Minimum latency data aggregation in the physical interference model

Min Kyung An, Nhat X. Lam, Dung T. Huynh *, Trac N. Nguyen

Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080, United States

A R T I C L E   I N F O

Article history:
Received 17 February 2012
Received in revised form 21 August 2012
Accepted 24 August 2012
Available online 4 September 2012

Keywords:
Data aggregation
Inapproximability
Signal-to-interference-noise-ratio
Approximation algorithm

A B S T R A C T

Data aggregation has been the focus of many researchers as one of the most important applications in Wireless Sensor Networks. A main issue of data aggregation is how to construct efficient schedules by which data can be aggregated without any interference. The problem of constructing minimum latency data aggregation schedules (MLAS) has been extensively studied in the literature although most of existing works use the graph-based interference model.

In this paper, we study the MLAS problem in the more realistic physical model known as signal-to-interference-noise-ratio (SINR). We first derive an \(\Omega(\log n)\) approximation lower bound for the MLAS problem in the metric SINR model. We also prove the NP-hardness of the decision version of MLAS in the geometric SINR model. This is a significant contribution as these results have not been obtained before for the SINR model. In addition, we propose two constant factor approximation algorithms whose latency is bounded by \(O(D + R)\) for the dual power model, where \(D\) is the maximum node degree of a network and \(R\) is the network radius. Finally we study the performance of the algorithms through simulation.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

In Wireless Sensor Networks (WSNs), a number of small-sized sensor nodes monitor nearby environmental condition and collect data using their radio signals. The collected data is forwarded to a destination called a sink. This type of application is called data aggregation, and it is one of the most crucial applications of WSNs.

Despite the remarkable improvement over traditional sensing technology, the sensor nodes still have limited resources. Therefore, reducing energy consumption to prolong the lifetime of the network is an important issue in WSN research. In order to avoid the unnecessary retransmission using sensors’ limited power, researchers have focused on assigning timeslots to sensor nodes to obtain a good schedule. Due to the periodical occurrence of data collection, reducing the latency of the schedules, that is, constructing schedules with minimum number of timeslots, has been widely investigated.

The problem of constructing minimum latency data aggregation schedules in the graph-based interference model has been studied by many researchers. In the collision-free model, [1] proposed a heuristic algorithm that constructs a tree with a schedule. It assigns a code and a timeslot to each node to communicate with its parent node. Later, [2] introduced a nearly constant factor approximation algorithm whose latency is bounded by \(23R + D - 18\), where \(R\) is the network radius and \(D\) is the maximum node degree of the network. And [3] proposed a \(\Delta\)-approximation algorithm, and showed NP-hardness for the grid topologies. Yu et al. [4] and Xu et al. [5] also proposed constant factor approximation algorithms whose latency is bounded by \(24D + 6D + 16\) and \(16D + D - 14\), respectively. In [6], three algorithms were proposed with latency bounds, \(15D + D - 4, 2R + O(\log R) + D\) and \(1 + O\left(\frac{\log R}{R}\right)\). While only collision was considered in those papers, [6,7] focused on interference as well. In the collision-interference-free model, given a transmission power level \(p(u)\) for each node \(u\), the interference range is defined as \(\rho \cdot p(u)\), where \(\rho > 1\) is the interference factor. Wan et al. [6] and An et al. [7] proposed constant factor approximation algorithms whose latency is bounded by \(O(D + R)\), and in [7], the authors also proved an \(\Omega(\log n)\) approximation lower bound in the metric model. Xu et al. [8] studied lower bounds for data aggregation latency for the cases \(1 < \rho < 3\) and \(\rho > 3\) in the uniform power model, and obtained latency bounds max\(\frac{D}{\rho} + R\) and max\(\Delta + R\), respectively, where \(\phi = \frac{2R}{\lfloor \text{access} \rfloor}\).

Recently, several researchers have started investigating data aggregation scheduling in the more realistic physical interference model known as signal-to-interference-noise-ratio (SINR). Unlike the graph-based interference model, in the SINR interference model, the signal sent by a node \(u\) fades, and background noise

---

* This paper is a revised and expanded version of a paper entitled “Minimum latency data aggregation in the physical interference model” presented at the 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011), Miami Beach, FL, USA, October 31 – November 4, 2011.

* Corresponding author. Tel.: +1 972 883 2169.

* E-mail addresses: mka081000@utdallas.edu (M.K. An), nxl081000@utdallas.edu (N.X. Lam), huynh@utdallas.edu (D.T. Huynh), nguyentn07@yahoo.com (T.N. Nguyen).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.08.015
and signals sent concurrently by other nodes can interfere with u's signal. Thus, the signal sent to node u may not be strong enough to be received and, hence, transmitted data is lost. In the SINR model, only if the SINR is beyond a certain threshold, the transmitted data can be successfully received at the receiver node.

The first investigation of the problem of constructing minimum latency data aggregation schedules in the SINR model was done by Li et al. [9]. Li et al. [9] introduced a scheduling algorithm whose latency is bounded by $O(\lambda + R)$ which is asymptotically optimal. Later, [10] proposed a distributed algorithm that yields $O(g)$ timeslots, where $g$ is the logarithm of the ratio between the lengths of the longest and shortest links in a network. They also proposed a centralized algorithm whose latency is $O(\log^2 n)$ which was improved by [11] to $O(\log n)$. Recently, [12] introduced a distributed algorithm that computes schedules whose latency is bounded by $O(R + \Delta \log n)$. Note that [1–9] assumed that all nodes are initially assigned a uniform power level, and [10–12] assumed that the transmission power of each node is large enough to cover the maximum node distance in the network.

While these studies were concerned with data aggregation, some other researchers focused on related problems including scheduling [13–17], broadcasting [18] and gossiping [19,20] in the SINR model. Goussevskaia et al. [14] and An et al. [20] were the first studies of NP-hardness of scheduling and gossiping, respectively, in the geometric SINR model. The NP-hardness of the broadcasting problem is, to the best of our knowledge, still open.

In this paper, we continue the study of minimum latency data aggregation scheduling (MLAS) in the SINR model. We show that there is no algorithm having an approximation ratio better than $O(\log n)$ in the metric SINR model. In addition, we prove the NP-hardness of the MLAS problem in the geometric SINR model. Additionally, based on an algorithm proposed by [9], we also introduce two constant factor approximation algorithms yielding a latency bounded by $O(\lambda + R)$ for geometric MLAS in the dual power model. While existing works studied the problem with a uniform power assignment [1–9] or allowing unlimited power levels [10–12], our algorithms can be adapted to the more realistic case where multiple power levels are present, and the maximum power level is bounded.

This paper is organized as follows. In Section 2, we describe our network models and introduce the definitions used in this paper. In Section 3, we show the $O(\log n)$ approximation lower bound for MLAS problem in the metric model, and derive the NP-hardness of the MLAS problem in the geometric model. In Section 4, we introduce two constant factor approximation algorithms for the MLAS problem in dual power model and analyze them in Section 5. In Section 6, we compare the performance of these algorithms whereas Section 7 contains some concluding remarks.

### 2. Network models

In this paper, a wireless network consists of a number of arbitrarily distributed sensor nodes. Each node is equipped with a radio frequency transceiver that sends and receives data. We model such a network as $(V, D, p)$, where $V$ is the set of $n$ sensor nodes, $D : V \times V \rightarrow R^+$ represents the distance function between nodes in $V$ and $p : V \rightarrow \{p_1, p_2, \ldots, p_k\}$ is the power assignment function that assigns each node a power level $p(v) \in \{p_1, p_2, \ldots, p_k\}$.

