows.

Theorem 4: The number of time slots used by the SBPS algorithm to collect N continuous snapshots is at most $8\omega M\sqrt{2n \log n} + 16\omega^2 MN \log n$.

Proof: Suppose the number of time slots used by SBPS to collect N continuous snapshots is T. By Lemma 6, it costs a segment $16\omega^2 M \log n$ time slots to transmit the data of a snapshot to the subsequent segment. Afterwards, that segment starts to transmit data for the following snapshot immediately according to SBPS. Therefore, by the formed segment data transmission pipeline, the sink can collect the data of a snapshot in every $16\omega^2 M \log n$ time slots after it receives the data of the first snapshot. Thus, to receive the data of N continuous snapshots, we have $T \leq 16\omega^2 M \log n \cdot \lceil \frac{\lambda}{\omega} \rceil + (N-1) \cdot 16\omega^2 M \log n \leq 16\omega M \lambda \log n + 16\omega^2 MN \log n = 16\omega M \sqrt{\frac{cn}{2c \log n}} \log n + 16\omega^2 MN \log n = 8\omega M\sqrt{2n \log n} + 16\omega^2 MN \log n$. □

Now, we are prepared to derive the achievable network capacity of SBPS for CDC as shown in Theorem 5.

Theorem 5: The achievable network capacity of the SBPS algorithm is $\Omega(\sqrt{\frac{n}{\log n}}W)$ when $N \leq \sqrt{\frac{n}{\log n}}$; or $\Omega(\frac{n}{\log n}W)$ when $N > \sqrt{\frac{n}{\log n}}$.

Proof: By Theorem 4, it takes the SBPS algorithm at most $8\omega M\sqrt{2n \log n} + 16\omega^2 MN \log n$ time slots to collect N continuous snapshots by the sink. Then, we discuss the achievable network capacity of SBPS case by case.

case 1: $N \leq \sqrt{\frac{n}{\log n}}$. In this case, $T \leq 8\omega M\sqrt{2n \log n} + 16\omega^2 MN \log n \leq O(8\omega M\sqrt{2n \log n})$. Thus, we have

$$C = \frac{nN \cdot b}{\tau} \geq \frac{nN \cdot b}{O(8\omega M\sqrt{2n \log n}) \cdot t} = \Omega(\sqrt{\frac{n}{\log n}}W),$$

since ω is a constant value and $M \ll n$.

case 2: $N > \sqrt{\frac{n}{\log n}}$. In this case, $T \leq 8\omega M\sqrt{2n \log n} + 16\omega^2 MN \log n \leq O(16\omega^2 MN \log n)$. Thus, we have

$$C = \frac{nN \cdot b}{\tau} \geq \frac{nN \cdot b}{O(16\omega^2 MN \log n) \cdot t} = \Omega(\frac{n}{\log n}W).$$

□

From Theorem 5, the proposed scheduling algorithm SBPS can achieve a surprisingly high network capacity by combining the pipeline and CDG techniques. Since the current best result is $\Omega(W)$ [17], our result is at least $\sqrt{\frac{n}{\log n}}$ or $\frac{n}{\log n}$ times better than the current best result, which is a very significant improvement.

VI. Conclusion and Future Work

Most existing works focus on network capacity of *unicast*, *multicast* or/and *broadcast*, which are different communication modes from data collection, especially CDC. In this paper, we first study the SDC problem under the PhIM, and propose a *Cell-Based Path Scheduling* (CBPS) algorithm based on network partitions. Theoretical analysis of CBPS shows that its achievable network capacity is $\Omega(W)$, which is order-optimal. For CDC, we propose a novel *Segment-Based Pipeline Scheduling* (SBPS) algorithm. SBPS significantly speeds up the CDC process, and achieves a surprising network capacity, which is at least $\sqrt{\frac{n}{\log n}}$ or $\frac{n}{\log n}$ times better than the current best result. Furthermore, the numerical evaluation results also validate that the proposed algorithms significantly improve network capacity compared with existing works.

The future work of this paper can be conducted according to the following directions. First, instead of assuming all the nodes are randomly deployed, we will study the achievable capacity of WSNs where the nodes are arbitrarily deployed. Second, since most of the existed works that study the network capacity issue are for centralized WSNs, we will investigate the achievable data collection capacity of distributed WSNs.

ACKNOWLEDGMENT

This work is partly supported by the NSF under grant No. CCF-054566.

REFERENCES

[1] P.-J. Wan, S. C.-H. Huang, L. Wang, Z. Wan and X. Jia, Minimum-Latency Aggregation Scheduling in Multihop Wireless Networks, Mobihoc 2009.

[2] S. Chen, S. Tang, M. Huang, and Y. Wang, Capacity of Data Collection in Arbitrary Wireless Sensor Networks, Infocom 2010.

[3] S. Chen, Y. Wang, X.-Y. Li, and X. Shi, Order-Optimal Data Collection in Wireless Sensor Networks: Delay and Capacity, Secon 2009.

