# Efficient Data Aggregation in Multi-hop Wireless Sensor Networks Under Physical Interference Model

Xiang-Yang Li[†,⋆,‡], XiaoHua Xu[⋆], ShiGuang Wang[⋆], ShaoJie Tang[⋆], GuoJun Dai[†], JiZhong Zhao[‡], Yong Qi[‡]
[†]Institute of Computer Application Tech., Hangzhou Dianzi Univ., Hangzhou, PRC. {xli,daigj}@hdu.edu.cn
[⋆]Dept. of Computer Science, Illinois Inst. of Tech., Chicago, USA, {xxu23,swang44,stang7}@iit.edu
[‡]School of Science and Technology, Xi'an JiaoTong University, Xi'an, China. {zjz,qiy}@mail.xjtu.edu.cn

***Abstract*—Efficient aggregation of data collected by sensors is crucial for a successful application of wireless sensor networks (WSNs). Both minimizing the energy cost and reducing the time duration (or called latency) of data aggregation have been extensively studied for WSNs. Algorithms with theoretical performance guarantees are only known under the protocol interference model, or graph-based interference models generally. In this paper, we study the problem of designing time efficient aggregation algorithm under the physical interference model. To the best of our knowledge, no algorithms with theoretical performance guarantees are known for this problem in the literature. We propose an efficient algorithm that produces a data aggregation tree and a collision-free aggregation schedule. We theoretically prove that the latency of our aggregation schedule is bounded by $O(R+\Delta)$ time-slots. Here $R$ is the network radius and $\Delta$ is the maximum node degree in the communication graph of the original network. In addition, we derive the lower-bound of latency for any aggregation scheduling algorithm under the physical interference model. We show that the latency achieved by our algorithm asymptotically matches the lower-bound for random wireless networks. Our extensive simulation results corroborate our theoretical analysis.**

***Index Terms*—Wireless sensor networks, aggregation, scheduling, physical interference model.**

## I. Introduction

One practical issue in wireless sensor networks (WSNs) is efficient data processing. As we know, in WSN, data are generated everywhere, such as temperature reporting and so on. In most scenarios, we assume there is a distinguished sink sensor that has more computing ability than other sensors. Thus we only need to send all data from sensor nodes within network to the sink sensor which is termed as data collection or data aggregation. Different from data collection, data aggregation allows in-network processing. This means that data can be compressed within the network. This feature introduces a possibility of a new energy or time efficient method to collect data, comparing to data collection. For data aggregation, different objectives will be emphasized, depending on the applications, such as to minimize the energy consumption, to minimize the latency, to increase the accuracy of the reported data, and so on.

In this paper, we concentrate on minimizing latency for data aggregation. It can be roughly defined as follows: given a set of sensor nodes distributed in a two-dimensional Euclidean plane, the objective is to compute an aggregation function (which will be defined later) on the input data from all sensors within the networks. Data aggregation under the protocol interference model has been extensively studied recently [1], [6], [12], [16], [23], [25]. Protocol interference model is generally a graph-based interference model, under which, given a network $G$, there is a conflict graph $H$ such that two vertices in $H$ cannot be activated simultaneously. Here vertices of $H$ could be all wireless nodes of $G$, or all wireless links of $G$, depending on the interference model. For minimizing the latency of data aggregation, a class of constant-ratio approximation algorithms have been given under various protocol interference models captured by some conflict graphs. When the conflict graph $H$ is a unit disk graph, Li *et al.* proposed an algorithm [2] that achieves latency at most $24D + 6\Delta + 16$ for a network of diameter $D$ and maximum node degree $\Delta$. This was recently improved to $16R + \Delta$ model by Xu *et al.* [27] and $\left(1 + (\log R/\sqrt[3]{R})\right) R + \Delta$ by Wan *et al.* [25] under the same network model.

However, protocol interference model and other graph-based interference models can not reflect some key features in real WSNs. They are only approximate interference models. It is a folklore that physical interference model captures the interferences between links more accurately. Surprisingly, few previous literatures have studied data aggregation under this model. This may be due to challenges in handling physical interferences in which we have to take care of the effect of aggregated interferences from nodes that are far away. The potential interference effect from far-away nodes makes it difficult to design a scheduling method to ensure the SINR (Signal to Interference-plus-Noise Ratio) of every receiver node is always above the threshold.

To the best of our knowledge, we are the first to study the problem of data aggregation under the physical interference model. Our main contributions are as follows. Assume that we are given a network $G$ consisting of $n$ wireless nodes $V$. Every node in $V$ will transmit with a constant power $P$ and the power received by a node at distance $d$ is assumed to be $P \cdot \min(1, d^{-\alpha})$. The variance of the background noise is assumed to be $N_0 > 0$ and for a successful transmission, the received SINR is required to be above a threshold value $\beta$. To ensure that we can perform scheduling using some local information, we will focus only on links that are not too long. Specifically, let $\mathbf{r} = (\frac{\beta N_0}{P})^{-\alpha}$ be the maximum length of a link which can transmit alone successfully. For the set $V$ of wireless nodes, we will only consider all links $(u, v)$ with Euclidean length at most $\delta\mathbf{r}$, where $0 < \delta < 1$ is a small constant. The network formed by these links is denoted as $G(V, \delta\mathbf{r})$. We first present an algorithm for data aggregation

978-1-4244-5113-5/09/$25.00 

scheduling for the network $G(V, \delta \mathbf{r})$ under the physical interference model. We analytically proved that our algorithm can achieve a constant approximation ratio on the latency of data aggregation where the optimum is also computed using network $G(V, \delta \mathbf{r})$. We then present the latency lower-bound of any algorithm under this network model. Notice that using links longer than $\delta \mathbf{r}$, we may be able to reduce the latency of data aggregation (it is an open question at the current stage whether we can reduce the latency by more than a constant factor). Fortunately, we are able to prove that our method is asymptotically optimum for random wireless networks where nodes are uniformly and randomly distributed in a square region. We prove that, using only $G(V, \delta \mathbf{r})$, a subgraph of the original communication graph $G(V, \mathbf{r})$, our method will achieve a latency that is within a constant factor of the optimum latency achievable using all links from $G(V, \mathbf{r})$ for random WSN. On the negative side, we show that, given any constant $\delta < 1$, there is an example of $n$ nodes $V$ such that *any* algorithm for data aggregation scheduling using only links in $G(V, \delta \mathbf{r})$ will have latency that is at least $n$, while the optimum latency using all links in $G(V, \mathbf{r})$ is only $O(\sqrt{n})$. Our extensive simulation studies show that our method performs well in practice. To further reduce the latency, we also adopt several strategies to compress the scheduling by possibly merging the scheduled links in previous time-slots. We found that this compressive scheduling will almost halve the latency achieved by our first algorithm for sparse networks (with maximum node degree $\Delta \leq 25$, which is always true in practice).

The rest of the paper is organized as follows. Section II formulates the data aggregation problem. Section III presents our scheduling algorithm and Section IV analyzes its performances. Section V discusses the overall lower-bound under our model. Furthermore, Section VI provides the analytical results in randomly deployed networks. Section VII presents the simulation results. Section VIII outlines the related work. Section IX concludes the paper.

## II. System Models

In this section, we describe the network model that we will use and some related terminologies.

### A. Network Model

Consider a WSN consisting of $n$ nodes $V$ where $v_s \in V$ is the sink node. Each node can send (receive) data to (from) all directions. The objective is to find a "valid" data aggregation schedule such that after using the schedule, all data can be aggregated to the sink node. Here "valid" means that, in this schedule, a node can receive data successfully if and only if its Signal to Interference-plus-Noise Ratio (SINR) is greater than some threshold. In other words, we consider data aggregation scheduling under the physical interference model. We define physical interference model formally as follows.

Physical Interference Model: Assume all nodes have fixed transmission power $P$. We define $P_v(u) = P \cdot g(u, v)$ as the received power at the receiver node $v$ of the signal transmitted by node $u$, Here $g(u, v) \leq 1$ is called the path-gain from node $u$ to node $v$. A receiver node $v$ can successfully receive a packet from a sender node $u$ if and only if:

$$\frac{P_v(u)}{N_0 + \sum_{w \in S_u} P_v(w)} \geq \beta \tag{1}$$

Here $\beta > 0$ denotes the SINR threshold, $N_0 \geq 0$ denotes the ambient noise, and $S_u$ denotes the set of other simultaneous senders with sender $u$. Note that $S_u$ is not necessarily the same for all time-slots.