Considering a communication link $(s, r)$ in this network, where $s$ is a sender and $r$ a receiver, let $X$ be the set of other concurrently transmitting links. Adopting the SINR interference model [21], we define the SINR value of the link $(s, r)$ as follows:

$$\text{SINR}(s, r) = \frac{p(s)D(s, r)^{-2}}{N + \sum_{u \neq s} p(u)D(u, r)^{-2}}$$

where $N > 0$ is the ambient noise and $2 < z < 6$ is the path loss exponent. The receiver $r$ can successfully receive the signal from the sender $s$ if and only if the SINR $(s, r)$ value exceeds a given threshold $\beta \geq 1$. Thus, a node $u$ with power $p(u)$ can send signals to only nodes that are within a distance $d \leq \left(\frac{\alpha \log n}{p(u)}\right)^{1/z}$. We call these nodes $u$’s neighbors. Implicitly, the network communication is modeled as a directed graph $G(V, E)$, where $E = \left\{ (s \rightarrow r) | s, r \in V, D(s, r) \leq \left(\frac{\alpha \log n}{p(u)}\right)^{1/z} \right\}$.

The minimum aggregation scheduling problem is defined as follows. A schedule is defined to be a sequence of timeslots, at each of which, several nodes are scheduled to send its aggregated data to one of its neighbors, and every node can be scheduled as a sender only once. Formally, at each timeslot $t$, we have an assignment vector $\pi_t = (l_1, l_2, \ldots, l_n)$, in which $l_i$ is a directed link from $s_i$ to $r_i$, satisfying SINR threshold inequalities, i.e., all senders $s_i$ can transmit concurrently. And a schedule, as a sequence of assignment vectors, is denoted by $\Pi = (\pi_1, \pi_2, \ldots, \pi_t)$, where $M$, the length of the schedule, is also called its latency.

Table 1 shows the notations used in this section.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G$</td>
<td>A directed graph</td>
</tr>
<tr>
<td>$V$</td>
<td>A set of nodes</td>
</tr>
<tr>
<td>$C$</td>
<td>A set of source nodes</td>
</tr>
<tr>
<td>$E$</td>
<td>A set of communication links</td>
</tr>
<tr>
<td>$D$</td>
<td>The distance function</td>
</tr>
<tr>
<td>$D(s, r)$</td>
<td>The distance between two nodes, $s$ and $r$</td>
</tr>
<tr>
<td>$n$</td>
<td>The number of nodes</td>
</tr>
<tr>
<td>$k$</td>
<td>The number of power levels</td>
</tr>
<tr>
<td>$p$</td>
<td>A power assignment function</td>
</tr>
<tr>
<td>$p(v)$</td>
<td>A power level of a node $v$</td>
</tr>
<tr>
<td>$(l, r)$</td>
<td>A communication link from a sender $l$ to a receiver $r$</td>
</tr>
<tr>
<td>$X$</td>
<td>A set of concurrently active links</td>
</tr>
<tr>
<td>$\text{SINR}(s, r)$</td>
<td>The SINR value of the link $(l, r)$</td>
</tr>
<tr>
<td>$N$</td>
<td>The background noise</td>
</tr>
<tr>
<td>$x$</td>
<td>The path loss exponent</td>
</tr>
<tr>
<td>$\beta$</td>
<td>The SINR threshold</td>
</tr>
<tr>
<td>$t$</td>
<td>A timeslot</td>
</tr>
<tr>
<td>$\pi_t$</td>
<td>An assignment vector at the timeslot $t$</td>
</tr>
<tr>
<td>$\Pi$</td>
<td>A schedule</td>
</tr>
<tr>
<td>$M$</td>
<td>The length of a schedule, i.e., the latency</td>
</tr>
</tbody>
</table>

| Table 1 Notations used in Section 2 |

Minimum Latency Aggregation Scheduling (MLAS): Given a set $C \subset V$ of source nodes and a sink node $s \in V \setminus C$, the objective of the data aggregation scheduling problem is to find the minimum latency schedule to aggregate data from all source nodes in $C$ to the sink node $s$.

Metric Model: In the metric version of the problem, we assume sensor nodes are located in a metric space $(V, D, p)$, where $D$ satisfies the triangle inequality.

Geometric Model: The geometric model is a special case of the metric model, where sensor nodes are deployed in the plane and $D$ is defined as the Euclidean distance between nodes.

### 3. Complexity of minimum latency aggregation scheduling

#### 3.1. Inapproximability of metric MLAS

In this section, we prove an $\Omega(\log n)$ approximation lower bound for the MLAS problem in the metric model. The result even holds true in the uniform power model ($k = 1$).

**Theorem 1.** There is no approximation algorithm having an approximation ratio better than $\Omega(\log n)$ for the MLAS problem in the metric model unless $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$. 

Proof. We construct a polynomial-time approximation-preserving reduction from the Set cover problem that is known to be hard to approximate [22,23]. Let $I_1$ be an instance of the Set cover problem consisting of a collection $S$ of subsets of a finite set $E$ of elements. Let $n$ and $m$ denote the cardinalities of $E$ and $S$, respectively. A solution to $I_1$ is a subset $S' \subseteq S$ such that every element $e \in E$ is in at least one set $A \subseteq S'$. Let $OPT_1$ denote an optimal solution to $I_1$.

Given the instance $I_1$ of the Set cover problem, we construct in polynomial time an instance $I_M$ of the MLAS problem as follows. $I_M$ consists of $n + mn + n$ nodes. The set of nodes $V$ in $I_M$ is partitioned into three subsets of nodes. The first subset contains only a single node $s_0$ as the sink node. The second subset $C_S$ and the third subset $C_E$ consist of $mn$ and $n$ nodes, respectively. All nodes in $C_E$ are source nodes (see Fig. 1).

The set $C_S$ is broken down into $n$ layers of nodes. Each layer $k$ contains $m$ nodes, namely $s_{k,1}^1, s_{k,1}^2, \ldots, s_{k,m}^1$. Consider the first layer of nodes in this set denoted by $C_S$. These $m$ nodes in $C_S$ correspond to the $m$ sets in the collection $S$. For each node $a \in C_S$, let $S(a)$ be the set in $S$ that $a$ represents. Similarly, the $n$ nodes in $C_E$ correspond to the $n$ elements in $E$ of $I_1$. And $e(a)$ denotes the element in $E$ that corresponds to node $a$ in $C_E$.

We now define the distance function $D$ on $V \times V$ for the metric model of the MLAS problem. Let $\tau > 0$ be some constant, and define the distance between nodes $u, v \in V$ as follows:

$$D(u, v) = \begin{cases} \tau & u = s_{k,i}, \quad v = s_{k+1,i}, \quad 0 \leq k < n \\ 2\tau & u = s_{k,i}, \quad v = s_{k,j}, \quad 1 \leq i < j \leq m, \quad 0 \leq k < n \\ 2\tau & u = s_{k,i}, \quad v = v_{C_E}(e(v) \in S(u)) \\ \tau & u \in C_S, \quad v = v_{C_E}(e(v) \in S(u)) \\ \tau & u \in C_E, \quad v = v_{C_E}(e(v) \in S(u)) \\ \tau & u \in C_S, \quad v = v_{C_E}(e(v) \in S(u)) \end{cases}$$

All other distances follow from symmetry or are induced by the shortest path metric. We define the transmission power for each node as $p = \beta n^\tau$. From this definition, observe that there does not exist any direct connection between nodes in $C_S$, nor between nodes in different columns of $C_S$. Similarly, there is no direct connection between any node $a$ in $C_E$ and nodes in $C_S$ whose corresponding sets do not contain $e(a)$.