[4] S. Ji, Y. Li, and X. Jia, Capacity of Dual-Radio Multi-Channel Wireless Sensor Networks for Continuous Data Collection, Infocom 2011.

[5] B. Liu, D. Towsley, and A. Swami, Data Gathering Capacity of Large Scale Multihop Wireless Networks, MASS 2008.

[6] O. Chipara, C. Lu, and J. Stankovic, Dynamic Conflict-free Query Scheduling for Wireless Sensor Networks, ICNP 2006.

[7] E. J. Duarte-Melo and M. Liu, Data-Gathering Wireless Sensor Networks: Organization and Capacity, Computer Networks, 43: 519-537, 2003.

[8] D. Marco, E. J. Duarte-Melo, M. Liu, and D. L. Neuhoff, On the Many-to-One Transport Capacity of a Dense Wireless Sensor Network and the Compressibility of Its Data, IPSN 2003.

[9] X.-Y. Li, S. Tang and O. Frieder, Multicast Capacity for Large Scale Wireless Ad Hoc Networks, Mobicom 2007.

[10] Z. Wang, H. R. Sadjadpour and J. J. Garcia-Luna-Aceves, A Unifying Perspective on the Capacity of Wireless Ad Hoc Networks, Infocom 2008.

[11] M. Garetto, P. Giaccone and E. Leonardi, On the Capacity of Ad Hoc Wireless Networks Under General Node Mobility, Infocom 2007.

[12] G. Sharma, R. Mazumdar and N. B. Shroff, Delay and Capacity Trade-Offs in Mobile Ad Hoc Networks: A Global Perspective, IEEE/ACM ToN, 15(5): 981-992, 2007.

[13] W. Huang, X. Wang, Q. Zhang, Capacity Scaling in Mobile Wireless Ad Hoc Network with Infrastructure Support, ICDCS 2010.

[14] V. Bhandari and N. H. Vaidya, Connectivity and Capacity of Multi-Channel Wireless Networks with Channel Switching Constraints, Infocom 2007.

[15] 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, Infocom 2008.

[16] G. Zhang, Y. Xu, X. Wang, M. Guizani, Capacity of Hybrid Wireless Networks with Directional Antenna and Delay Constraint, IEEE Tran. on Comm., 58(7): 2097-2106, 2010.

[17] S. Chen, Y. Wang, X.-Y. Li, and X. Shi, Data Collection Capacity of Random-Deployed Wireless Sensor Networks, Globecom 2009.

[18] O. Goussevskaia, R. Wattenhofer, M. M. Halldorsson and E. Welzl, Capacity of Arbitrary Wireless Networks, Infocom 2009.

[19] M. Andrews and M. Dinitz, Maximizing Capacity in Arbitrary Wireless Networks in the SINR Model: Complexity and Game Theory, Infocom 2009.

[20] S. Li, Y. Liu and X.-Y. Li, Capacity of Large Scale Wireless Networks Under Gaussian Channel Model, Mobicom 2008.

[21] H. E. Gamal, On the Scaling Laws of Dense Wireless Sensor Networks, IEEE Tran. on Inf. Theo., 51(3): 1229-1234, 2003.

[22] M. Franceschetti, O. Dousse, D. N. C. Tse, and P. Thiran, Closing the Gap in the Capacity of Wireless Networks Via Percolation Theory, IEEE Tran. on Inf. Theo., 53(3): 1009-1018, 2007.

[23] U. Niesen, P. Gupta and D. Shah, The Balanced Unicast and Multicast Capacity Regions of Large Wireless Networks, IEEE Tran. on Inf. Theo., 56(5): 2249-2271, 2010.

[24] X. Wang, Y. Bei, Q. Peng, L. Fu, Speed Improves Delay-Capacity Tradeoff in MotionCast, IEEE TPDS, 2011.

[25] A. Keshavarz-Haddad, V. Ribeiro, and R. Riedi, Broadcast Capacity in Multihop Wireless Networks, Mobicom 2006.

[26] 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, MASS 2008.

[27] B. Sirkeci-Mergen and M. Gastpar, On the Broadcast Capacity of Wireless Networks, ITAW 2007.

[28] P. Gupta and P. R. Kumar, The Capacity of Wireless networks, IEEE Trans. Inf. Theo., 46(2): 388-404, 2000.

[29] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, Algorithmic Aspects of Capacity in Wireless Networks, Sigmetrics 2005.

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

[31] C.-K. Chau, M. Chen and S. C. Liew, Capacity of Large-Scale CSMA wireless Networks, Mobicom 2009.

[32] Y. Xu and W. Wang, Scheduling Partition for Order Optimal Capacity in Large-scale Wireless Networks, Mobicom 2009.

[33] S. R. Kulkarni and P. Viswanath, A Deterministic Approach to Throughput Scaling in Wireless Networks, IEEE Tran. on Inf. Theo., 50(6):1041-1049, 2004.