In our paper, we set path-gain as $g(u, v) = \min(1, \|uv\|^{-\alpha})$, where the constant $\alpha \geq 2$ is the path-gain exponent, and $\|uv\|$ is the Euclidean distance between node $u$ and $v$.

Communication Graph: By the above definition of physical interference model, we can compute the maximum link length as $\mathbf{r} = (\frac{P}{N_0 \beta})^{-\frac{1}{\alpha}}$, *i.e.*, any node $u$ cannot transmit a packet to a node $v$ that is more than the distance $\mathbf{r}$ away successfully even there is no other simultaneous transmission. In other words, a pair of nodes can possibly communicate and thus be connected if and only if their mutual distance is smaller or equal to $\mathbf{r}$. If we draw a link between every pair of nodes whose distance is smaller than $\mathbf{r}$, we can derive a communication graph (denote as $G(V, \mathbf{r})$). Generally in this paper, given a set of nodes $V$, we use $G(V, r)$ to denote the graph that has all edges $(u, v)$ whose Euclidean length is at most $r$.

As we know, the longer the link, the smaller the path-gain, thus the smaller the SINR can be obtained by its corresponding receiver. Therefore, a long link with length comparatively close to $\mathbf{r}$ is not a good candidate in practice for transmission since the SINR at the receiver is very small. Even worse, it prevents many possible simultaneous transmissions. In brief, a shorter link is a better candidate for the transmission. Thus, in this paper we assume that we will only use some links that are smaller than $\mathbf{r}$. Specifically, we assume that there is a constant factor $\delta \in (0, 1)$ and we consider only the links shorter or equal to $\delta \mathbf{r}$. We call these short links as strong connected links. Then we can construct a strong connected communication graph (denote as $G(V, \delta \mathbf{r})$) with only these strong connected links. In the strong connected communication graph, only the nodes which lie within $\delta \mathbf{r}$ of $u$ can receive the data transmitted by $u$ successfully. Note that this requirement is not necessary. However it allows multiple simultaneous transmissions for data communications. Next, we consider data aggregation scheduling in this new communication graph $G(V, \delta \mathbf{r})$.

Data Aggregation Scheduling: For simplicity, we only consider data aggregation scheduling in a synchronous message passing model in which time is divided into slots. In each time-slot, a node $v \in V$ is able to send a message to one of its neighboring nodes for unicast communication. Note that, at the cost of higher communication cost, our scheduling can also be implemented in asynchronous communication settings using the notions of synchronizer. The data aggregation scheduling is to define transmission time-slot(s) for every node $v_i \in V$ such that with a minimum latency (measured in time-slots), the sink node will be able to get the final correct aggregation results.

Consider a time-slot $t$, and let $A$ be the set of senders and $B$ be the receivers. Assume $A, B \subset V$ and $A \cap B = \emptyset$. Then data

are aggregated from $A$ to $B$ in communication graph $G(V, \delta\mathbf{r})$ in one time-slot if:

1) all the transmission links exist in $G(V, \delta\mathbf{r})$;
2) all the nodes in $A$ merge their data and the receiving data if any using *an aggregation function* and then transmit data in one time-slot simultaneously;
3) all data are received by some nodes in $B$ with the SINR at each receiving node greater or equal to $\beta$.

Here aggregation functions will be defined in Section II-B. Then a valid aggregation schedule in $G(V, \delta\mathbf{r})$ with latency $L$ can be defined as a sequence of sender sets $S_1, S_2, \cdots, S_L$ satisfying the following conditions:

1) $S_i \cap S_j = \emptyset$, $\forall i \neq j$;
2) $\cup_{i=1}^{L} S_i = V \setminus \{v_s\}$;
3) Data are aggregated from $S_k$ to $V \setminus \cup_{i=1}^{k} S_i$ in $G(V, \delta\mathbf{r})$ at time-slot $k$, for all $k = 1, 2, \cdots, L$ and all data are aggregated to the sink node $v_s$ in $L$ time-slots.

Notice that here $\cup_{i=1}^{L} S_i = V \setminus \{v_s\}$ is to ensure that all data will be aggregated; $S_i \cap S_j = \emptyset$ ($\forall i \neq j$) is to ensure that every datum is aggregated at most once. To simplify our analysis, we will relax the requirement that $S_i \cap S_j = \emptyset$, $\forall i \neq j$. When the sets $S_i$, $1 \leq i \leq L$ are not disjoint, in the actual data aggregation, a node $v$, that appears multiple times in $S_i$, $1 \leq i \leq L$, will participate in the data aggregation only once (say the smallest $i$ when it appears in $S_i$), and then it will only serve as a relay node in later appearances.

We will find a valid aggregation schedule $\{S_1, S_2, \cdots, S_L\}$ in the graph $G(V, \delta\mathbf{r})$ with minimum latency $L$. Here, we assume the network $G(V, \delta\mathbf{r})$ is connected. This assumption is necessary, otherwise the disconnected nodes cannot send their data to the sink node via the network. For simplicity, we define $\Delta(G), D(G), R(G)$ as the maximum degree, network diameter, network radius of graph $G(V, \delta\mathbf{r})$ respectively.

### B. Related Terminologies

AGGREGATION FUNCTIONS: Data aggregation functions can be classified into three categories: distributive (*e.g., maximum, minimum, sum, count*), algebraic (*e.g., minus, average, variance*) and holistic (*e.g., median, $k^{th}$ smallest or largest*). Here we only focus on the distributive or algebraic aggregation functions. A aggregation function $f$ is said to be *distributive* if for every pair of disjoint data sets $X_1, X_2$, we have $f(X_1 \cup X_2) = h(f(X_1), f(X_2))$ for some function $h$. For example, when $f$ is *sum*, then $h$ can be set as *sum*; when $f$ is *count*, $h$ is *sum* as well. Given a distributive aggregation function $f$, it can be expressed as a combination of $k$ distributive functions for some integer constant $k$, *i.e.*,

$$f(X) = h(g_1(X), g_2(X), ..., g_k(X)).$$

For example, when $f$ is *average*, then $k = 2$, $g_1(X) = \sum_{x_i \in X} x_i$, $g_2(X) = \sum_{x_i \in X} 1$, (obviously both $g_1$ and $g_2$ are distributive) and $h(g_1, g_2) = g_1/g_2$. When $f$ is *variance*, then $k = 3$, $g_1(X) = \sum_{x_i \in X} x_i^2$, $g_2(X) = \sum_{x_i \in X} x_i$, $g_3(X) = \sum_{x_i \in X} 1$, and $h(g_1, g_2, g_3) = g_1 - \frac{g_2^2}{g_3}$. Hereafter, we assume that an algebraic function $f$ is given in formula $h(g_1, g_2, \cdots, g_k)$. Thus, instead of computing $f$, we just compute $g_i(X)$ distributively for $i \in [1, k]$ and $h(g_1, g_2, \cdots, g_k)$ at the sink node.

CONNECTED DOMINATING SET: In a graph $G = (V, E)$, a subset $V_0$ of $V$ is a dominating set (DS) if each node in $V$ is either in $V_0$ or adjacent to some node in $V_0$. Nodes in $V_0$ are called dominators, whereas nodes not in $V_0$ are called dominatees. A subset $C$ of $V$ is a connected dominating set (CDS) if $C$ is a dominating set and $C$ induces a connected subgraph. Consequently, the nodes in $C$ can communicate with each other without using nodes in $V \setminus C$.

## III. AGGREGATION SCHEDULING

In this section we first present an algorithm to construct a data aggregation tree. Then based on this tree, we design a schedule of links' transmissions to approximately minimize the latency of the aggregation.

### A. Aggregation Tree Construction

We construct the aggregation tree on the communication graph $G(V, \delta\mathbf{r})$ using Algorithm 1. The basic idea of Algorithm 1 is to construct a tree similar to the breadth-first-search tree, with the following properties: (1) the depth of the tree is within a small constant factor of the diameter $D(G)$, (2) each internal node will be connected with at most a constant number of other internal nodes. The second property ensures that we can schedule the transmissions of internal nodes in constant time-slots. Our algorithm is similar to [21] with two crucial modifications:

1) We select the *topology center* of $G(V, \delta\mathbf{r})$ as the root of our BFS tree. Notice that, previous methods use the sink node as the root. Our topology center selection enables us to reduce the latency to a function of the network radius $R(G)$, instead of the network diameter $D(G)$ proved by previous methods. Here a node $v_0$ is called the *topology center* in a graph $G$ if $v_0 = \arg\min_v \{\max_u d_G(u, v)\}$, where $d_G(u, v)$ is the hop distance between nodes $u$ and $v$ in graph $G$. Notice that in most networks, the topology center is different from the sink node.
2) After the topology center gathered the aggregated data from all nodes, it will then send the aggregated result to the sink node via the shortest path from the topology center $v_0$ to the sink node $v_s$. This will incur an additional latency $d_G(v_0, v_s)$ of at most $R(G)$.