It has been shown that the Set cover problem cannot be approximated with an approximation factor better than $O(\log n)$ unless $NP \subseteq DTIME(n^{O(\log n)})$ [22,23]. Let $OPT_M$ be the optimal solution to instance $I_M$. The following two implications will be shown in Lemma 2 and Lemma 3 below. For some constant $\tau > 2$:

$$|OPT_1| \leq v \Rightarrow |OPT_M| \leq n + n^v$$

$$|OPT_1| > \ln(n) \Rightarrow |OPT_M| > n \ln(n)^v$$

Thus, for the inapproximability factor $r$ the following holds:

$$r > \frac{n \ln(n)^v}{n + n^v} \geq \frac{\ln(n)^v}{v + 1} \geq \frac{\ln(n)^v}{2}$$

In addition, we have that $\ln|V| < \ln(n + mn + 1) < 2 \ln(n) + c$, for some constant $c$. Substituting this into the above inequality concludes the proof of the theorem. \(\square\)

In order to prove Lemmas 2 and 3, we need the following fact.

Fact 1. While a node in $V$ is transmitting, no other node can send data without any interference.

Proof. Consider a sender node $u$ with the transmission power $p = \beta n^\tau$. A node $v$ can successfully receive data from $u$ if $D(u, v) < \frac{\ln(n)^v}{\tau}$. Now observe that the distances between $u$ and its neighbors are at least $\tau$. Therefore, it cannot send data simultaneously with other nodes. Thus, the Fact 1 follows. \(\square\)

Lemma 2. If the size of $OPT_S$ is $\leq v$, then $L(OPT_M) \leq n + n^v$.

Proof. Let us consider a solution of $I_M$ constructed from $OPT_S$ of $I_S$ that is constructed as follows. Every node in $C_S$ sends to its neighbor whose corresponding set is in $OPT_S$. Then, these selected nodes send data to the sink $s_0$ through all intermediate nodes sequentially.

According to Fact 1, the length of this schedule is $n|OPT_S| + n$. So, if $|OPT_S| \leq v$, we also have $L(OPT_M) \leq n^v + n$. \(\square\)

Lemma 3. If the size of $OPT_S$ is $> \ln(n)^v$, then $L(OPT_M) > n \ln(n)^v$.

Proof. Observe that nodes in the first layer of $C_S$ can only send their data to the sink node sequentially through the nodes in their column. And for each starting node in the first layer of $C_S$, it requires at least $n$ steps to reach the sink node. Thus, if the size of $OPT_S$ is $> \ln(n)^v$, then at least $\ln(n)^v$ such nodes in the first layer of $C_S$ are needed to collect data from all nodes in $C_E$. $L(OPT_M)$ is $> n \ln(n)^v$. \(\square\)

3.2. NP-hardness of geometric MLAS

In this section, we prove the NP-hardness of the MLAS decision problem in the geometric model for a special case, where the objective is to aggregate data from all nodes to the sink. Thus, the general version of the problem is also NP-hard.

In order to prove its NP-hardness, we construct a polynomial time reduction from the Partition problem which was proven NP-complete [24]. This decision problem is defined as follows. Given a finite set of distinct and positive integers, the objective is to determine if it is possible to divide this set into two subsets such that the sums of all integers in each subset are equal.

Let $I_P$ be an instance of the Partition problem consisting of a set $S$ of $n$ distinct and positive integers $a_1, a_2, \ldots, a_n$. Without loss of generality, assume that $a_1 < a_2 < \cdots < a_n$. We construct in polynomial time an instance $I_M$ of the geometric MLAS problem as follows (See Fig. 2).

$I_M$ consists of $2n + 3$ nodes which include $2n$ nodes $s_i$ and $r_i$, 2 nodes $s_{n+1}$ and $s_{n+2}$ and a sink node $a$. Letting $P$ be some constant, we deploy these nodes on the plane at the following positions:
Finally, we see that all other nodes have a distance at least \( d_{\text{min}} \) away from \( s_i \). Thus, \( s_i \) cannot send data to any such node \( u \) since \( P_d(s_i, u)^{-2} \leq P_d(s_i)^{-2} = \frac{b_i^2}{N \beta} < N \beta \).

**Fact 4.** \( s_{n+1} \) and \( s_{n+2} \) can send data only to \( s \).

**Proof.** Observe that \( s_{n+1} \) (or \( s_{n+2} \)) can send data to \( s \) since \( P_d(s_{n+1}, s)^{-2} = \frac{N \beta (A^2 \beta + 24)}{b_{\text{min}}} > N \beta \). But \( s_{n+1} \) (or \( s_{n+2} \)) cannot send data to any node \( s_j \) since \( P_d(s_{n+1}, s_j)^{-2} \leq P_d(s, s_j)^{-2} < N \beta \).

Next we claim that \( s_{n+1} \) and \( s_{n+2} \) cannot communicate with each other. This holds true because \( P_d(s_{n+1}, s_{n+2})^{-2} = \frac{N \beta (A^2 - 1)}{b_{\text{min}}} < \frac{N \beta (A^2 - 1)}{N \beta} = N \beta \).

**Fact 5.** When a node \( r_i \) is sending to \( r_{i+1} \) (or \( s_r \)), no other node can send without interference.

**Proof.** Assume that there exists a node \( s \) sending data at the same time with the link \((r_i, r_{i+1})\) and this node causes an interference \( I_u > 0 \) at \( r_{i+1} \). We have \( SINR(r_i, r_{i+1}) = \frac{P(r_i) \beta}{P(r_{i+1}) \beta + I_u} < \frac{P(r_i) \beta}{P(r_i) \beta} = \beta \). This means that \( r_i \) cannot successfully send data to \( r_{i+1} \).

**Fact 6.** It holds for all \( 1 \leq i < n \) that \( r_i \) can aggregate data to \( s \) through \( r_{i+1} \) only.

**Proof.** Due to the power of \( r_i \), it is obvious that \( r_i \) cannot send data to any nodes \( s_j \) and \( r_j \) for all \( j > i + 1 \), nor to node \( s_{n+1} \). In addition, Fact 3 shows that \( s_i \) can send data to \( r_i \) only. Although, \( r_i \) may be able to communicate with nodes on its right-hand side, in order to reach \( s \) on its left-hand side it must send data to \( r_{i+1} \).

**Lemma 4.** \( I_u \) has a solution if and only if \( I_u \) has a \( n+2 \) latency aggregation schedule.

**Proof.** First, we will show that every aggregation schedule of \( I_u \) requires at least \( n+2 \) timeslots. To this end, observe that due to Fact 5 and Fact 6, every solution to \( I_u \) must contain \( n \) timeslots, in each of which, \( r_i \) is scheduled to send its data to \( r_{i+1} \). Moreover, due to Fact 4 and Fact 5, every solution must contain 2 additional timeslots in which \( s_{n+1} \) and \( s_{n+2} \) send their data to the sink. Thus, the lower bound \( n+2 \) follows. We now return to the proof of Lemma 4. For the only-if direction, assume that \((S_i, S_j)\) is a partition of \( S \) where \( \sum_{a_i \in S_j} a_i = \sum_{a_i \in S_i} a_i \). We show how to construct an \((n+2)\)-latency schedule to aggregate data from all nodes to \( s \). Together with the \( n+2 \) lower bound, it follows that this schedule is an optimal solution to \( I_u \).

In the first timeslot, we schedule all nodes \( s_i \) that correspond to integers \( a_i \) in \( S_1 \), to send data to \( r_i \). Also in this timeslot, we schedule \( s_{n+1} \) to send data to \( s \).

All remaining nodes \( s_j \), that correspond to integers \( a_i \) in \( S_2 \), are assigned to send data to \( r_j \) in the second timeslot. And also in this timeslot, \( s_{n+2} \) is scheduled to send its data to \( s \).

The \( i \)th timeslot of next \( n \) timeslots is used for \( r_i \) to send its data to \( r_{i+1} \), where \( r_{n+1} \equiv s \). These \( n \) timeslots do not have any interference since each contains only one link. Thus, we only need to consider the first two timeslots. Due to Fact 2, it is guaranteed that the \( SINR \) value at all receivers except \( s \) exceeds \( \beta \). Therefore, we only need to check the \( SINR \) value at \( s \) in these two timeslots.
Since the analyses of these two timeslots are similar, we just consider the first one. We have
\[
\text{SINR}(s) = \frac{N(\beta^2/24)}{N + \sum_{a \in c(s)} P_d(s, a)} = \frac{N(\beta^2/24)}{N + \sum_{c(s)} a_i} = \beta
\]
This means that all links in the first timeslot can successfully transmit their data.

