Multi-Channel Scheduling Algorithms for Fast Aggregated Convergecast in Sensor Networks

Amitabha Ghosh†, Özlem Durmaz Incel‡, V.S. Anil Kumar‡, and Bhaskar Krishnamachari‡∗
†Ming Hsieh Department of Electrical Engineering - Systems, University of Southern California
Los Angeles, CA 90089, USA, {amitabhg, bkrishna}@usc.edu
‡Department of Computer Science, University of Twente, Enschede, 7522NB, the Netherlands, o.durmaz@cs.utwente.nl
∗Virginia Bio-Informatics Institute, Virginia Tech, Blacksburg, VA 24061, USA, akumar@vbi.vt.edu

USC CENG Technical Report CENG-2008-8

Abstract—Fast and periodic collection of aggregated data is of considerable interest for mission-critical and continuous monitoring applications in sensor networks. In the many-to-one communication paradigm, known as convergecast, we consider applications wherein data packets are aggregated at each hop along a tree-based routing topology and focus on maximizing the data collection rate at a sink node. The primary hindrance in maximizing the data collection rate is the presence of interfering links. In this work, we employ TDMA scheduling and exploit the benefits of multiple frequency channels by assigning them appropriately to the parents of the routing tree in order to mitigate the effects of interference.

Our key result in the paper lies in proving that minimizing the schedule length for an arbitrarily deployed network (possibly even worst case) in the presence of multiple frequencies is NP-hard, and in designing approximation algorithms with provable performance guarantee for geometric networks. In particular, we design a constant factor approximation algorithm for networks modeled as unit disk graphs, where every node has a uniform transmission range, and a $\Delta(T) \log n$ approximation algorithm for general disk graphs, where nodes could have different transmission ranges; $n$ is the number of nodes in the network and $\Delta(T)$ is the maximum node degree on a given routing tree $T$. We also prove that a constant factor approximation is achievable on unit disk graphs even for unknown routing topologies so long as the maximum degree of any node in the tree is bounded by a constant. To the best of our knowledge, these are the first approximation results on the optimal schedule length under multiple frequencies. Our other contributions are in showing that finding the minimum number of frequencies required to remove all the interfering links in an arbitrary network in NP-hard. We give an upper bound on the maximum number of such frequencies required and propose a polynomial time algorithm that minimizes the schedule length under this scenario. Finally, we evaluate our algorithms through simulations and show various trends in performance for different network parameters.

I. INTRODUCTION

Consider a large-scale wireless sensor network (WSN) deployed for a continuous and periodic monitoring application, such as a security-surveillance network for monitoring facilities, or an environmental network for monitoring critical phenomena. The successful operation of such an application depends on its ability to extract data from the network, which often comprises periodic summaries or aggregates of raw sensor readings. It also requires fast and periodic delivery of this aggregated data from the source nodes to a common sink. Typically, the routing structure used for such data communication is a spanning tree rooted at the sink. As the data flows up this routing tree, it is aggregated at each hop, thereby eliminating redundancy and minimizing the number of transmissions [13]. We refer to this process of many-to-one communication of aggregated data from the various sources to the sink node as aggregated convergecast [12].

In this paper, we focus on the link scheduling problem of maximizing the aggregated data collection rate at the sink node under the setting of TDMA scheduling using multiple frequency channels. The key challenge in designing such scheduling protocols is that of interference. Because in a wireless environment, simultaneous transmissions from proximate nodes may interfere with each other, thus prohibiting links to be scheduled simultaneously, finding mechanisms to eliminate interference is of biggest concern. While there is a lot of research on single-channel scheduling protocol design for WSN, exploiting parallelism using multiple channels has not been well explored. Given that the current WSN hardware already provides multiple frequencies, such as the 16 orthogonal frequencies with 5MHz spacing supported by CC2420 radios on TmoteSky, it is imperative to take their full advantage in minimizing the interference and achieving faster data collection rate by concurrent transmissions. We consider contention-free scheduling protocols (e.g., TDMA) because they offer better performance for periodic data collection by eliminating collisions and retransmissions, as opposed to contention-based protocols [14].

In our time slotted system, the duration of each slot is long enough to accommodate the successful transmission or reception of a single packet. Packets are assumed to be of the same size and slots are grouped into equal sized non-overlapping frames that are repeated for periodic scheduling. We assume that each node generates only one packet in the beginning of every frame, and it has the ability to aggregate all the packets from its children as well as its own into a single packet before transmitting to its parent. The classes of aggregation functions that fall in this category include distributive and algebraic aggregation functions [13], where the size of aggregated data is constant (e.g., fits within a single packet) and does not depend on the size of the raw measurements. Examples of such aggregation functions are MIN, MAX, MEDIAN, COUNT, SUM, AVERAGE, etc.
Thus, if every node transmits only once during each frame, the sink will receive aggregated packets from all the nodes in the network starting from a certain frame. This is when we say that the network has reached a pipelined state. Note that, due to periodic scheduling, once a pipeline is established, the sink keeps receiving aggregated packets from all the nodes once every frame. We call the number of time slots in each frame the schedule length. Under this scenario, maximizing the aggregated data collection rate at the sink node is equivalent to minimizing the schedule length.