**Algorithm 1** Aggregation Tree Construction

1: select the topology center $v_0$ of $G(V, \delta\mathbf{r})$;
2: using $v_0$ as the root, perform BFS over $G(V, \delta\mathbf{r})$ to build the BFS tree $T_G$; we denote the height of $T_G$ as $R = R(G)$, the *radius* of the network $G(V, \delta\mathbf{r})$.
3: select the MIS of $T_G$ by an existing approach [21]. We call all nodes in MIS as dominators and all the other nodes in $T_G$ as dominatees;
4: connect MIS using some nodes (we call them as connectors) to form a CDS of the graph $G(V, \delta\mathbf{r})$ (note that connectors are some dominatees originally);
5: output the CDS $\mathbf{G_c}$ as the backbone of the aggregation tree with $v_0$ as the root.

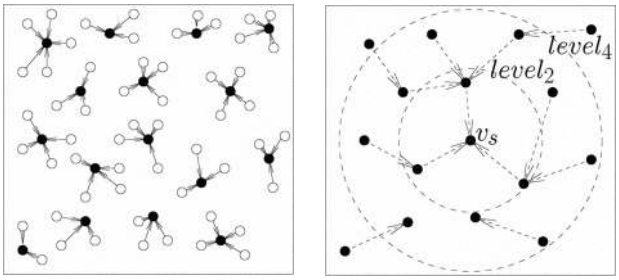
Observe that here the CDS $\mathbf{G_c}$ does not contain all nodes $V$ yet: some dominatees are not in $\mathbf{G_c}$. For each dominatee

$v$ not in $\mathbf{G_c}$, we will connect it to the neighboring dominator that has the smallest hop-distance to the topology center $v_0$. The tree formed by these additional links is called the final data aggregation tree $T$. For the constructed aggregation tree $T$, we can prove that the depth of $T$ is at most $2R(G)$, where $R(G)$ is the radius of the network $G(V, \delta\mathbf{r})$. Additionally, each node $v \in \mathbf{G_c}$ is connected to at most 19 nodes in $\mathbf{G_c}$.

### B. Our solution

We then propose our aggregation scheduling algorithm based on the aggregation tree constructed in Algorithm 1.

Assume that each wireless node already knows its geometric locations. A number of methods have been proposed in the literature to approximate the geometric locations of nodes. We partition the deployment plane into grids by using a set of vertical lines $a_v : x = v \cdot \ell$ $(v \in \mathbb{Z})$ and horizontal lines $b_h : y = h \cdot \ell$ $(h \in \mathbb{Z})$. Hereafter $v$ $(h)$ is called the index of the vertical (horizontal) line $a_v$ $(b_h)$. We call the cell formed by a pair of neighboring vertical lines $a_v, a_{v+1}$ and neighboring horizontal lines $b_h, b_{h+1}$ as $g_{v,h}$. Then we color the cells (shown in Algorithm 2 and Algorithm 3) with the specification that any pair of cells that have the same color are far away enough for simultaneously transmitting. Then, for any cell, at any time-slot, we will only choose at most one node from this cell to transmit. In Theorem 1, we will formally give the closest distance between any two cells with the same color in terms of the number of cells between them. And we prove that this distance guarantees the simultaneous transmission.




(a) Phase I (b) Phase II

Fig. 1. The process of Algorithm 4: the black nodes are dominators and white nodes are dominatees.

Then, our aggregation scheduling algorithm can be divided into two phases:

**Phase I:** every dominator aggregates the data from all its dominatees (as described in line $3-9$ of Algorithm 4 and also shown in Figure 1(a));

**Phase II:** dominators aggregate their data to the sink node $v_s$ level by level (as described in line $10-20$ of Algorithm 4 and also shown in Figure 1(b).).

For each level (which is an iteration in the pseudo-code as shown in line $11-20$ of Algorithm 4) in the second phase, the process can be further divided into two sub-phases:

- every dominator aggregates its data to its corresponding connectors (as shown in line $11-15$ of Algorithm 4);
- every connector transmits its data to the dominator in the upper level (as shown in line $16-20$ of Algorithm 4).

**Algorithm 2** Link Coloring by receiver locations

**Input:** Set of links $\mathcal{L} = \{l_1, l_2, \cdots, l_n\}$;
**Output:** Colored links;

1: **for** $r = 0, \cdots, K-1$ and $s = 0, \cdots, K-1$ **do**
2: **for** $i, j \in \mathbb{Z}$ **do**
3: **if** $i \mod K = r$ and $j \mod K = s$ **then**
4: select one link from $\mathcal{L}$ whose receiver is located within $g_{i,j}$;
5: all the selected links are colored as $C_{r,s}$;

**Algorithm 3** Link Coloring by sender locations

**Input:** Set of links $\mathcal{L} = \{l_1, l_2, \cdots, l_n\}$;
**Output:** Colored links;

1: **for** $r = 0, \cdots, K-1$ and $s = 0, \cdots, K-1$ **do**
2: **for** $i, j \in \mathbb{Z}$ **do**
3: **if** $i \mod K = r$ and $j \mod K = s$ **then**
4: select one link whose sender is located within $g_{i,j}$;
5: all the selected links are colored as $C_{r,s}$;

## IV. Performance Analysis of Our Algorithm

In this section we first show that our schedule is valid, *i.e.*, all receiver nodes have SINR at least $\beta$. We then analytically prove that the latency of data aggregation using our schedule is at most a small constant factor of the optimum.

### A. Correctness

We first prove that at any single time-slot, every receiver in our schedule can receive data successfully.

*Theorem 1:* After the plane is divided into cells, we can set $K = \lceil (\frac{4\beta\tau P \cdot \ell^{-\alpha}}{(\sqrt{2})^{-\alpha} P \cdot \ell^{-\alpha} - \beta N_0})^{\frac{1}{\alpha}} + 1 + \sqrt{2} \rceil$ to ensure the SINR at every receiver is at least the threshold $\beta$. Here $\tau = \frac{\alpha(1+2^{-\frac{\alpha}{2}})}{\alpha-1} + \frac{\pi 2^{-\frac{\alpha}{2}}}{2(\alpha-2)}$, $\ell = \delta\mathbf{r}/\sqrt{2}$.

### B. Upper-bound on the latency of Algorithm 4

In this section, we prove that the latency achieved by Algorithm 4 is $O(D + \Delta)$ where $D$ is the diameter of the network $G(V, \delta\mathbf{r})$ and $\Delta$ is the maximum node degree in $G$.

We first bound the number of connectors that a dominator $u$ will use to connect to all dominators within two-hops. Our proof is based on a technique lemma implied from lemmas proved in [24].

*Lemma 2:* [24] Suppose that dominator $v$ and $w$ are within 2-hops of dominator $u$, $v'$ and $w'$ are the corresponding connectors for $v$ and $w$ respectively. Then either $|wv'| \leq 1$ or $|vw'| \leq 1$ if $\angle vuw \leq 2 \arcsin \frac{1}{4}$.

*Lemma 3:* There are at most 12 connectors for each dominator.

*Proof:* After we delete all the redundant connectors, for each connector, there exists at least one dominator which has only this connector as its adjacent connector. This can be proven by contradiction as follows, assume there exists a connector $c_i$ such that each of its adjacent dominators is also a neighbor of some other connector, then we can always delete this redundant connector $c_i$.