For the if direction, we show that if \( \text{IM} \) has an \( n+2 \) latency aggregation schedule, then \( \text{lp} \) has a solution. To this end, observe that due to the \( n+2 \) lower bound and Fact 5, in order to obtain a \((n+2)\)-latency schedule, all nodes \( s_i \) must be scheduled with either \( s_{i+1} \) or \( s_{i+2} \). This means that we have to split the set of all nodes \( s_i \) into two different timeslots such that in each timeslot the interference experienced at \( s \) is exactly \( \frac{\beta}{24} \). It is straightforward to see that the partition corresponding to this schedule yields a solution to \( \text{lp} \). \( \square \)

**Theorem 5.** The MLAS decision problem in the geometric model is NP-hard.

**Proof.** In Lemma 4, we have proved that Partition is polynomial-time reducible to MLAS, and therefore it is NP-hard. \( \square \)

4. Constant factor approximation algorithms

In this section, we introduce two constant factor approximation algorithms for the geometric MLAS problem in the dual power model \((k=2)\).

In practice, the links \((s, r)\), where \( d(s, r) = \left(\frac{p(s)}{N}\right)^{1/2} \), are not good candidates as only node \( s \) can be a sender to send its data to its receiver node \( r \); other nodes cannot transmit concurrently. Thus we are more interested in links \((s, r)\), where the Euclidean distance between \( s \) and \( r \), \( d(s, r) < \left(\frac{\min\{p(s), p(r)\}}{N}\right)^{1/2} \), for some constant \( \gamma > 1 \).

In designing the constant factor approximation algorithms for MLAS, we assume that the undirected graph \( G(V, E) \) where
\[
E = \left\{ (s, r) | d(s, r) = \left(\frac{\min\{p(s), p(r)\}}{N}\right)^{1/2} \right\},
\]
is connected. We further make the commonly accepted assumption that \( \alpha > 2 \).

If the dual power model, each node is assigned the low power level \( p_0 \) or the high power level \( p_0 \), where \( \delta > 1 \). We use \( p_0 \) to denote a node that is assigned power level \( p_0 \), also called high power node, and \( c(t) \) to denote a node that is assigned power level \( p_0 \), also called low power node. We use \( d_{in} = \left(\frac{p_0}{N}\right)^{1/2} \) and \( d_{in} = \left(\frac{p_0}{N}\right)^{1/2} \) to denote the maximum link lengths in the graph \( G(V, E) \) generated by \( p_0 \)'s and \( p_0 \)'s, respectively.

Table 2 shows the notations used in Sections 4 and 5.

4.1. Data aggregation tree construction

Our algorithms schedule the nodes based on a data aggregation tree on \( G \). Before presenting the scheduling algorithms, we first show how to obtain a data aggregation tree on \( G \).

We construct a data aggregation tree on \( G \) as done by [9].

Choose a center node \( c \), and then construct a breadth first search (BFS) tree (cf. [25]) on \( G \) rooted at node \( c \) in order to get a latency bound in terms of the network radius \( R \) rather than its diameter. (A node \( v \) can be chosen as the center node if the distance from \( v \) to the most distant node from \( v \) is minimum.) Once the data is aggregated to the center node \( c \), it sends the aggregated data to the sink node via a shortest path. This takes only \( O(R) \) timeslots. Based on the BFS tree, we compute a Maximal Independent Set (MIS) on \( G \) using an algorithm in [26]. (A subset \( V \subseteq V \) of a graph \( G = (V, E) \) is said to be independent if for any vertices \( u, v \in V \), \( u, v \notin E \). An independent set is said to be maximal if it is not a proper subset of another independent set.) We call the nodes in an MIS dominators, and the others dominated. The MIS constructed by [26] guarantees that the distance between any pair of its complementary subsets is exactly two hops. To obtain a Connected Dominating Set (CDS) of \( G \), [9] connects the dominators using some connectors that were originally dominated. (A dominating set (DS) is a subset \( V \subseteq V \) such that every vertex \( v \) is either in \( V \) or adjacent to a vertex in \( V \). A DS is said to be connected (CDS) if it induces a connected subgraph.) Since there may exist dominatees that are not connected to the CDS, we connect each of the remaining dominatees to its neighboring dominator that has the smallest hop-distance to the root of the BFS tree. We denote the newly formed tree by \( T_{Agg} \), and use it as the data aggregation tree in our algorithms.

4.2. Data aggregation scheduling algorithms

4.2.1. Square-based aggregation scheduling algorithm

The first algorithm called the Square-Based Aggregation Scheduling (SBAS) algorithm is an extension of [9]'s to the dual power model. SBAS starts by partitioning a network into square cells each of which has diagonal lengths \( d_i \) or \( d_6 \). This induces two grids where the upper-left corners have coordinates \((1, 1)\). Let us denote the squares whose diagonal length is \( d_0 \) by \( HS \) (HighSquare), and the squares whose diagonal length is \( d_6 \) by \( LS \) (LowSquare). Each HS (LS) cell is denoted by a Cell-ID \( HS(i, j) \) (\( LS(l, m) \)). If its upper-left corner has coordinates \((i, j) \) \((l, m)\).
The SBAS algorithm schedules the nodes in $T_{Agg}$ starting with the dominants so that they can send data successfully to their upper level dominators. While scheduling the dominants, high power dominators are considered first followed by low power dominators. After all dominators are scheduled, only dominators and connectors are scheduled. (Steps 8 – 14 in Algorithm 1). At the second iteration, all the leaves to be scheduled are dominators (e.g., the nodes $v_1, v_2, v_3$ and $v_4$ in Fig. 3(b)), and these dominators are scheduled to send their data to their upper level dominators. (Steps 9 – 13). At the second iteration all the leaves to be scheduled are dominators (e.g., the nodes $v_5, v_6, v_7$ and $v_8$ in Fig. 3(c)), and they are scheduled to send their data to their upper level dominators. (Steps 9 – 13 in Algorithm 1). These two iterations are repeated until all dominators and connectors are scheduled. (Steps 8 – 14 in Algorithm 1). At each iteration, high power nodes are scheduled first followed by low power nodes. While scheduling (Steps 4 – 12 in Algorithm 1), SBAS uses Algorithm 2 as a subroutine to assign the same timeslots to high (low) power nodes if they are $[H_{pl}]$ ($[H_{ph}]$) cells away from each other. Once all data is aggregated to the center node $c$, $c$ sends the aggregated data to the sink node $s$ via a shortest path (Step 15 in Algorithm 1).

4.2.2. Hexagon-based aggregation scheduling algorithm

We first introduce a new hexagonal tessellation technique in which each hexagon has a diagonal length $d$. Once a network is partitioned into hexagons, the hexagons are labeled with $l$-labelings, where $l$ is a positive integer. Fig. 4(a) shows a 1-labeling, and we get a $l$-labeling by enclosing the 1-labeling with a layer of hexagons as shown in Fig. 4(b). Similarly, we obtain a 19-labeling shown in Fig. 4(c), and recursively we obtain in general $(3l^2 - 3l + 1)$-labelings. Fig. 4(d) shows an example of tessellating a network with hexagons using 19-labeling.