Our key result in the paper lies in proving that minimizing the schedule length for an arbitrary network in the presence of multiple frequencies is NP-hard, and in designing approximation algorithms for arbitrarily deployed geometric networks. In particular, we design a constant factor approximation algorithm for networks modeled as unit disk graphs (UDG), where every node has a uniform transmission range, and a \( \Delta(T) \log n \)-approximation algorithm for general disk graphs, where nodes could have different transmission ranges; \( n \) is the number of nodes in the network, and \( \Delta(T) \) is the maximum node degree on a given routing tree \( T \). We also prove that a constant factor approximation is achievable on UDG even when the routing topology is unknown to the algorithm designer a priori, so long as the maximum in-degree of any node in the tree is bounded by a constant. To the best of our knowledge, these are the first approximation results on the optimal schedule length under multiple frequencies. Our other contributions are in showing that finding the minimum number of frequencies required to remove all the interfering links in an arbitrary network in NP-hard. We give an upper bound on the maximum number of such frequencies required and propose a polynomial time algorithm that minimizes the schedule length under this scenario. Finally, we evaluate our algorithms through simulations and show various trends in performance for different network parameters.

A. Related Work and Paper Overview

The non-aggregated version of the convergecast problem is considered by Gandham et al. in the presence of a single channel and TDMA protocols, where the goal is to minimize the schedule length [9]. The authors describe an integer linear programming formulation and propose a distributed scheduling algorithm that requires at most \( 3n \) time slots for general networks, where \( n \) is the number of nodes in the network. A similar study [5] is presented by Choi et al. in which an NP-completeness result is proved on minimizing the schedule length under a single frequency for non-aggregated convergecast. Minimizing the schedule length by using orthogonal codes or hopping sequences to get rid of interference is studied by Annamalai et al., where they consider assigning different time slots and code pair to interfering links [1].

The problem of joint scheduling, routing, and transmission power control to improve network throughput and interference was studied by Bhatia et al. [3], and also by Bhat et al. [7]. A prominent recent work is by Moscibroda, in which scaling laws describing the achievable rate for aggregated convergecast in arbitrarily deployed sensor networks are presented under the SINR (Signal-to-Interference-plus-Noise-Ratio) model [15]. Worst-case capacity results are also proved by employing non-linear power assignment to nodes and exploiting SINR-effects. Cruz et al. use a duality based approach to address the problem of finding an optimal link scheduling and power control policy, which minimizes the total average transmission power and support high data rates [6].

In the context of general ad hoc networks, the use of multiple channels has been well researched. To improve network throughput, So et al. propose a MAC protocol that switches channels dynamically and avoids the hidden terminal problem using temporal synchronization [18]. A link-layer protocol called SSCH is proposed by Bahl et al. that increases the capacity of IEEE 802.11 networks by utilizing frequency diversity [2]. In the context of WSN there exist fewer works utilizing multiple channels. The first multi-frequency MAC protocol, MMSN, is proposed by Zhou et al. where the goal is to increase aggregated throughput [20].

Most closely related is our previous work [12], in which we described a realistic simulation-based study on tree-based data collection utilizing transmission power control, multiple frequencies, and efficient routing topologies. It is shown that once all the interfering links are removed by use of multiple frequencies, the data collection rate becomes limited by the maximum degree of the tree. We also showed that this rate can further be increased on degree-constrained trees. Our present work is different from the rest in that we propose algorithms and prove several important theoretical results on the aggregated convergecast problem under multiple frequencies.

The rest of the paper is organized as follows. In Section II we describe our problem formulation and the assumptions. In Section III, we prove two complexity results on general graphs for the aggregated convergecast problem. In Section IV, we focus on unit disk graphs and propose frequency and time slot assignment schemes that achieve constant factor approximation on the optimal schedule length. Section V focuses on general disk graphs. In Section VI we present our evaluation results, and finally Section VII concludes the paper.

II. PRELIMINARIES

In this paper, we are interested in designing link scheduling protocols that exhibit provably good performance on arbitrarily deployed networks (possibly even worst case) in the Euclidean plane under multiple frequencies. We model the network as an undirected graph \( G = (V,E) \), where \( V \) is the set of nodes and an edge \( e = (u,v) \in E \) exists between any two nodes \( u,v \in V \) if they are within the communication range of each other. We are also given a fixed node \( s \in V \) that represents the sink, and a spanning tree \( T = (V,E_T) \) rooted at \( s \) that serves as the routing topology. All the nodes except \( s \) are transmitters.
from which data is received on the corresponding time slot. For instance, \( s \) receives aggregated data from \( b, c, \) and \( f \) in slot 3 starting from frame 1. In this case, it takes two frames to reach a pipeline, as the data from \( g \) does not reach \( s \) in frame 1. Thus, from frame 2 onwards, \( s \) receives aggregated data from all the nodes in the network once in every three time slots; so the minimum schedule length is 3. Note that, there may exist other assignments, such as \( f_2 \) to \( a, c, \) and \( s, \) and \( f_1 \) to \( b \) yielding the same schedule length. However, if we had only one frequency, the minimum schedule length is 5, as shown in Fig. 1(d).