**Algorithm 4** Minimum Latency Aggregation Scheduling

1: Partition the deployment plane into cells, each with side length $\delta \mathbf{r}/\sqrt{2}$.
2: Construct an aggregation tree using Algorithm 1;
3: **while** there still exist some unmarked dominatees **do**
4: For each dominator, we select an unmarked neighboring dominatee. All the links form a set $\mathcal{L}$.
5: We then apply Algorithm 2 to color the links in $\mathcal{L}$;
6: **for** $r = 0, \cdots, K-1$ and $s = 0, \cdots, K-1$ **do**
7: schedule all links in $\mathcal{L}$ with color $C_{r,s}$;
8: Mark all the selected dominatees;
9: Unmark all nodes (including all dominators and dominatees);
10: **for** $i$= $R$ to 2 **do**
11: **while** there still exist some dominator in level $i$ whose data have not been aggregated to their parent nodes (connectors) **do**
12: For every dominator in level $i$, we select the link from itself to its parent. All the links form a set $\mathcal{L}$.
13: Apply Algorithm 3 to color the links in $\mathcal{L}$;
14: **for** $r = 0, \cdots, K-1$ and $s = 0, \cdots, K-1$ **do**
15: schedule all links in $\mathcal{L}$ with color $C_{r,s}$;
16: **while** $\exists$ some dominators in level $i-1$ that did not aggregate the data from all its connectors **do**
17: For every dominator in level $i$, we select the link from its connector to itself, and this link to $\mathcal{L}$.
18: Apply Algorithm 2 to color the links in $\mathcal{L}$;
19: **for** $r = 0, \cdots, K-1$ and $s = 0, \cdots, K-1$ **do**
20: schedule all links in $\mathcal{L}$ with color $C_{r,s}$;

It implies that if there are 13 connectors for one dominator, we must have at least 13 non-sharing dominators. By Lemma 2, we know that there are at least two dominators will share a same connector which contradicts to fact that all of those 13 dominators should be non-sharing with each other. ■

*Lemma 4:* Each connector has at most $12 \cdot 5$ neighboring connectors.

*Proof:* This lemma can be proven by contradiction. We first prove that each connector has at most 5 neighboring dominators, assuming there exists a connector $c_i$ which has at least 6 neighboring dominators, then there must exist two dominators $d_j$ and $d_k$ such that $\angle d_j c_i d_k \leq 60°$, this contradicts to the fact that no two dominators are adjacent to each other.

Then since each connector has at most 5 neighboring dominators and there are at most 12 connectors for one dominator, so there are no more than $12 \cdot 5$ neighboring connectors for each connector, otherwise, there must exist one dominator which has at least 13 neighboring connectors. It contradicts to our previous results. This finishes the proof. ■

*Lemma 5:* In the first phase of Algorithm 4, all the dominators can aggregate the data from all its corresponding dominatees in $K^2 \cdot \Delta$ time-slots.

*Proof:* Since the maximum degree of each dominator $d_i$ is $\Delta$ and the side length of each cell is $\delta \mathbf{r}/\sqrt{2}$, so there are at most 1 dominator and $\Delta$ dominatees falling in the same cell. It follows that the load of each cell in the first phase is at most $\Delta$. According to Theorem 1, because every cell can be scheduled at least once every $K^2$ time-slots, all the links can be scheduled in $K^2 \cdot \Delta$ time-slots. ■

*Lemma 6:* In the second phase of Algorithm 4, the sink can receive all the aggregated data in at most $62 \cdot K^2 \cdot R$ time-slots.

*Proof:* Since there is at most one dominator fallen in one cell, the total load of each cell in the first sub-phase of phase two is at most 1. Further, Lemma 4 shows that every connector has at most 60 neighboring connectors, so the total number of connectors fallen in the same cell is 61, then we get the load of each cell in the second sub-phase is at most 61. By summarizing the above analysis on the two sub-phases, we have the total load of each cell is at most $1 + 61 = 62$. Again, based on Theorem 1, we find that all the data can be aggregated from level $i+1$ to $i$ by at most $62 \cdot K^2$ time-slots, note that $K$ is a constant. Since the depth of our aggregation tree is $R$, we finally get a upper-bound $62 \cdot K^2 \cdot R$ on the latency for the second phase. ■

Lemma 5 and Lemma 6 together imply following theorem.

*Theorem 7:* The latency of Algorithm 4 is at most $K^2 \cdot R + K^2 \cdot \Delta$.

## V. Overall lower-bound on latency

Here we discuss the overall lower-bound on the latency for data aggregation under the physical interference model. An overall lower-bound is the minimum time-slots (which may not be tight) needed to finish the data aggregation by any possible algorithm for the given network $G(V, \delta \mathbf{r})$.

*Lemma 8:* Under the physical interference model, for a communication graph $G(V, \delta \mathbf{r})$, it requires at least $R$ time-slots for any algorithm to finish the aggregation transmission . Here $R$ is the maximum hop-distance for any node to the sink node in $G(V, \delta r)$.

*Proof:* We can see that the scheduled links can only be selected from the edges in $G(V, \delta r)$. It requires at least $R$ time-slots for the farthest node $v$ to transmit its data to the sink node $v_s$. Since $R \geq \frac{D}{2}$, the latency is at least $D/2$. ■

*Lemma 9:* Under the physical interference model, for a communication graph $G(V, \delta \mathbf{r})$, there are at most $\omega = \frac{(\delta \mathbf{r})^\alpha}{\beta} - 1 = \delta^\alpha \frac{N_0}{P} - 1$ senders transmitting simultaneously which are all neighbors of a single node.

*Corollary 10:* Under the physical interference model, for a communication graph $G(V, \delta \mathbf{r})$, it requires $\frac{\Delta}{\omega}$ time-slots for any algorithm to finish the aggregation transmission.

*Proof:* We consider the node $v$ whose degree reaches maximum value $\Delta$ (if there are multiple such nodes, choose one randomly). Since for every neighboring node of $v$, it needs to report its data to the sink. This means every neighboring node of $v$ needs to transmit at least once. By Lemma 9, at each time-slot, at most $\omega$ neighboring node can send simultaneously, thus it requires at least $\frac{\Delta}{\omega}$ time-slots to finish aggregation scheduling. ■

From Lemma 8 and Corollary 10, we can derive the overall lower-bound for data aggregation scheduling as follows.

*Theorem 11:* Under the physical interference model, for any $\delta < 1$, for a communication graph $G(V, \delta \mathbf{r})$, it requires

$\max\{R, \Delta/\omega\}$ time-slots for *any* algorithm to finish the aggregation transmission. Here $R$ is the maximum hop-distance for any node to the sink node in graph $G(V, \delta r)$ and $\Delta$ is the maximum degree of the graph.

This theorem implies that our algorithm achieves the asymptotically optimum latency for data aggregation when we can only use links with length at most $\delta\mathbf{r}$.

## VI. Compare With Optimum Using All Links

In this section, we then study the performance of our algorithm compared with the optimum solution when all links in the network $G(V, \mathbf{r})$ could be used for data aggregation. We first study the case when $V$ is randomly placed and then study the case for a set $V$ of arbitrarily placed nodes.

### A. Randomly Placed Nodes

When we study WSNs deployed in real life, we find that in many scenarios, the sensors are deployed randomly and uniformly. Thus we can use a random graph to denote the topology in which all the sensors are deployed randomly and uniformly. For random networks, we assume that all $n$ nodes are randomly deployed in a square region of side-length $A$. Notice that we assume that the transmission power of every node is a fixed constant $P$ and the background noise is $N_0$. This implies that the maximum link length is $\mathbf{r}$. Then it is well-known that to ensure that the random network is connected with high probability, the degree of each node should be in the order of $\Omega(\log n)$. Then the side-length $A$ is assumed to be in the order $A = O(\frac{n}{\log n})$ here. Before we study the latency for random networks, we first introduce the following lemma.

*Lemma 12:* [18] Assume there is a graph with $n$ nodes randomly deployed in a square region with side-length $O(\frac{n}{\log n})$. We partition the deployment square into cells, each of side-length a constant $a$. Then there is a sequence of $\delta(n) \to 0$ such that

$$\mathbf{Pr}(\text{every cell contains a node}) \geq 1 - \delta(n)$$

In a graph $G$, the hop distance between a pair of nodes $u$ and $v$ is defined as the smallest number of hops between them in the topology graph $h_G(u, v)$. In random wireless sensor networks, if we connect any pair of sensor nodes with distance smaller than $\delta\mathbf{r}$ and $\mathbf{r}$ respectively, we get two topology graphs $G(V, \delta\mathbf{r})$ and $G(V, \mathbf{r})$.

*Lemma 13:* For any pair of nodes $u$ and $v$, *w.h.p*, we have $h_{G(V,\delta\mathbf{r})}(u, v) \leq \rho \cdot h_{G(V,\mathbf{r})}(u, v)$, where $\rho$ is a constant.