The second algorithm named the Hexagon-Based Aggregation Scheduling (HBAS) algorithm starts by partitioning into hexagons of cell which has diagonal length $d$. Let us denote the hexagons whose diagonal length is $d$, by $HX$ (HighHexagon), and the hexagons whose diagonal length is $d$, by $LX$ (LowHexagon). Once a network is partitioned into hexagons, $HX$ cells are labeled using $l$-labeling, where

$$L_0 = 3L_{ph}^2 - 3L_{ph} + 1, l_0 = \left\lceil \frac{16(\tau_s/d_h + 1)^2 - 3j^2 + 7}{6} \right\rceil,$$

and $\tau_h = \left(\frac{2\pi}{\sqrt{|h|+1}}\right)^2$. Similarly, it labels the $LX$ cells using $(3L_{pl}^2 - 3L_{pl} + 1)$-labeling, where

$$L_0 = 3L_{pl}^2 - 3L_{pl} + 1, l_0 = \left\lceil \frac{16(\tau_s/d_i + 1)^2 - 3j^2 + 7}{6} \right\rceil.$$
and $\tau_i = \left(\frac{p_{i,2\pi}}{N_{i,1}\pi} \right)^{\frac{1}{2}}$.

Then, a number of iterations are performed to find a schedule on the data aggregation tree obtained. The scheduling is based on the labels of $HX$ ($LX$) cells in the $L_h$-labeling ($L_l$-labeling), where $L_h$ ($L_l$) is a constant value that guarantees that two high (low) power senders can transmit data successfully if they are located in the $HX$ ($LX$) cells with the same labels. The details are contained in Algorithms 3 and 4.

### Algorithm 3. HBAS Scheduling

**Input:** A set $V$ of nodes with two power levels $p_h$ and $p_l$  
**Output:** Length of schedule

1: Partition the network into $HX$ and $LX$ cells each of which has diagonal length $\left(\frac{p_{h,2\pi}}{N_{h,1}\pi} \right)^{\frac{1}{2}}$ and $\left(\frac{p_{l,2\pi}}{N_{l,1}\pi} \right)^{\frac{1}{2}}$, respectively, and label the $HX$ and $LX$ cells using $L_h$-labeling and $L_l$-labeling, respectively.
2: Construct $T_{agg}$ using the algorithm of [9].
3: Set the starting timeslot $t = 1$
4: Let $S_h \subseteq T_0$ be the set of high power dominantes, and $S_l \subseteq T_0$ be the set of low power dominantes.
5: $t \rightarrow HB-HighPower(S_h, t)$
6: $t \rightarrow HB-LowPower(S_l, t)$
7: $r \rightarrow 1$
8: **while** $T_r \neq \{c\}$ **do**
9: $Z_r \rightarrow$ leaves of $T_r \rightarrow \{c\}$
10: Let $S_h$ be the set of high power nodes in $Z_r$, and $S_l$ be the set of low power nodes in $Z_r$.
11: $t \rightarrow HB-HighPower(S_h, t)$
12: $t \rightarrow HB-LowPower(S_l, t)$
13: $r \rightarrow r + 1$
14: **end while**
15: Send the aggregated data from the center node $c$ to the sink node $s$ via a shortest path $f$.
16: **return** $(t - 1) \cdot \text{length of f}$

### Algorithm 4. HB-HighPower (HB-LowPower) Algorithm

**Input:** A set $S$ of high (low) power nodes and a starting timeslot $t$  
**Output:** Timeslot $t$

1: **while** $S \neq \emptyset$ **do**
2: Pick one high (low) power node $v \in S$ as a sender in each $HX$ ($LX$) cell. Let $S' \subseteq S$ be the set of such nodes.
3: **for** $i = 1$ to $L_h$ ($L_l$) **do**
4: Let $Z \subseteq S'$ be the set of nodes whose $HL$ is $i$.
5: **for** each $w \in Z$ **do**
6: Assign timeslot $t$ to $w$.
7: **end for**
8: $t \rightarrow t + 1. S \leftarrow S' \setminus S', S' \leftarrow S \setminus Z$
9: **end for**
10: **end while**
11: **return** $t$

Similar to SBAS, the HBAS algorithm first schedules the dominantes (Steps 4 – 6 in Algorithm 3), and then several iterations are performed to schedule the remaining dominators and connectors (Steps 8 – 14 in Algorithm 3). At each step for scheduling dominantes, dominators or connectors, it also schedules first high power nodes, and then low power nodes as done in SBAS. While scheduling, HBAS uses Algorithm 4 as a subroutine to assign the same timeslots to high (low) power nodes which are located in the $HX$ ($LX$) cells with the same label. As the final step, the aggregated data at the center node $c$ is sent to the sink node $s$ via a shortest path (Step 15 in Algorithm 3).

### 5. Analysis of algorithms

In this section, we analyze the Square-Based Aggregation Scheduling (SBAS) and the Hexagon-Based Aggregation Scheduling (HBAS) algorithms (Algorithms 1 and 3), and bound the latency of the schedule produced by them.

#### 5.1. Analysis of SBAS

First, we analyze the SBAS (Algorithm 1) and bound the latency of the schedule it produces. We first prove that any two high (low) power nodes can send data without any interference if they are $[H_h \cup (H_l)] HS$ cells apart. Note that while scheduling, high power nodes and low power nodes are not scheduled to send data at the same time.

**Lemma 6.** For SINR threshold $\beta > 1$, path loss exponent $\alpha > 2$, background noise $N > 0$, and some constant $\gamma > 1$, let $H_b = \left(\frac{p_{h,2\pi}}{N_{h,1}\pi} \right)^{\frac{1}{2}} \cdot \sqrt{2 \cdot \left(\frac{N}{N_{h,1}}\right)} + 2$. Then any two high power nodes that are at least $[H_h \cup (H_l)] HS$ cells away from each other can send data simultaneously without interference.

**Proof.** We bound $H_b$ under the assumption that all nodes transmit with power level $p_h$ since high power nodes are not scheduled at the same time with low power nodes in the SBAS algorithm.

In order to bound the shortest distance between any two high power nodes that can send data simultaneously, we first consider the following worst case for a sender node $i$. As node $i$ is trying to send its data to a receiver node $j$, the possible longest distance between $i$ and $j$ is $\left(\frac{p_{h,2\pi}}{N_{h,1}\pi} \right)^{\frac{1}{2}}$. In order that the receiver $j$ receives data from the sender $i$ successfully, for all other sender nodes that are sending data simultaneously, the following must be satisfied:

$$\frac{p_h \cdot \left(\frac{p_{h,2\pi}}{N_{h,1}\pi} \right)^{\frac{1}{2}} \cdot \sum_{u \in [i,j]} d(u) \cdot \sum_{v \in [i,j]} d(v) - 2 \cdot \frac{p_h \cdot (N_{h,1})}{N_{h,1}^{\gamma - 1}} \int_0^{2\pi} 2\pi \cdot r \cdot r^{d - 2} dr}{N_{h,1}^{\gamma - 1}} \leq \beta$$

From inequality (1), we get

$$\frac{p_h \cdot (N_{h,1})}{N_{h,1}^{\gamma - 1}} \leq \frac{p_h \cdot 2\pi \cdot d^{d - 2}}{\alpha - 2}$$

Now, we bound the shortest distance $d$ satisfying the following inequality:
\[ \frac{p_h \cdot 2\pi}{N(y - 1)} \cdot \frac{d^2 - 3}{x - 2} \leq 1 \]

We have
\[ d^2 - 3 \leq \frac{N(y - 1)(x - 2)}{p_h \cdot 2\pi} \Rightarrow d \geq \left( \frac{p_h \cdot 2\pi}{N(y - 1)(x - 2)} \right)^{\frac{1}{3}} \]

Let \( \tau_h := \left( \frac{p_h \cdot 2\pi}{N(y - 1)(x - 2)} \right)^{\frac{1}{3}} \). Then \( \tau_h \) is a lower bound for \( d \).