### III. Assignment on General Graphs

#### A. Scheduling With Unlimited Frequencies

From the illustration above, we see that assigning different frequencies to the receivers in an appropriate way can mitigate the effects of interference and shorten the schedule length. In this subsection, we first study the problem of finding the minimum number of frequencies needed to remove all the secondary interfering links. We say that a secondary interfering link is removed if the two receivers of an edge pair are assigned different frequencies. Note that, primary interference cannot be removed using multiple frequencies.

**Minimum Frequency Assignment**: Given a tree \( T \) on an arbitrary graph \( G \) and an integer \( q \), is there a frequency assignment to the receivers in \( T \) using at most \( q \) frequencies such that all the secondary interfering links are removed?

**THEOREM 1.** Minimum Frequency Assignment is NP-complete.

The proof is by reduction from the Vertex Color problem and is omitted here due to lack of space. We now give an upper bound on the maximum number of such frequencies required and describe a polynomial time algorithm that minimizes the schedule length under this scenario.

**LEMMA 1. Construct a constraint graph \( G_C = (V_C, E_C) \) from the original graph \( G \) as follows. For each receiver (i.e., non-leaf nodes) in \( G \), create a node in \( G_C \). Create an edge between two such nodes in \( G_C \) if their corresponding receivers in \( G \) are on two edges that form secondary interfering links. Then, the number \( K_{\text{max}} \) of frequencies that will remove all the secondary interfering links is bounded by: \( K_{\text{max}} \leq \Delta(G_C) + 1 \), where \( \Delta(G_C) \) is the maximum node degree in \( G_C \).

**Proof:** Since we create an edge between every two nodes in \( G_C \) whenever their corresponding receivers in \( G \) form a secondary interfering link, assigning different frequencies to every such receiver-pair in \( G \) is equivalent to assigning different colors to adjacent nodes in \( G_C \). Thus, \( K_{\text{max}} \) is equal to the minimum of the number of colors needed to vertex color \( G_C \), called its \textit{chromatic number} \( \chi(G_C) \). Since \( \chi(G) \leq \Delta(G) + 1 \), for arbitrary \( G \), the lemma follows.

Once all the secondary interfering links are removed, the problem of minimizing the schedule length on the graph \( G \)
reduces to one on the tree $T$. In the following, we propose an algorithm, BFS-TIMESLOT ASSIGNMENT (Algorithm 1) that runs in $O(|E_T|^2)$ time and minimizes the schedule length under this scenario.

**Algorithm 1 BFS-TIMESLOT ASSIGNMENT**

1. Input: $T = (V, E_T)$
2. while $E_T \neq \emptyset$ do
3. $e \leftarrow$ next edge from $E_T$ in BFS order
4. Assign minimum time slot to $e$ respecting adjacency constraint
5. $E_T \leftarrow E_T \setminus \{e\}$
6. end while

In each iteration (lines 2-6) of the Breadth-First Search (BFS) time slot assignment, an edge $e$ is chosen in the BFS order (starting from any node), and is assigned the minimum time slot that is different from all its adjacent edges. We prove such an assignment gives a minimum schedule length that is equal to the maximum degree $\Delta(T)$ of $T$.

**Theorem 2.** Algorithm BFS-TIMESLOT ASSIGNMENT on a tree $T$ gives a minimum schedule length equal to $\Delta(T)$.

**Proof:** The proof is by induction on $i$. Let $T^i = (V^i, E_T^i)$ denote the subtree of $T$ in the $i^{th}$ iteration constructed in the BFS order, where $E_T^i$ comprises all the edges that are assigned a slot, and $V^i$ comprises the set of nodes on which the edges in $E_T^i$ are incident. Note that, $|E_T^i| = i$, because at every iteration exactly one edge is assigned a slot. For $i = 1$, clearly the number of slots used is 1, equal to $\Delta(T^1)$. Now, assume that the number of slots $N(i)$ needed to schedule the edges in $T^i$ is $\Delta(T^i)$. In the $(i+1)^{th}$ iteration, after assigning a slot to the next edge in BFS order, the number of slots needed in $T^{i+1}$ can either remain the same as before, or increase by one. Thus,