*Proof:* In a graph $G(V, \delta\mathbf{r})$, any pair of nodes $u$ and $v$ from two adjacent cells (sharing a common side) will be able to communicate with each other directly. Then by choosing one node from each cell which is crossed by segment $uv$, we can connect $u$ and $v$ using $\Theta(\|uv\|/\mathbf{r})$ hops.

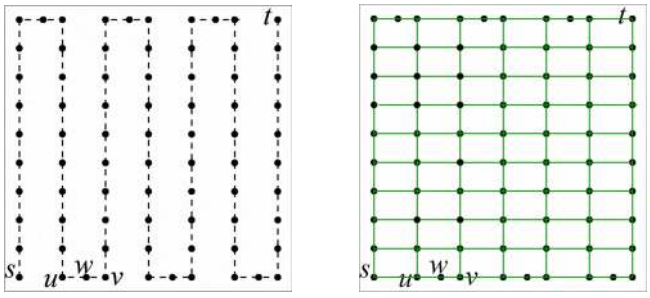
Similarly, if we partition the network into squares with side-length $\delta\mathbf{r}/\sqrt{5}$. Then together with the assumption that the random network is connected with high probability, it follows that by only using the links with length $\delta\mathbf{r}$, we can connect any two vertices $u$ and $v$ within $\Theta(\|uv\|/\delta\mathbf{r})$ hops. Notice that $\Theta(\|uv\|/\delta\mathbf{r}) = \Theta(\|uv\|/\mathbf{r})$. It implies that under random network model, the minimum number of hops used to connect any two vertices in $G(V, \mathbf{r})$ is in the same order of the minimum number of hops used in $G(V, \delta\mathbf{r})$. ■

*Theorem 14:* If there is an algorithm for aggregation scheduling in $G(V, \delta\mathbf{r})$ with approximation ratio $a$, then we can achieve an solution for the original topology $G(V, \mathbf{r})$ with approximation ratio $\rho \cdot a$ *w.h.p.*

*Proof:* Assume the optimum schedules for $G(V, \mathbf{r})$ and $G(V, \delta\mathbf{r})$ are $OPT(G(V, \mathbf{r})), OPT(G(V, \delta\mathbf{r}))$. We overload the terms as the latency achieved using the schedules respectively. By Lemma 13, the hop distance in the graph $G(V, \delta\mathbf{r})$ is at most $\rho$ times of the hop distance in the original graph $G(V, \mathbf{r})$. So $R_{\delta\mathbf{r}} = \max_{u \in V} \min h(u, v_s)$ of the graph $G(V, \delta\mathbf{r})$ is at most $\rho$ times of $R_{\mathbf{r}} = \max_{u \in V} \min h(u, v_s)$ of the original graph $G(V, \mathbf{r})$. Here $h(u, v_s)$ is the number of hops between the node $u$ and the sink node $v_s$. By Lemma 8, $R$ is a lower bound of the data aggregation under the physical interference model. At the same time, the maximum degree of graph $G(V, \delta\mathbf{r})$ ($\Delta_{\delta\mathbf{r}}$) is at most the maximum degree of graph $G(V, \mathbf{r})$ ($\Delta_{\mathbf{r}}$). In Corollary 10, we have proved that $\Delta$ is another lower bound of the data aggregation under physical interference model. Then $OPT(\delta\mathbf{r}) \leq \rho \cdot OPT(\mathbf{r})$. Thus by definition, $ALG(\delta\mathbf{r}) \leq a \cdot OPT(\delta\mathbf{r}) \leq \rho \cdot a \cdot OPT(\mathbf{r})$. ■

### B. Arbitrarily Placed Nodes

Theorem 14 shows that the optimum solutions for the communication graphs $G(V, \delta\mathbf{r})$ and $G(V, \mathbf{r})$ are in the same order for random wireless sensor networks. One may conjecture that this nice property holds for an arbitrary network. Unfortunately, Fig 2 gives such an example where the latency of data aggregation using these two graphs will be different when $0 < \delta < 1$ is a given constant.




Graph $G(V, \delta\mathbf{r})$ Graph $G(V, (\delta + \epsilon)\mathbf{r})$

Fig. 2. A network example: the optimum solutions for $G(V, \mathbf{r})$ and $G(V, \delta\mathbf{r})$ are not in the same order. The distance $d_h$ between consecutive vertical lines is $(\delta + \epsilon)\mathbf{r}$. The distance between two consecutive nodes on a vertical line is $d_v = \delta\mathbf{r}$.

In Fig 2, there are $n$ nodes deployed in a rectangle region. There are $\sqrt{n}$ vertical evenly spaced lines with Euclidean distance $d_h = (\delta + \epsilon)\mathbf{r}$ between consecutive lines. Here $\delta < 1$ is a given constant and $0 < \epsilon < 1 - \delta$. For example, we can set $\epsilon = \min(\delta, \frac{1-\delta}{2})$. For each such line, we evenly place $\sqrt{n}$ nodes with Euclidean distance $d_v = \delta\mathbf{r}$ between consecutive nodes. Additionally, $\sqrt{n}-1$ nodes, like $w$ between $u$ and $v$ in the example, act as the bridges to guarantee the network connectivity when doing aggregation in graph $G(V, \delta\mathbf{r})$. Distance between $w$ and $u$ (or $v$) is smaller or equal to $\delta\mathbf{r}$, and distances between $w$ and other nodes are larger than

$\delta\mathbf{r}$. Thus, in graph $G(V, \delta\mathbf{r})$, if we want to aggregate the data from $s$ to $t$, we should strictly follow the dashed path as shown in Fig 2. The latency of data aggregation in $G(V, \delta\mathbf{r})$ thus is at least $n$. On the other hand, in graph $G(V, \mathbf{r})$, the red solid path may be a possibility for getting data from a source node $s$ to the sink node $t$. It is easy to show that for graph $G(V, \mathbf{r})$, the network radius is in the order of $\Theta(\sqrt{n})$. On the other hand, the graph $G(V, \mathbf{r})$ contains the graph $G(V, (\delta + \epsilon)\mathbf{r})$ as a subgraph. Notice that, for the graph $G(V, (\delta + \epsilon)\mathbf{r})$, the maximum node degree is $\Delta \leq 10$ and the radius of the network is at most $R \leq 2\sqrt{n}$. Then using our scheduling algorithm, for graph $G(V, (\delta + \epsilon)\mathbf{r})$, the latency of data aggregation is at most $O(\Delta + R) = O(\sqrt{n})$. Consequently, the latency of data aggregation in the original network $G(V, \mathbf{r})$ is at most $O(\sqrt{n})$. Thus, we have the following theorem

*Theorem 15:* For any constant $0 < \delta < 1$, there are examples of networks with $n$ nodes $V$ in two-dimensional such that the latency using only links in $G(V, \delta\mathbf{r})$ is at least $n$ while the latency using all links in $G(V, \mathbf{r})$ is only $O(\sqrt{n})$ using Algorithm 4. In other words, there is no universal constant $\delta$ that ensures a constant approximation ratio for the latency of data aggregation by *any* algorithm that only considers links in $G(V, \delta\mathbf{r})$.

We then show that the ratio of the latency for data aggregation in $G(V, \delta\mathbf{r})$ over the latency for data aggregation in $G(V, \mathbf{r})$ is at most $O(n^{2/3})$ for any network of $n$ nodes $V$ in a two-dimensional space. Let us consider any $n$ nodes $V$ distributed in a two-dimensional region. Let $\Delta$ and $R$ be the maximum node degree and radius for network $G(V, \mathbf{r})$ respectively. Obviously, we have $\pi R^2 \cdot \Delta \geq n$. This implies that $\max(R, \Delta) \geq \pi^{1/3} n^{1/3}$. Thus, the latency of data aggregation in $G(V, \mathbf{r})$ is at least $\pi^{1/3} n^{1/3}$. On the other hand, the latency of data aggregation in $G(V, \delta\mathbf{r})$ is at most $n$. Then we have the following theorem.

*Theorem 16:* The ratio of the latency for data aggregation in $G(V, \delta\mathbf{r})$ over the latency for data aggregation in $G(V, \mathbf{r})$ is at most $n^{2/3}/\pi^{1/3}$ for any network of $n$ nodes $V$ in a two-dimensional space.