Next, we bound the number of HS cells between \( j \) and another sender that is closest to \( j \), say \( i \), as follows. Assume a straight line between \( i \) and \( j \), and relay nodes with the power level \( p_h \) on the line. Let \( h \) be the number of HS cells between \( i \) and \( j \). As \( \tau_h < \left( \frac{p_h \cdot 2\pi}{N(y - 1)(x - 2)} \right)^{\frac{1}{3}} \), we have \( \tau_h \cdot \frac{\sqrt{2}}{\frac{\sqrt{\pi}}{p_h}} < h \). Therefore, the receiver and another sender should be at least \( \sqrt{h} \cdot \frac{\sqrt{\pi}}{p_h} \) HS cells apart, and any two senders should be at least
\[ H_h = \sqrt{h} \cdot \frac{\sqrt{\pi}}{p_h} \]

\( \sqrt{h} \) HS cells apart. Thus, if any two high power nodes that are at least \( |H_h| \) HS cells apart from each other, they can send data simultaneously without any interference. \( \square \)

Using a similar argument as in the proof of Lemma 6, we can set \( H_i \) as follows.

**Lemma 7.** Let \( H_i = \left( \frac{p_h \cdot 2\pi}{N(y - 1)(x - 2)} \right)^{\frac{1}{3}} \cdot \frac{\sqrt{\pi}}{p_h} + 2 \) for a SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \gamma > 1 \). Then any two low power nodes that are at least \( |H_i| \) LS cells apart can send data simultaneously without interference.

Next, we prove that the latency of a schedule found by SBAS is bounded by \( O(\Delta + R) \). We need the following lemmas.

**Lemma 8.** [9,27] Suppose that dominators \( v \) and \( w \) are within 2 hops from dominator \( u \), and \( v' \) and \( w' \) are the corresponding connectors for \( v \) and \( w \), respectively. \( p(v) = p(w) = p'(v') = p'(w') = p(u) = r \). If \( \angle vuw \leq 2 \arcsin \frac{1}{2} \), then either \( \frac{v}{w} \leq r \) or \( \frac{w}{v} \leq r \).

**Lemma 9.** The number of low power connectors in an LS cell is at most 12.

**Proof.** First, let \( \sec(v, \theta, d) \) denote a sector with an angle of \( \theta \) radian of a circle (broadcasting disk) centered at \( v \) with the radius of \( d \). Now, consider a dominator \( v \) in an LS cell and its low power connectors. Noting that the connectors connect dominators which are 2-hops away from \( v \) in CDS, we consider the following two cases:

- Case 1: \( v \) is a low power dominator
- Case 2: \( v \) is a high power dominator

In Case 1, consider the dominators located in the area of \( \sec(v, 2 \arcsin \frac{1}{2}, d_1 + d_2) - \sec(v, 2 \arcsin \frac{1}{2}, d_1) \). We can let the dominators share one connector by Lemma 8. As \( 13 - 2 \arcsin \frac{1}{2} > 2\pi \), \( v \) can have at most 12 such areas in each of which all dominators share one connector. In Case 2, consider the dominators located in the area of \( \sec(v, 2 \arcsin \frac{1}{2}, d_3 + d_4) - \sec(v, 2 \arcsin \frac{1}{2}, d_3) \). We still can apply Lemma 8 to let the dominators share one connector, as the area of \( \sec(v, 2 \arcsin \frac{1}{2}, d_3 + d_4) - \sec(v, 2 \arcsin \frac{1}{2}, d_3) \) is within the area of \( \sec(v, 2 \arcsin \frac{1}{2}, d_3 + d_4) - \sec(v, 2 \arcsin \frac{1}{2}, d_3) \). Thus, in both cases, we can conclude that the number of low power connectors in an LS cell is at most 12. \( \square \)

**Lemma 10.** The number of high power connectors in an HS cell is at most \( 4(\pi(2 + \frac{1}{2} + \frac{1}{2})) \).

**Proof.** First, let \( \sec(v, \theta, d) \) denote a sector with an angle of \( \theta \) radian of a circle (broadcasting disk) centered at \( v \) with the radius of \( d \). Considering a dominator \( v \) in an HS cell, let \( C \) be the set of \( v \)'s high power connectors. We further let \( C_h \subseteq C \) denote the set of high power connectors that connect high power dominators that are at most 2 hops away from \( v \), and \( C_l \subseteq C \) denote the set of high power connectors that connect low power dominators that are at most 2 hops away from \( v \).

Let us first bound \( |C_h| \), considering the following two cases:

- Case 1: \( v \) is a high power dominator
- Case 2: \( v \) is a low power dominator

In Case 1, following a similar argument in the proof of Lemma 9, \( |C_h| \) is bounded by 12. In Case 2, it is sufficient to bound the number of high power dominators that are at most 2 hops away from \( v \), i.e., the number of high power dominators located in \( \sec(v, 2\pi, d_3 + d_4) - \sec(v, 2\pi, d_4) \). Considering the area of \( \sec(v, \theta, d_3 + d_4) - \sec(v, \theta, d_4) \), where \( \theta = \frac{\theta}{\sqrt{\pi}} \), let us partition the area into 2 cells, \( c_1 \) and \( c_2 \), as in Fig. 5(a). In Fig. 5(a), \( \angle P_1P_2P_3 = \theta \).
where \( p_i, 0 \leq i < 2 \), represents a point. Then, the largest distance in each cell is \( d_1 \) [28], and hence there exist at most 1 high power dominator in each cell. This implies that there exist at most 2 high power connectors for the high dominators in the area of \( \sec(v, 0, d_1) \). Therefore, in Case 2, \( |C_a| \) is bounded by \( \lceil 2 \frac{2d_1}{C_1} \rceil = \left\lceil 4\pi(\delta + 1) \right\rceil \).

Next, let us bound \( |C_i| \) considering the following two cases:

- **Case 3**: \( v \) is a low power dominator
- **Case 4**: \( v \) is a high power dominator

In Case 3, following a similar observation in Case 1, \( |C_i| \) is bounded by 12. In Case 4, it is sufficient to bound the number of low power dominators that are at most 2 hops away from \( v_1 \), i.e., the number of low power dominators located in \( \sec(v, 2\pi, d_1) \). Considering the area of \( \sec(v, \theta, d_1) \), where \( \theta = \frac{\pi}{C_1} \), let us partition the area into 2 cells, \( c_3 \) and \( c_4 \), as in Fig. 5(b). In Fig. 5(b), \( c_p, c_p = \theta \), where \( p_i, 3 \leq i \leq 5 \), represents a point. Then, the largest distance in each cell is \( d_1 \) [28], and hence there exist at most 1 low power dominator in each cell. This implies that there exist at most 2 high power connectors for the low power dominators in the area of \( \sec(v, \theta, d_1) \). Therefore, in Case 4, \( v \) has at most \( \lceil 2 \frac{2d_1}{C_1} \rceil = \left\lceil 4\pi(\delta + 1) \right\rceil \) high power connectors.

Since \( |C_i| + |C_p| \leq \max\{12, \left\lceil 4\pi(\delta + 1) \right\rceil \} + \max\{12, \left\lceil 4\pi(\delta + 1) \right\rceil \} \), there are at most \( \left\lceil 4\pi(2 + \delta^2 + \delta) \right\rceil \) high power connectors in an HS cell.

**Lemma 11.** For any node at most \( \omega = 2^\delta \pi(\delta + 1) - 1 \) neighboring nodes can send data simultaneously in the dual power model.