$$N(i + 1) = \max \{N(i), N(i) + 1\} \tag{1}$$

If it remains the same, $N(i+1)$ is still the maximum degree of $T^{i+1}$ at end of $(i+1)^{th}$ iteration. Otherwise, if it increases by one, the new edge must be incident on a node $v^*$, common to both $T^i$ and $T^{i+1}$, such that the number of incident edges on $v^*$ that were already assigned a time slot at the end of $i^{th}$ iteration was $\Delta(T^i)$. This is so because in the BFS traversal, all the edges incident on a node are assigned a slot first before moving on to the next node, and because the slot assigned to the new edge is the minimum possible that is different from all that already assigned to the edges incident on $v^*$ until the $i^{th}$ iteration. Thus, at the end of $(i+1)^{th}$ iteration, the number of slots used $N(i)+1$ is equal to the number of assigned edges incident on $v^*$ which, in turn, equals $\Delta(T^{i+1})$. This proves the inductive step. Therefore, it holds at every iteration of the algorithm until the end when $i = |V| - 2$, yielding a schedule length equal to the maximum degree $\Delta(T) = \Delta(T^{|V|-1})$. Now, since assigning different time slots to the adjacent edges of $T$ is equivalent to edge coloring $T$, which requires at least $\Delta(T)$ colors, the schedule length is minimum.

**B. Scheduling With Limited Number of Frequencies**

We showed that with sufficient number of frequencies, all the secondary interfering links can be removed and a minimum length schedule can be found in polynomial time. However, typically there is a limitation on the number of frequencies over which a given transceiver can operate. In this subsection, we study the problem of minimizing the schedule length on an arbitrary graph when a limited number of frequencies are available (Multi-Frequency Scheduling problem). First, we state a known result in Lemma 2 on distance-2-edge-coloring (also called strong edge coloring) on trees that we use in the proof of one of our key results in Theorem 3.

**Definition 1.** Two edges $e, e' \in E$ in a graph $G = (V, E)$ are within distance 2 of each other if either they are adjacent or both are incident on a common edge.

A distance-2-edge-coloring of $G$ requires that every two edges that are within distance 2 of each other have distinct colors. The fewest such colors needed is called the strong chromatic index, $s\chi'(G)$, and finding it for general graphs is known to be NP-hard [10]. It is easy to see that even when all the receivers in $G$ are assigned the same frequency, the minimum schedule length is no more than $s\chi'(G)$.

**Lemma 2.** The strong chromatic index $s\chi'(T)$ of a tree $T = (V, E_T)$ is given by [8]:

$$s\chi'(T) = \max_{(u,v) \in E_T} \{\deg(u) + \deg(v) - 1\}$$

**Multi-Frequency Scheduling:** Given a tree $T$ on an arbitrary graph $G$, and two positive integers $p$ and $q$, is there an assignment of time slots to the edges in $T$ using at most $q$ frequencies such that the schedule length is at most $p$?

**Theorem 3.** Multi-Frequency Scheduling is NP-complete.

**Proof:** It is easy to see that Multi-Frequency Scheduling is in NP. Given a particular assignment, one can verify in polynomial time that (i) at most $q$ frequencies and $p$ time slots are used, (ii) either the receivers of every secondary interfering links are assigned different frequencies or their edges are on different time slots, and (iii) all adjacent edges are on different time slots.

To show NP-hardness, we reduce an instance $G' = (V', E')$ of the Vertex Color problem to an instance $G = (V, E)$ of the Multi-Frequency Scheduling problem, as illustrated with an example in Fig. 2. Our gadgets for reduction are as follows.

Let $|V'| = n$. For every vertex $v_i \in V'$, create a set $S_i$ of $q$ pairs of nodes $\{u_{is}, v_{is}\} : s = 1, \ldots, q\}$ in $G$, and join each pair with an edge $e_{is}$, treating $u_{is}$ as the parent of $v_{is}$. Then, create $\binom{q}{2} = q(q - 1)/2$ secondary interfering links between all such pairs in each $S_i$ as follows. Consider each $u_{is}$ in turn, for $s = 1, \ldots, q - 1$, and create an edge from $u_{is}$ to $v_{il}$, for all $l > s$ (see Fig. 2(b) for $q = 2$).
Next, for every edge $e_{ij} = (v_i, v_j) \in E'$, create $q^2$ secondary interfering links in $G$ by considering the two sets: $S_i = \{(u_{is}, v_{is}) : s = 1, \ldots, q\}$ and $S_j = \{(u_{js}, v_{js}) : s = 1, \ldots, q\}$, and creating an edge from each $u_{is}$ to each $v_{js}$. Then, for each $S_i$, construct a binary tree $T_i^k$ creating additional nodes and edges, and treating the set $\{u_{is}\}$ of nodes as leaves, for $s = 1, \ldots, q$.

Finally, treating the roots of $T_i^k$'s as leaves, create a binary tree on top of it, and designate the root of it as the sink $s$. The reduction clearly runs in polynomial time and creates an instance of the Multi-Frequency Scheduling problem. Next, we show that there exists a solution to the Vertex Color problem using at most $p$ colors and if only if there exists an assignment in $T$ using at most $q$ frequencies and at most $p$ plus a constant number of time slots.