This implies that our algorithm achieves a latency that is at most $O(n^{2/3})$ factor of the latency by the optimum algorithm for $G(V, \mathbf{r})$. Theorem 15 implies that there are network examples such that the latency achieved by our algorithm is at least $\Omega(n^{1/2})$ factor of the latency by the optimum algorithm for $G(V, \mathbf{r})$. Note that there is a gap between the performance bound of our algorithm for network $G(V, \mathbf{r})$. We conjecture that

*Conjecture 17:* The latency achieved by our algorithm is $\Theta(n^{1/2})$ factor of the latency by the optimum algorithm for $G(V, \mathbf{r})$.

## VII. Simulation results

In this section, we present our simulation results. We randomly deploy a number of $n$ nodes in a two-dimensional square area and randomly choose one node as the sink node. The objective is to measure the latency for the sink node to get the aggregation (*max, sum or average*) result of all data using our aggregation scheduling algorithm (Algorithm 4).

TABLE I
The parameters of physical interference model

| parameter | value | parameter | value |
|---|---|---|---|
| $\beta$ | 1 | $\alpha$ | 2.5 |
| $N_0$ | 0.1 | $P$ | 15 |

Note that latency is defined as the number of time-slots needed to aggregate all data from within the network to the sink node. We evaluate Algorithm 4 from different perspectives under the physical interference model. For simplicity, we first list the parameter setting of physical interference model in Table I. Note that the parameter $\beta$ is set to 1 while in reality this threshold could be much more than 1. Based on the parameters, we can compute the value of $\mathbf{r}$.

### A. The effects of $\Delta$ and $R$ in Algorithm 4

To evaluate the effect of maximum node degree $\Delta$, we measure the latencies of Algorithm 4 with varied values of $\Delta$.

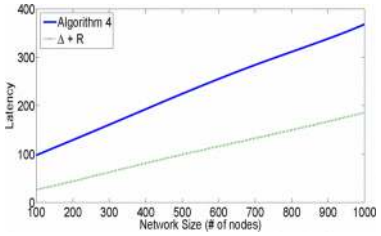
We fix the network deployment area as $(500m \times 500m)$ and vary the network size (# of nodes) from 100 to 1000 with step 50. By connecting every pair of nodes with distance no larger than $\delta\mathbf{r}$, we get a communication graph. Here we set $\delta = 0.6$ which can ensure that the corresponding communication graph is connected. Later we will discuss how to set the value of $\delta$.

By fixing the size of network deployment area, we find that the network radius $R$ is nearly fixed (around 25). At the same time, the network density ($\Delta$) increases monotonously with network size. We measure the performance of Algorithm 4 for 50 times under this condition, the average performance is illustrated in Figure 3(a). We can see that the latency increases monotonously when the network density ($\Delta$) increases. That is because the interferences will be greatly decreased when our algorithm aggregates data level by level.

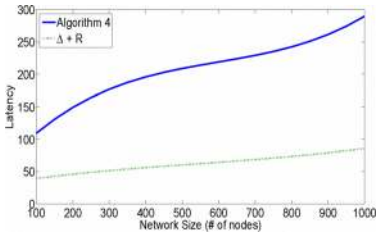
To evaluate the effect of network radius $R$, we create a network instance similar to the first one with two crucial modifications: we vary the network deployment area from $(200m \times 200m)$ to $(1000m \times 1000m)$ gradually with the network size and set $\delta = 0.1$. In other words, we fix the node density in the network deployment area. We find that $\Delta$ is nearly fixed (around 25) as well in the corresponding communication graph. At the same time, $R$ increases monotonously with the network size. We measure the performance of Algorithm 4 for 50 times under this condition, the average performance is illustrated in Figure 3(b). We can see that there is still a gap between the latency of Algorithm 4 and $\Delta + R$. This is because we use $\Delta + R$ as a lower bound on the latency. This means that either Algorithm 4 can be potentially further improved or the lower bound $\Delta + R$ should be improved for data aggregation under the physical interference model. We leave it as an open question to further close the gap between upper bound and lower bound on latency.

### B. Compare Algorithm 4 with compressive scheduling

In this section we show that in practice, our algorithm can be further improved, although the theoretical performances remain the same here. The new method, named *compressive scheduling*, will further reduce the latency by merging the links

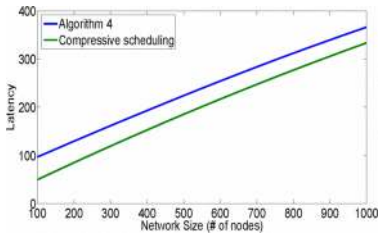



(a) Varied density

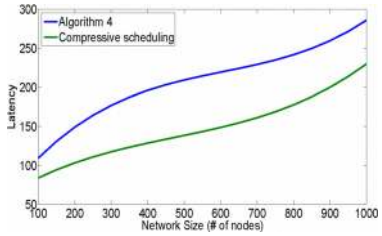



(b) Fixed density

Fig. 3. Simulation results of running Algorithm 4 with varied $\Delta$ and $R$.

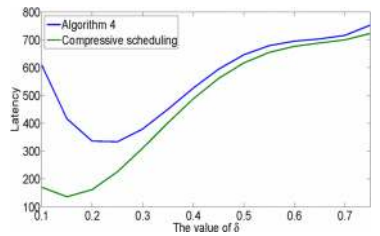



(a) Varied density

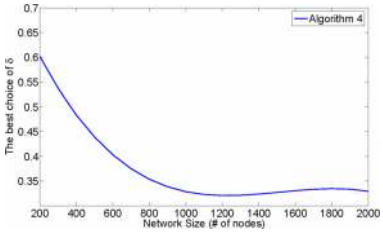



(b) Fixed density

Fig. 4. Comparing the latencies of Algorithm 4 and compressing scheduling. The legend "Compress" in both figures means the compressive scheduling algorithm.




(a) Latency with $\delta$




(b) The best choice of $\delta$

Fig. 5. Simulation results of running Algorithm 4 and compressive scheduling with varied $\delta$.

scheduled in different slots to a single slot without violating the SINR. We will then compare the performance of our algorithm (Algorithm 4) with that of compressive scheduling algorithm. Our compressive scheduling algorithm, which is in fact a greed algorithm, is described as follows.

1) find all leave nodes, say $N$, in the same BFS tree constructed in Algorithm 1.
2) partition the deployment region into cells using the same method in Section III-B.
3) find a subset $N_1 \subseteq N$ such that each cell contains at most one node from $N_1$. If some cell contains multiple nodes from $N$, we choose the one with the largest level in BFS tree.
4) for each node in $N_1$, we find the corresponding link from itself to its parent in the BFS tree. All the links found form a link set $\mathcal{L}$.
   - color the links in $\mathcal{L}$ using Algorithm 2, find a subset $\mathcal{L}_1$ of links with monotone color with the largest size. We then try to add more links greedily without destroying the SINR threshold constraint ( Equation (1) );
   - apply Algorithm 1 in [15] to find a feasible solution $\mathcal{L}_2$ which is a subset of $\mathcal{L}$;
   - select one of the two sets $\mathcal{L}_1$ and $\mathcal{L}_2$ with a larger size. Let all links in the selected set transmit simultaneously. Delete all transmitters which are leaf nodes in the BFS tree. Update the BFS tree accordingly.
5) repeat step 1)–4) until there is no leaf node in the BFS tree. This means that all data have been aggregated to the sink node.

Clearly, theoretically, the latency of our new compressive scheduling algorithm for data aggregation is at most the latency of our previous method. We do simulations for compressive scheduling algorithm and Algorithm 4 for 50 times and compare their latencies. Note all comparisons are fairly conducted in the same communication graph, and in each communication graph data are aggregated from the same set of nodes to the same sink node. Figure 4 shows the average latencies of different algorithms respectively. From Figure 4(a) to Figure 4(b), we can see that when network is sparse, compressive scheduling has remarkable effect on the latencies of data aggregation. It significantly reduces the latencies.

### C. Discussion on the choices of $\delta$

As we know, $\delta$ is a crucial parameter on the latencies of data aggregation. In previous subsections, we all set $\delta$ as some default values, which may not be an optimum choice to minimize the latencies. Here we discuss on how to choose the value of $\delta$. Generally, the smaller the value of $\delta$, the smaller the length of links we can choose in $G(V, \delta \mathbf{r})$. Thus we have less links in the corresponding communication graph $G$, this implies that in $G$:

1) each node has less incident links (edges), thus the maximum node degree $\Delta$ becomes smaller. This will help to reduce the latency of data aggregation.
2) the maximum length of links in $G$ becomes smaller, thus the network radius $R$ becomes larger. This will then increase the latency of the data aggregation.