**Proof.** Similar to the proof by [9], we prove this by contradiction. Let \( x \) be the length of the shortest link between any two of the neighbors. Assume that one of the end points of the link is a high power node \( v_i \), and it is sending its data to another end point \( v_j \). Let \( \mathcal{S} \) be the set of all the other concurrently-transmitting senders and assume that all nodes in \( \mathcal{S} \) are low power nodes so that the SINR of \( v_i \) is largest. Now suppose that \( |\mathcal{S}| = \alpha \omega \geq 2^\delta \pi/2 \). Then, the SINR of \( v_i \) is

\[
\text{SINR}(v_i) = \frac{p_i \cdot \delta}{N + \sum_{\eta \in \mathcal{S}} \frac{p_i \cdot \delta}{\alpha \pi}} \leq \frac{p_i \cdot \delta}{N + \sum_{\eta \in \mathcal{S}} \frac{p_i \cdot \delta}{\alpha \pi}} \leq \frac{p_i \cdot \delta}{\sum_{\eta \in \mathcal{S}} \frac{p_i \cdot \delta}{\alpha \pi}} \leq \frac{\delta}{\sum_{\eta \in \mathcal{S}} \frac{p_i \cdot \delta}{\alpha \pi}} < \frac{\delta}{\alpha \pi} \leq \delta \frac{2^\delta \pi(\delta + 1)}{\alpha \pi} \leq \beta
\]

Thus, \( v_i \) cannot send data without interference, a contradiction! □

**Corollary 12.** If \( \Delta \) be the maximum node degree in a network, then every data aggregation schedule has at least \( \max\{\Delta/\omega, R\} \) timeslots.

**Theorem 13.** SBAS produces data aggregation schedules whose latency is bounded by \( O(\Delta + R) \), and it is a constant factor approximation algorithm.

**Proof.** First consider the Steps 4 – 5 of Algorithm 1. There exist at most \( \Delta \) dominators in each HS and LS cell. Considering their dominator \( v_1 \), one of those \( \Delta \) dominators must be a connector to connect the dominator \( v_1 \) to another dominator. Therefore, the number of dominators in each HS and LS cell is bounded by \( \Delta - 1 \). Thus, data from all high power dominators can be aggregated to their upper level dominators in at most \( [H_1 + 1]^2(\Delta - 1) \) timeslots, and data from all low power dominators in at most \( [H_1 + 1]^2(\Delta - 1) \) timeslots. Therefore, Steps 4 – 5 take at most \( \left([H_1 + 1]^2 + [H_1 + 1]^2\right)(\Delta - 1) \) timeslots.

Next consider Steps 8 – 14 in Algorithm 1. At the first iteration all the leaves to be scheduled are dominators, and at the second iteration all the leaves to be scheduled are connectors. These two iterations are repeated until all dominators and connectors are scheduled. Consider the iterations for scheduling dominators. Since there exists at most 1 high power dominator in each HS cell and at most 1 low power dominator in each LS cell, data from dominators can be aggregated to their upper level dominators in at most \( [H_1 + 1]^2 + [H_1 + 1]^2 \) timeslots at each iteration. As this process is repeated at most \( \frac{R}{\omega} \) times, it takes at most \( \frac{R}{\omega} \left(H_1 + 1\right)^2 + \left[H_1 + 1\right]^2 \) timeslots for scheduling all dominators. Now, let us consider the iterations for scheduling connectors. Since there exist at most \( \left[4\pi(2 + \delta^2 + \delta)\right] \) high power connectors in each HS cell (Lemma 10) and at most \( 12 \) low power connectors in each LS cell (Lemma 9), data from connectors can be aggregated to their upper level dominators in at most \( \left[4\pi(2 + \delta^2 + \delta)\right] \cdot [H_1 + 1]^2 + 12[H_1 + 1]^2 \) timeslots at each iteration. As this process is repeated at most \( \frac{R}{\omega} \) times, it takes at most \( \frac{R}{\omega} \left(\left[4\pi(2 + \delta^2 + \delta)\right] \cdot [H_1 + 1]^2 + 12[H_1 + 1]^2 \right) \) timeslots for scheduling all connectors.

Finally, as Step 15 in Algorithm 1 takes at most \( R \) timeslots, the latency of CBAS is bounded by \( \left([H_1 + 1]^2 + [H_1 + 1]^2\right) \cdot (\Delta - 1) + \left([H_1 + 1]^2 + [H_1 + 1]^2\right) \cdot \frac{R}{\omega} \left([H_1 + 1]^2 + [H_1 + 1]^2\right) = R = O(\Delta + R) \). Thus, it is a constant factor approximation algorithm because any schedule requires \( \geq \max\{\Delta/\omega, R\} \) timeslots by Lemma 11 and Corollary 12. □

**5.2. Analysis of HBAS.**

In this section, we analyze the HBAS algorithm (Algorithm 3) and bound the latency of the schedule produced by the algorithm. We first show that any two high (low) power nodes which are located in the HS (LS) cells with the same label can send data without any interference. In order to show this, we need the following lemmas. Also note that while scheduling, high power nodes and low power nodes are not scheduled to send data at the same time as done in SBAS.

**Lemma 14.** For a SINR threshold \( \beta > 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \gamma > 1 \), let

\[
K_b = \left(\frac{p_h \cdot 2\pi}{N(\gamma - 1)(\alpha - 2)}\right)^{\gamma n} + \left(\frac{p_h}{\gamma N}\right)^{\frac{1}{\gamma}}
\]

Then any two high power nodes that are at least distance \( K_b \) apart from each other can concurrently send data successfully.

**Proof.** Let us bound the shortest distance \( K_b \) between any two high power nodes that can send data simultaneously. First consider the following worst case that a high power node \( i \) is trying to send its data to a receiver node \( j \) on a possible longest link whose distance is \( \left(\frac{p_h}{\pi}\right)^{\frac{1}{2}} \). Letting \( i \) be another sender that is closest to \( j \), from the proof of Lemma 6, for interference-free communication between \( i \) and \( j \), the distance between the two nodes \( i \) and \( j \) must be at least \( \tau_b = \left(\frac{p_h \cdot 2\pi}{N(\gamma - 1)(\alpha - 2)}\right)^{\gamma n} \). Since the maximum link length between a high power sender and a receiver is \( \left(\frac{p_h}{\pi}\right)^{\frac{1}{2}} \), any two high power nodes which are of distance at least \( K_b \) apart from each other can successfully send at the same time. □
Using a similar argument as in the proof of Lemma 14, we can set \( K_i \) as follows.

**Lemma 15.** For a SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \gamma > 1 \), let

\[
K_i = \left( \frac{p_i \cdot 2 \pi}{N(\gamma - 1)(\alpha - 2)} \right)^{\frac{1}{\alpha - 2}} + \left( \frac{p_i}{\gamma N^2} \right)^{\frac{1}{\alpha - 2}}
\]

Then any two low power nodes that are at least of distance \( K_i \) apart from each other can concurrently send data successfully.

**Lemma 16.** For a SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \gamma > 1 \), let

\[
l_h = \left( \frac{16(\tau_0/d_h + 1)^2 - 3[3^2 + 7]}{6} \right)
\]

where \( \mu_h = \left( \frac{p_i \cdot 2 \pi}{N(\gamma - 1)(\alpha - 2)} \right)^{\frac{1}{\alpha - 2}} \) and \( d_h = \left( \frac{p_i}{\gamma N} \right)^{\frac{1}{\alpha - 2}} \). Then any two high power nodes which are located in the \( \text{HX} \) cells with the same label according to the \( L_h \)-labeling can successfully send data simultaneously.

**Proof.** By Lemma 14, if the distance between any two \( \text{HX} \) cells with the same label is at least \( K_h \), then any two high power nodes which are located in such hexagons can successfully send data at the same time.

It is easy to show that the shortest distance between two \( \text{HX} \) cells with the same label is \( \frac{\sqrt{3}}{2} \cdot (6h_0 - 7)^2 + 3 \), where \( d_h \) is the diagonal length of a \( \text{HX} \) cell defined above. Letting \( \frac{3}{2} \cdot (6h_0 - 7)^2 + 3 \geq K_h \), we can set

\[
l_h = \left( \frac{16(\tau_0/d_h + 1)^2 - 3[3^2 + 7]}{6} \right)
\]

where \( \tau_0 = \left( \frac{p_i \cdot 2 \pi}{N(\gamma - 1)(\alpha - 2)} \right)^{\frac{1}{\alpha - 2}} \) and \( d_h = \left( \frac{p_i}{\gamma N} \right)^{\frac{1}{\alpha - 2}} \). □

Using a similar argument as in the proof of Lemma 16, we can set \( L_l \) as follows.