Suppose $G'$ is vertex colorable using at most $p$ colors, and $v_i$ is assigned color $i$. First, assign frequency $f_s$ to $u_{is}$, for $s = 1, \ldots, q$, in each $S_i$, and any one of the $q$ frequencies, say $f_s$, to all the parents in the rest of tree (see Fig. 2(c)). Then, assign time slot $t$ to all the $q$ edges connecting the pairs $(u_{is}, v_{is})$, for $s = 1, \ldots, q$, in each $S_i$. Because all the receivers in $T_i$ are on different frequencies, assigning the same time slot to all the edges in $S_i$ does not create secondary interfering links within each $S_i$. Also, since only non-adjacent vertices in $G'$ may have the same color, any two sets of edges $S_i$ and $S_j$ that are on the same time slot cannot have secondary interfering links between each other.

Next, the lowest level edges, which connect to the $\{u_{is}\}$ nodes, of all the binary trees $T_i^k$, $\forall i$, can be scheduled using at most 2 time slots, because all the edges in each $S_i$ are assigned the same slot. Finally, all the remaining edges in the binary tree can be scheduled in polynomial time because a distance-2-edge-coloring on trees can be computed in polynomial time [17], and within number of time slots no more than its strong chromatic index which, from Lemma 2, equals at most 5.

Conversely, suppose there exists a valid assignment in $G$ that uses at most $q$ frequencies and at most $p$ plus a constant number of time slots. Assign colors to the vertices in $G'$ as follows. For each frequency $f_s$, consider the set of edges $E_{ts} = \{(u_{is}, v_{is})\}$, which are assigned time slot $t$, for $t = 1, \ldots, p$, in order. Since the edges in $E_{ts}$ are on the same slot and their receivers are on the same frequency, they cannot form secondary interfering links, and so each one of them must lie in a different $S_i$. Therefore, each edge in $E_{ts}$ has a corresponding vertex in $G'$, no two of which are adjacent. Select those edges in $E_{ts}$ whose corresponding vertices are unassigned, and assign color $t$ to all of them. Repeat the above assignment for all the frequencies $f_s$, for $s = 1, \ldots, q$. Clearly, this uses at most $p$ colors and assigns different colors to adjacent vertices. Also, because we run the above procedure over all frequencies and over all time slots, and select an edge from $E_{ts}$ only if its corresponding vertex is unassigned, exactly one edge gets picked from each $S_i$. Therefore, every node in $G'$ gets a proper color, and the theorem follows.

IV. ASSIGNMENT ON UNIT DISK GRAPHS

In this section, we consider networks that are modeled as unit disk graphs and design approximation algorithms on the optimal schedule length.

We first consider the scenario when the routing topology is known a priori. In the next subsection, we will consider the case when the routing topology is unknown to the algorithm designer. The significance of the latter is that an optimal algorithm can then choose any arbitrary routing tree to minimize the schedule length, while the goal of the algorithm designer will be to construct a good routing topology such that scheduling on top of it would still guarantee provably good performance bounds.

A. Known Routing Topology

We divide the area covering all the nodes into a set of grid cells $\{c_i\}$, each of size $\alpha \times \alpha$, as illustrated in Fig. 3. Under a UDG model, there exists an edge between any two nodes that are at most a unit distance apart from each other. We say that two cells are adjacent to each other if they share a common edge or a common grid point. We say that an edge $e_k$ belongs to a cell $c_i$ if the receiver of $e_k$ lies within $c_i$. Thus, a cell can have 3, 5, or 8 adjacent cells depending on whether it is a corner cell, an edge cell, or an interior cell, respectively. Since the interfering links are of length at most one, interference is spatially restricted, and so time slots can be reused across cells that are spatially well separated.

In our approach to design an approximation algorithm to minimize the schedule length, we separate the frequency and time slot assignment phases. First, we assign the frequencies to the receivers of $T$ in such a way that the maximum number of nodes transmitting on the same frequency is...
In the following, we describe an algorithm $F$ for minimizing the makespan of the given jobs, which is known to be NP-hard. We denote the load on the maximally loaded frequency under assignment is equivalent to, as shown in Lemma 3, scheduling all the receivers in non-increasing order of their in-degrees. Let $m(v_j) = f_k$, then the children of $v_j$ transmit on $f_k$.

**Definition 2.** We define a load-balanced frequency assignment in $c_i$ as an assignment of the $K$ frequencies to the receivers in $R_{c_i}$ such that the maximum number of nodes transmitting on the same frequency is minimized.