Thus, it is not straightforward whether we should select smaller $\delta$ or larger $\delta$ here. Since the time spent (denoted as $l_1$) for collecting data to dominators increases monotonically with $\Delta$. Thus $l_1$ becomes smaller as $\delta$ becomes smaller. The time spent (denoted as $l_2$) for collecting data from dominators to dominators level by level increases monotonically with $R$. Thus $l_2$ becomes larger as $\delta$ becomes smaller. To sum $l_1$ and $l_2$ as the total latency, when $\delta$ become smaller, it is not clear whether the total latency will decrease or increase.

However, $\delta$ cannot be too small or too large (close to 1). First if $\delta$ is too small, the communication graph may not be connected, thus some un-connected node can not send its datum to the sink node. Second, if $\delta$ is close to 1, then the network is strongly connected by a lot of long

links. These long links prevent the simultaneous transmissions, which potentially increases the latency. From this perspective, a link with smaller length is preferred for scheduling. We conjecture the smallest $\delta$ which can ensure the connectivity of $G(V, \delta \mathbf{r})$ is an optimum choice. So far our conjecture cannot be proved formally.

We conduct extensive simulations to see the effect of $\delta$ in different scenarios. Fig 5(a) shows the latencies of both Algorithm 4 and compressive scheduling with varied $\delta$. Fig 5(b) shows the best choices of $\delta$ with varied network size in a fixed deployment area. Here a best choice means that the latency of Algorithm 4 achieves minimum when $\delta$ is set as this value.

## VIII. Related work

Both data aggregation in sensor network and link scheduling under the physical interference model have been both well studied recently. However, no paper combine these issues together, that is, studying the problem of data aggregation under the physical interference model. Therefore our literature part can only review data aggregation under graph-based interference models and link scheduling separately.

### A. Related work on data aggregation

There are a lot of existing researches on in-network aggregation [7], [8], [10], [22]. The tradeoff between energy consumption and time latency was considered in [28] and some heuristic algorithms for data aggregation were proposed [1], [23], aiming at reducing time latency and energy consumption. Kesselman *et al.* [16] proposed a randomized and distributed algorithm for aggregation in wireless sensor networks with an expected latency of $O(\log n)$. A collision-free scheduling method for data collection is proposed in [17], aiming at optimizing energy consumption and reliability.

All these work did not discuss the minimal-latency aggregation scheduling problem. In addition, the minimum latency of data aggregation problem was proved to be NP-hard [6]. The distributed algorithms for aggregation scheduling were proposed in [2]. It proposed a distributed scheduling algorithm generating a collision-free schedule that has a latency bound of $24D + 6\Delta + 16$, where $D$ is the network diameter when the conflict-graph is the original UDG communication graph. This was recently improved to $16R + \Delta$ model by Xu *et al.* [27] and $\left(1 + (\log R / \sqrt[3]{R})\right) R + \Delta$ by Wan *et al.* [25] under the same network model.

### B. Link scheduling under the physical interference model

The problem of joint scheduling and power control under the physical interference model has been well studied previously. For instance, in [9], [11], optimization models and heuristics for this problem are proposed. In [13], [20], topology control with SINR constraints is studied. In [19], a power-assignment algorithm which schedules a strongly connected set of links in poly-logarithmic time is presented. In [4], the combined problem of routing and power control is addressed.

In [14], the scheduling problem without power control under the physical interference model, where nodes are arbitrarily distributed in Euclidean space, has been shown to be NP-complete. A greedy scheduling algorithm with approximation ratio of $O(n^{1-2/(\Psi(\alpha)+\epsilon)}(\log n)^2)$, where $\Psi(\alpha)$ is a constant that depends on the path-loss exponent $\alpha$, is proposed in [3]. Notice that this result can only hold when the nodes are distributed uniformly at random in a square of unit area. In [14], an algorithm with a factor $O(g(L))$ approximation guarantee in arbitrary topologies, where $g(L) = \log \vartheta(L)$ is the diversity of the network, is proposed. In [5], an algorithm with approximation guarantee of $O(\log \Delta)$ was proposed, where $\Delta$ is the ratio between the maximum and the minimum distances between nodes. Obviously, it can be arbitrarily larger than $\vartheta(L)$. Most recently, Goussevskaia *et al.* [15] propose an algorithm which has a constant approximation guarantee. Unfortunately, their method [15] works correctly only if the background noise is 0. Recently, Xu *et al.* [26] resolved this issue by proposing an efficient link scheduling algorithm that has a constant approximation ratio, *i.e.*, the number of scheduled links is at least a constant factor of the optimum.

## IX. Conclusions

In this paper we study the aggregation scheduling with minimum latency under the physical interference model in wireless sensor networks. We proposed a distributed aggregation scheduling algorithm, and proved that the latency achieved by our algorithm is $O(R + \Delta)$ time-slots, which is within a constant factor of optimum if we can only use links in $G(V, \delta \mathbf{r})$. We further provided the overall lower-bound of latency by any algorithm for aggregation scheduling under the physical interference model. We also analyzed aggregation latency in a randomly deployed wireless networks. Some interesting questions are left for future research. The first is to improve the constant approximation ratio of our scheduling algorithm. The second is to design efficient data aggregation method that has the asymptotical optimum performance guarantee compared with the optimum latency using $G(V, \mathbf{r})$. The third is to extend our algorithm to deal with a more general path loss model.

## X. Acknowledgment

The research of authors are partially supported by NSF CNS-0832120, National Natural Science Foundation of China under Grant No. 60828003, No.60773042 and No.60803126, the Natural Science Foundation of Zhejiang Province under Grant No.Z1080979, National Basic Research Program of China (973 Program) under grant No. 2010CB328100, National Basic Research Program of China (973 Program) under grant No. 2006CB30300, the National High Technology Research and Development Program of China (863 Program) under grant No. 2007AA01Z180, Hong Kong RGC HKUST 6169/07, the RGC under Grant HKBU 2104/06E, and CERG under Grant PolyU-5232/07E.

## References

[1] Annamalai, V., Gupta, S., and Schwiebert, L. On tree-based convergecasting in wireless sensor networks.

[2] Bo Yu, J. L., and Li, Y. Distributed Data Aggregation Scheduling in Wireless Sensor Networks. In *INFOCOM* (2009).

[3] BRAR, G., BLOUGH, D., AND SANTI, P. Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks. In *ACM MobiCom* (2006), pp. 2–13.

[4] CHAFEKAR, D., KUMAR, V., MARATHE, M., PARTHASARATHY, S., AND SRINIVASAN, A. Cross-layer latency minimization in wireless networks with SINR constraints. In *ACM MobiCom* (2007), pp. 110–119.

[5] CHAFEKAR, D., KUMAR, V., MARATHE, M., PARTHASARATHY, S., AND SRINIVASAN, A. Approximation Algorithms for Computing Capacity of Wireless Networks with SINR Constraints. In *IEEE INFOCOM* (2008), pp. 1166–1174.

[6] CHEN, X., HU, X., AND ZHU, J. Minimum Data Aggregation Time Problem in Wireless Sensor Networks. *LNCS 3794* (2005), 133.

[7] CHU, D., DESHPANDE, A., HELLERSTEIN, J., AND HONG, W. Approximate data collection in sensor networks using probabilistic models. In *International Conference on Data Engineering (ICDE)* (2006).

[8] CONSIDINE, J., LI, F., KOLLIOS, G., AND BYERS, J. Approximate aggregation techniques for sensor databases. In *ICDE*, 2004, pp. 449–460.

[9] CRUZ, R., AND SANTHANAM, A. Optimal routing, link scheduling and power control in multihop wireless networks. In *IEEE INFOCOM* , vol. 1., 2003.

[10] DESHPANDE, A., GUESTRIN, C., HONG, W., AND MADDEN, S. Exploiting Correlated Attributes in Acquisitional Query Processing. In *ICDE* , 2005.

[11] ELBATT, T., AND EPHREMIDES, A. Joint scheduling and power control for wireless ad hoc networks. *IEEE Transactions on Wireless Communications, 3*, 1 (2004), 74–85.