**Lemma 17.** For a SINR threshold \( \beta \geq 1 \), path loss exponent \( \alpha > 2 \), background noise \( N > 0 \), and some constant \( \gamma > 1 \), let

\[
l_l = \left( \frac{16(\tau_0/d_l + 1)^2 - 3[3^2 + 7]}{6} \right)
\]

where \( \tau_l = \left( \frac{p_i \cdot 2 \pi}{N(\gamma - 1)(\alpha - 2)} \right)^{\frac{1}{\alpha - 2}} \) and \( d_l = \left( \frac{p_i}{\gamma N} \right)^{\frac{1}{\alpha - 2}} \). Then any two low power nodes which are located in two \( \text{LX} \) cells with the same label according to the \( L_l \)-labeling can successfully send data simultaneously.

Next, we prove that the latency of a schedule found by HBAS is bounded by \( O(\Delta + R) \). We need the following two Lemmas 18 and 19, and these can be proved using similar arguments as in the proofs of Lemma 9 and Lemma 10, respectively.

**Lemma 18.** The number of low power connectors in an \( \text{LX} \) cell is at most 12.

**Lemma 19.** The number of high power connectors in an \( \text{HX} \) cell is at most \( 4(2 + \delta^\frac{1}{2} + \delta^\frac{1}{4}) \).

**Theorem 20.** HBAS produces data aggregation schedules whose latency is bounded by \( O(\Delta + R) \), and it is a constant factor approximation algorithm.

**Proof.** It can be easily proven using a similar argument as in the proof of Theorem 13. Steps 4 – 5 in Algorithm 3 takes at most \( (L_h + L_l)(\Delta - 1) \) timeslots. Considering Steps 8 – 14 in Algorithm 3, as there exists at most 1 high power dominator in each \( \text{HX} \) cell and at most 1 low power dominator in each \( \text{LX} \) cell, data from all dominators can be aggregated to their upper level connectors in at most \( \frac{\Delta}{2} \cdot (L_h + L_l) \) timeslots. And, as there exist at most \( 4(2 + \delta^\frac{1}{2} + \delta^\frac{1}{4}) \) high power connectors in each \( \text{HX} \) cell (Lemma 19) and at most 12 low power connectors in each \( \text{LX} \) cell (Lemma 18), data from all connectors can be aggregated to their upper level dominators at most \( \frac{\Delta}{2} \cdot \left( 4(2 + \delta^\frac{1}{2} + \delta^\frac{1}{4}) \cdot L_h + 12L_l \right) \) timeslots.

Finally, as Step 15 in Algorithm 3 takes at most \( R \) timeslots, the latency of HBAS is at most \( (L_h + L_l) \cdot (\Delta - 1) + \frac{\Delta}{2} \cdot (L_h + L_l) + \frac{\Delta}{2} \cdot \left( 4(2 + \delta^\frac{1}{2} + \delta^\frac{1}{4}) \cdot L_h + 12L_l \right) + R = O(\Delta + R) \). Thus, it is a constant factor approximation because any schedule requires \( \geq \max \{\Delta/\alpha, R\} \) timeslots by Lemma 11 and Corollary 12. □

### 6. Performance of algorithms

In this section, we compare the performance of the Square-Based Aggregation Scheduling (SBAS) and Hexagon-Based Aggregation Scheduling (HBAS) algorithms (Algorithms 1 and 3) on the same networks through simulation. In our simulation, networks are generated randomly in the Euclidean plane where the number of nodes are \( n = 300, 400, \) and 500, respectively. For each \( n \), we generate 50 different networks, and average the lengths of the schedules produced by the algorithms over the networks generated. The sink node \( s \) is randomly picked among the nodes that are deployed on an area of size \( 100 \times 100 \). For the simulation, the initial powers of nodes, \( \gamma \) and \( \delta \) are set as follows:

1. **Initial Power Assignment:** We first use Kruskal’s algorithm [25] to find the minimum spanning tree \( T_{\text{MS}} \) using edge weights defined as the distance between any two nodes. Denoting the distance of the longest edge on \( T_{\text{MS}} \) by \( d \), we set \( p_h = \gamma^\beta N d^\beta \) and \( p_l = \gamma^\beta N d^\beta \). Next, if there exists any node \( u \) such that \( \gamma^\beta N d^\beta (u, v) < p_h \), where \( v \) is the farthest neighboring node of \( u \) on \( T_{\text{MS}} \), then \( u \) is assigned \( p_h \). Otherwise, it is assigned \( p_h \). Given

![Fig. 6. Latency of SBAS.](image-url)
the initial power assignment, we get the communication graph\[G = (V, E),\] where \[E = \{(u, v) \mid d(u, v) \leq \left(\frac{\alpha}{2\beta}\right)^{\frac{1}{2}}\}\].

2. Choice of \(\gamma\): Consider a pair of a sender \(s\) and a receiver \(r\) on the link \((s, r)\), where \[d(s, r) = \left(\frac{\alpha}{2\beta}\right)^{\frac{1}{2}}\]. When \(s\) sends its data to \(r\), if there is another concurrently sending node \(u\), then \(r\) cannot receive data successfully. In order to avoid that only one node can be a sender, the power of nodes are increased by a factor of \(\gamma\), where \(\gamma = 1.1, 2.5, 10, 20, 40, 60, 80\) and \(100\).

3. Choice of \(\delta\): In our simulation, we also consider the uniform power model \((\delta = 1)\), i.e., for all \(u \in V\), \(p(u) = p_s = p_t\). For the dual power model, we use \(\delta = 2, 10, 20, 40, 60, 80\) and \(100\).

Figs. 6 and 7 show the performance of SBAS and HBAS with \(\alpha = 5, N = 1\) and \(\beta = 1\). Let us first consider the latencies produced by the SBAS algorithm. As seen in Fig. 6, for fixed \(\delta\), as \(\gamma\) becomes larger, the latency decreases. However, at some point the latency stops decreasing, i.e., increasing the power levels does not decrease latencies anymore. For fixed \(\gamma\), on the sparser networks \((n = 300)\) as \(\delta\) becomes larger (i.e., the number of high power nodes becomes larger), the latency decreases. On the denser networks \((n = 400\) and \(500)\), SBAS produces smaller-latency schedules with very few high power nodes and many low power nodes (e.g., \(\delta \leq 10\)). (See Fig. 8 that shows the averages of the number of high power nodes and the number of low power nodes for various \(\delta\).) Considering various \(\gamma\) and \(\delta\), it can be observed that SBAS produces relatively good latencies when \(\gamma \leq 10\) and \(\delta \leq 2\). Unlike the SBAS algorithm, Fig. 7 shows that there is no significant difference for various \(\gamma\) and \(\delta\) using the HBAS algorithm. It can also be observed that SBAS outperforms HBAS in most cases, although the technique which partitions networks into hexagons is commonly believed to yield better results than the square-based method.

7. Conclusion

In this paper, we have studied the problem of constructing minimum latency data aggregation schedules (MLAS) in the SINR model. We have shown that there is no approximation algorithm for MLAS having an approximation ratio better than \(\Omega(\log n)\) in the metric model unless \(\text{NP} \subseteq \text{DTIME}(n^{\log \log n})\). We have also shown that NP-hardness holds for the MLAS decision problem in the geometric model. In addition, we have proposed two constant factor approximation algorithms for the dual power model whose latency is bounded by \(O(\Delta + R)\). We also have studied the performance of the algorithms through simulation. The question as to whether geometric MLAS is APX-hard remains an open problem.

Acknowledgments

The authors thank the reviewers for some helpful comments to improve the presentation of the paper.

References