To formulate this, we define the load on frequency $f_k$ in cell $c_i$ under mapping $m$ as the total number of children of all the receivers in $R_{c_i}$ that are assigned $f_k$, and denote it by $l_{c_i}^m(f_k)$. We call the number of children of $v_j$ its in-degree, and denote it by $\text{deg}^m(v_j)$. Thus,

$$ l_{c_i}^m(f_k) = \sum_{v_j \in R_{c_i} : m(v_j) = f_k} \text{deg}^m(v_j) \quad (2) $$

Then, a load-balanced frequency assignment $m^*$ in $c_i$ is:

$$ m^* = \arg \min_m \max_{f_k} \left\{ l_{c_i}^m(f_k) \right\} \quad (3) $$

We denote the load on the maximally loaded frequency under $m^*$ in $c_i$ by $L_{c_i}^{m^*}$. Finding a load-balanced frequency assignment is equivalent to, as shown in Lemma 3, scheduling jobs on identical machines to minimize the makespan (last finishing time of the given jobs), which is known to be NP-hard [11]. In the following, we describe an algorithm **FrequencyGreedy** (Algorithm 2) that achieves a $(\frac{3}{2} - \frac{1}{3K})$-approximation on the optimal load.

**Algorithm 2 FrequencyGreedy**

1. In each cell $c_i$, do the following:
   1. Sort the nodes in $R_{c_i}$ in non-increasing order of their in-degrees. Let this order be: $\text{deg}^m(v_1) \geq \text{deg}^m(v_2) \geq \ldots \geq \text{deg}^m(v_n)$
   2. Starting from $v_1$, assign each successive node a frequency from \{ $f_1, \ldots , f_K$ \} that has the least load on it so far, breaking ties arbitrarily.

**Lemma 3.** The algorithm **FrequencyGreedy** in each cell $c_i$ gives a $(\frac{3}{2} - \frac{1}{3K})$-approximation on $L_{c_i}^{m^*}$.

**Proof:** Consider a job scheduling problem with $K$ identical machines $m_1, \ldots , m_K$, and $n$ jobs $1, \ldots , n$. Executing a job $j$ on any machine takes time $t_j > 0$. Thus, if $\Psi(k)$ denote the set of jobs assigned to machine $m_k$, then the total time $m_k$ takes is $\sum_{j \in \Psi(k)} t_j$, and the makespan is $\max_{1 \leq k \leq K} \left\{ \sum_{j \in \Psi(k)} t_j \right\}$. The objective is to find an assignment of the jobs to the machines that minimizes the makespan.

In our load-balanced frequency assignment formulation, we map each receiver $v_j \in R_{c_i}$ to job $j$, and the in-degree $\text{deg}^m(v_j)$ to $t_j$. Map each frequency $f_k$ to machine $m_k$. The load on $f_k$ is therefore equal to the total time $m_k$ takes. Thus, minimizing the maximum load over all the frequencies is equivalent to minimizing the makespan over all the machines. Under this mapping, **FrequencyGreedy** is identical to Graham’s list scheduling algorithm according to the Longest Processing Time (LPT) [11] first, which achieves a $(\frac{3}{2} - \frac{1}{3K})$-approximation on the minimum makespan. Therefore, the lemma follows.

**Lemma 4.** Let $\gamma_i$ denote the set of time slots needed to schedule all the edges in cell $c_i$. Then, the minimum schedule length $\Gamma$ for the whole network is bounded by: $\Gamma \leq 4 \cdot \max_{c_i} \left\{ |\gamma_i| \right\}, \forall \alpha \geq 2$.

**Proof:** Since in a UDG the distance between any two adjacent nodes is at most one, two edges that belong to non-adjacent cells must have their transmitters at least two hops away from the receiver of the other, for any $\alpha \geq 2$. Therefore, two such edges can be scheduled on the same time slot regardless of their receiver frequencies, such as $c_1$ and $c_4$ in Fig 3. Thus, the set $\gamma_i$ of time slots needed to schedule all the edges in $c_i$ can be reused in any other cell $c_j$ that is non-adjacent to $c_i$, for any $\alpha \geq 2$. This is equivalent to vertex coloring a graph in which each node represents a cell and an edge exists between any two nodes if the corresponding cells are adjacent to each other; the colors represent pair-wise disjoint sets of time slots.

Although vertex coloring an arbitrary graph is NP-hard [10], the particular color assignment to the cells shown in Fig. 3 gives an optimal assignment because of the regular grid structure. Since we need at most four different colors, at most four pair-wise disjoint sets of time slots are sufficient to schedule the whole network.

**Lemma 5.** If $L^{\phi}_{c_i}$ denote the load on the maximally loaded frequency in $c_i$ under mapping $\phi : R_{c_i} \rightarrow \{ f_1, \ldots , f_K \}$ achieved by algorithm **FrequencyGreedy**, then any greedy time slot assignment can schedule all the edges in $c_i$ within $2 \cdot L^{\phi}_{c_i}$ time slots.