[12] GANDHAM, S., ZHANG, Y., AND HUANG, Q. Distributed Minimal Time Convergecast Scheduling in Wireless Sensor Networks. In *IEEE ICDCS,* (2006).

[13] GAO, Y., HOU, J., AND NGUYEN, H. Topology Control for Maintaining Network Connectivity and Maximizing Network Capacity under the Physical Model. In *IEEE INFOCOM* (2008), pp. 1013–1021.

[14] GOUSSEVSKAIA, O., OSWALD, Y., AND WATTENHOFER, R. Complexity in geometric SINR. In *ACM MobiCom* (2007), pp. 100–109.

[15] GOUSSEVSKAIA, O., W. R. H. M., AND WELZL, E. Capacity of Arbitrary Wireless Networks. In *IEEE INFOCOM* (2009).

[16] KESSELMAN, A., AND KOWALSKI, D. Fast distributed algorithm for convergecast in ad hoc geometric radio networks. *Journal of Parallel and Distributed Computing 66*, 4 (2006), 578–585.

[17] LEE, H., AND KESHAVARZIAN, A. Towards Energy-Optimal and Reliable Data Collection via Collision-Free Scheduling in Wireless Sensor Networks. In *IEEE INFOCOM* (2008), pp. 2029–2037.

[18] LI, X. Multicast Capacity of Wireless Ad Hoc Networks. *IEEE/ACM Transaction on Networking* (2008).

[19] MOSCIBRODA, T., AND WATTENHOFER, R. The Complexity of Connectivity in Wireless Networks. In *IEEE INFOCOM* (2006).

[20] MOSCIBRODA, T., WATTENHOFER, R., AND ZOLLINGER, A. Topology control meets SINR: the scheduling complexity of arbitrary topologies. In *ACM MobiCom* (2006), pp. 310–321.

[21] P.-J. WAN, K. A., AND FRIEDER, O. Distributed construction of connected dominating set in wireless ad hoc networks. In *IEEE INFOCOM* (2002).

[22] SHRIVASTAVA, N., BURAGOHAIN, C., AGRAWAL, D., AND SURI, S. Medians and beyond: new aggregation techniques for sensor networks. In *ACM SenSys* (2004), pp. 239–249.

[23] UPADHYAYULA, S., ANNAMALAI, V., AND GUPTA, S. A low-latency and energy-efficient algorithm for convergecast in wireless sensor networks. In *IEEE GLOBECOM* (2003), vol. 6.

[24] WAN, P.J. AND YI, C.W. AND JIA, X. AND KIM, D. Approximation algorithms for conflict-free channel assignment in wireless ad hoc networks, *Wiley Wireless Communication and Mobile Computing* (2006), vol. 6, pages 201.

[25] WAN, P.J. AND SCOTT, C.H.H. AND WANG, L. AND WAN, Z. AND JIA, X. Minimum-latency aggregation scheduling in multihop wireless networks, In *ACM MOBIHOC* (2009).

[26] XU, X.-H., TANG, S.-J. A Constant Approximation Algorithm for Link Scheduling in Arbitrary Networks With Physical Interference Model *ACM MobiHoc FOWANC workshop*, 2009

[27] XU, X.-H., WANG, S.-G., MAO X.-F., TANG S.-J. AND LI, X.-Y. An Improved Approximation Algorithm for Data Aggregation in Multi-hop Wireless Sensor Networks *ACM MobiHoc FOWANC workshop*, 2009

[28] YU, Y., KRISHNAMACHARI, B., AND PRASANNA, V. Energy-latency tradeoffs for data gathering in wireless sensor networks. In *IEEE INFOCOM* (2004), vol. 1.


## APPENDIX

### PROOF OF THEOREM 1.

*Proof:* When a sender $u$ transmits to a node $v$, the SINR at the receiver $v$ is $\frac{P(\sqrt{2}\ell)^{-\alpha}}{N_0+\mathcal{P}}$. Here the $\mathcal{P}$ is the total interference at the receiver $v$ from all other simultaneous senders. As shown in Fig. 6, the shortest hop-number of $v$ and another sender is at least $h' = h - (1+\sqrt{2})$.

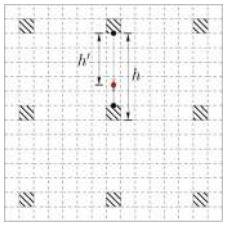



Fig. 6. Cell partition of the region ($h$ and $h'$ are hop-number).

We then bound the total interference $\mathcal{P}$.

$$
\begin{aligned}
\mathcal{P} &\le \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} P\left(\sqrt{i^2+j^2}\cdot h'\ell\right)^{-\alpha} \\
&= 4\cdot\left(\sum_{i=1}^{\infty} P(ih'\ell)^{-\alpha} + \sum_{i=1}^{\infty}\sum_{j=1}^{\infty} P(\sqrt{i^2+j^2})^{-\alpha}\cdot(h'\ell)^{-\alpha}\right) \\
&= 4P(h'\ell)^{-\alpha}\cdot\left(\sum_{i=1}^{\infty} i^{-\alpha} + \sum_{i=1}^{\infty}\sum_{j=1}^{\infty}(\sqrt{i^2+j^2})^{-\alpha}\right) \\
&= 4P(h'\ell)^{-\alpha}\cdot\left(1+\sum_{i=2}^{\infty} i^{-\alpha} + 2^{-\frac{\alpha}{2}} + 2\sum_{i=2}^{\infty}(\sqrt{1+i^2})^{-\alpha}\right. \\
&\quad \left. + \sum_{i=2}^{\infty}\sum_{j=2}^{\infty}(\sqrt{i^2+j^2})^{-\alpha}\right) \\
&\le 4Ph'\ell^{-\alpha}\left(\frac{\alpha(1+2^{-\frac{\alpha}{2}})}{\alpha-1} + \frac{\pi 2^{-\alpha/2}}{2(\alpha-2)}\right) = 4Ph'\ell^{-\alpha}\tau
\end{aligned}
$$

Here $\tau = \frac{\alpha\cdot(1+2^{-\frac{\alpha}{2}})}{\alpha-1} + \frac{\pi\cdot 2^{-\frac{\alpha}{2}}}{2(\alpha-2)}$. Thus the SINR at receiver $v$ is at least $\frac{P\cdot(\sqrt{2}\ell)^{-\alpha}}{N_0+\mathcal{P}} \ge \frac{P\cdot(\sqrt{2}\ell)^{-\alpha}}{N_0+4\tau P\cdot(h'\cdot\ell)^{-\alpha}} \ge \beta$. Then $h' \ge (\frac{4\beta\tau P\cdot\ell^{-\alpha}}{(\sqrt{2})^{-\alpha}P\ell^{-\alpha}-\beta N_0})^{\frac{1}{\alpha}}$, So $K = \lceil(\frac{4\beta\tau P\cdot\ell^{-\alpha}}{(\sqrt{2})^{-\alpha}P\cdot\ell^{-\alpha}-\beta N_0})^{\frac{1}{\alpha}} + 1 + \sqrt{2}\rceil$ is a lower bound of $h$. ■

### PROOF OF LEMMA 9.

*Proof:* Assume by contradiction that there are $\omega'$ senders transmitting simultaneously which are all neighbors of a single node. Here $\omega' > \omega$, thus $\omega' \ge \frac{(\delta\mathbf{r})^{\alpha}}{\beta}$. Consider the sender $s_i$ whose incident link $l_i$ is the shortest among all links of the senders, we assume its corresponding receiver is $r_i$. Then the interference experienced by node $r_i$ is at least $SINR_{S_{\omega'}}(l_i) = \frac{\frac{P}{\|l_i\|^{\alpha}}}{\sum_{s_j\in S_{\omega'}}\frac{P}{\|s_j r_i\|^{\alpha}}+N_0} \le \frac{P}{\sum_{s_j\in S_{\omega'}}\frac{P}{\|s_j r_i\|^{\alpha}}+N_0} < \frac{P}{\sum_{s_j\in S_{\omega'}}\frac{P}{\|s_j r_i\|^{\alpha}}} \le \frac{1}{\sum_{s_j\in S_{\omega'}}\frac{1}{\|s_j r_i\|^{\alpha}}} \le \frac{1}{\omega'\cdot\frac{1}{(\delta\mathbf{r})^{\alpha}}} \le \beta$. This contradicts the fact that $l_i$ can transmit without interference. ■