**Proof:** Consider a multi-graph $H$ =
maximize slot if frequencies can be colored in \( \max \) using generalizes Vizing’s theorem for edge coloring on \( G \). By running \( \mathcal{F} \), it will take at least \( v_n ≥ 4 \cdot \max \{ |\gamma_i| \} \) time slot. Let the schedule length of an optimal algorithm be \( \Gamma \), and that of an optimal algorithm be \( OPT \).

Because of the presence of interfering links, there exists a constant \( \mu_\alpha > 0 \), depending on the cell size \( \alpha \) and the deployment distribution, such that at most \( \mu_\alpha \) edges in any cell, whose receivers are on the same frequency, can be scheduled simultaneously by an optimal algorithm.

Now, regardless of the assignment chosen by an optimal algorithm, it will take at least \( L_{\max} / \mu_\alpha \) time slots to schedule the edges in \( c_i \), because \( L_{\max} \) is the minimum of the maximum number of edges on the same frequency in \( c_i \). Thus, \( OPT ≥ \max \{ L_{\max} / \mu_\alpha \} \), \( \forall c_i \), \( ⇒ OPT ≥ \max \{ L_{\max} / \mu_\alpha \} \); so

\[
\max \{ L_{\max} \} ≤ \mu_\alpha \cdot OPT
\]

By running \( \mathcal{F} \), Lemma 3 implies

\[
L_{\ Psi}^\phi ≤ \left( \frac{4}{3} - \frac{1}{3K} \right) \cdot L_{\max}^\Psi
\]

and by scheduling a maximal number of edges in each time slot, Lemma 5 implies \(|\gamma_i| ≤ 2 \cdot L_{\ Psi}^\phi \). Then, from Lemma 4:

\[
\Gamma \leq 8 \cdot \max \{ L_{\ max}^\phi \} \leq 8 \cdot \max \{ \left( \frac{4}{3} - \frac{1}{3K} \right) \cdot L_{\max}^\Psi \} ≤ 8 \mu_\alpha \cdot \left( \frac{4}{3} - \frac{1}{3K} \right) \cdot OPT
\]

\( \blacksquare \)

### B. Unknown Routing Topologies

In this subsection, we consider the scenario when the routing topology is not known to the algorithm designer a priori. Our goal is to find the properties of a routing tree that will still allow for a constant factor approximation on the optimal schedule length.

**Theorem 5.** Given a network modeled as a UDG and \( K \) frequencies, there exists an algorithm \( \mathcal{H} \) that achieves a constant factor \( \mathcal{H}_{\max} \cdot \Delta_C \)-approximation on the optimal schedule length so long as the maximum in-degree of any node in the routing tree is bounded by a constant \( \Delta_C > 0 \), where \( \mu_\alpha > 0 \) is a constant for a given cell size \( \alpha \) ≥ 2.

**Proof:** Let \( V_{c_i} \) denote the set of nodes in cell \( c_i \). We note that the set of receivers in the tree depends on the routing topology, but the total number of nodes \( V_{c_i} \) depends only on the graph. Because an optimal algorithm can simultaneously schedule at most a constant number \( \mu_\alpha > 0 \) of nodes (edges) in any \( c_i \), whose parents are on the same frequency, the best it could do with \( K \) frequencies is distributing the nodes in \( V_{c_i} \) evenly among all the frequencies so that \( |\{ V_{c_i} / K \} | \) is the minimum of the maximum number of nodes transmitting on the same frequency. Thus, \( OPT ≥ |\{ V_{c_i} / K \} | / \mu_\alpha \), \( \forall c_i \), \( ⇒ OPT ≥ \max \{ |\{ V_{c_i} / K \} | / \mu_\alpha \} \), so

\[
\max \{ |\{ V_{c_i} / K \} | \} ≤ \mu_\alpha \cdot OPT
\]

Suppose \( R_{c_i}(T) = \{ v_1, \ldots, v_{n_i} \} \) denote the set of receivers in \( c_i \) on an arbitrary routing tree \( T \), and suppose \( \Delta_C^m \) be the maximum in-degree of any node in \( T \). Define a cyclic frequency assignment under mapping \( \psi \) : \( R_{c_i}(T) \rightarrow \{ f_1, \ldots, f_K \} \) as follows:

\[
\psi(v_i) = \begin{cases} 
  i \text{ mod } K, & \text{if } i \neq qK \\
  K, & \text{if } i = qK 
\end{cases}
\]

where \( q \in \mathbb{N}_+ \) is a positive integer. It is easy to see that the maximum number of receivers that are on the same frequency is \( |\{ R_{c_i}(T) / K \} | \). Therefore, the load \( L_{c_i}^\psi \) on the maximally loaded frequency in \( c_i \) is bounded by:

\[
L_{c_i}^\psi ≤ |\{ R_{c_i}(T) / K \} | \cdot \max_{v_j \in R_{c_i}(T)} \{ \deg_{c_i}(v_j) \} ≤ |\{ V_{c_i} / K \} | \cdot \Delta_C(T)
\]

The load \( L_{c_i}^\psi \) on the maximally loaded frequency produced by \( \mathcal{F} \) cannot be more than \( L_{\ max}^\phi \); thus

\[
L_{c_i}^\psi ≤ L_{c_i}^\phi ≤ |\{ V_{c_i} / K \} | \cdot \Delta_C(T)
\]
Then, scheduling a maximal number of edges in each time slot and using Lemma 4, Lemma 5 as before, and (9) it follows that:

$$\Gamma_T \leq 8 \cdot \max \{ [[V_{ci}] / K] \cdot \Delta^{in}(T) \}$$  \hspace{1cm} (10)$$

Since $|V_{ci}|$ and $\Delta^{in}(T)$ are independent of each other, we can take the maximum separately on the two terms as:

$$\Gamma_T \leq 8 \cdot \max \{ [[V_{ci}] / K] \cdot \max \{ \Delta^{in}(T) \} \}
= 8 \cdot \max_{ci} \{ [[V_{ci}] / K] \cdot \Delta^{in}(T) \}$$

$$\leq 8 \mu_\alpha \cdot \Delta^{in}(T) \cdot OPT$$ \hspace{1cm} (11)

Thus, (11) implies that so long as the maximum in-degree of a node in $T$ is bounded by a constant $\Delta_C > 0$, the theorem holds. Although finding a degree-bounded spanning tree on a general graph is known to be NP-hard [10], for any UDG it is always possible to find a spanning tree of degree at most 5 [19]. Therefore, the theorem follows.

V. ASSIGNMENT ON GENERAL DISK GRAPHS

In this section, we relax our assumption of nodes having a uniform transmission range, and consider networks where nodes could transmit at different power levels resulting in different transmission ranges. Such networks could be modeled as general disk graphs, where a directed edge from node $u$ to $v$ exists if $d(u,v) \leq r(u)$, where $r(u)$ is the transmission range of $u$. We consider only those edges that are bidirectional, i.e., both $r(u)$ and $r(v)$ are greater than or equal to their Euclidean distance.

For an edge $e = (u, v)$, we use the convention that $u$ is the transmitter and $v$ is the receiver. The Euclidean length of $e$ is denoted by $l(e)$. Define $I(e)$ as the set of edges that are either adjacent to $e$ or form a secondary interfering link with $e$. Also, define $I\geq(e) \subseteq I(e)$ as the subset of edges of $I(e)$ whose Euclidean length is greater than or equal to that of $e$, i.e.,

$$I\geq(e) = \{ e' : e' \in I(e) \text{ and } l(e') \geq l(e) \}. \hspace{1cm} (12)$$

As before, we separate the two phases of frequency assignment and time slot assignment. Our approach in the frequency assignment phase is to assign the frequencies in such a way that maximizes the maximum number of edges that interfere with any given edge. Once this is done, we devise a time slot assignment strategy that optimizes the schedule length.

We formulate the frequency assignment subproblem as a 0-1 Integer Linear Program (ILP). Define the indicator variables $X_{ek}$ for edge $e = (u, v)$ as follows:

$$X_{ek} = \begin{cases} 
1, & \text{if } v \text{ is assigned frequency } f_k \\
0, & \text{otherwise}
\end{cases}$$

A frequency assignment is therefore a 0-1 assignment to the variables $X_{ek}, \forall e = (u, v) \in E_T, \forall f_k$. Furthermore, for an edge $e = (u, v)$ on frequency $f_k$, define $Z_{ek}$ as the total number of edges in $I\geq(e)$ that are also on frequency $f_k$. Thus,

$$Z_{ek} = \sum_{e'=(u',v') \in I\geq(e)} \sum_{v'} n(e,v') X_{v'k},$$

where $n(e,v') = |\{ e' = (u',v') \in I_2(e) : l(e') \geq l(e) \}|$.

If we are given a frequency assignment $X$, the following lemma (based on [21], [22]) shows how the schedule length is related to the $Z_{ek}$’s.

**Lemma 6.** [21], [22]) Let $X_{ek}$ and $Z_{ek}$ be as defined above. Then, $\Omega(\max_{e,f_k} Z_{ek})$ is a lower bond on the length of any schedule for the edges. Also, it is possible to schedule all the edges using $\max_{e,f_k} Z_{ek}$ time slots.

**Proof:** (sketch) The lower bound directly follows from [21], [22]. We briefly sketch a greedy scheduling algorithm (also from [21], [22]), which implies the upper bound.

Let $E_T = \{ e_1, e_2, \ldots, e_{n-1} \}$, with the edges numbered so that $l(e_1) \geq l(e_2) \ldots$. Our scheduling algorithm assigns a time slot $t(e_i)$ for each edge $e_i$ in the following manner: consider the edges in the order $e_1, e_2, \ldots, e_{n-1}$, and for edge $e_i$, assign the smallest available time slot $t = t(e_i)$ so that (i) for each edge $e_j$ with $j < i$ having the same receiver as $e_i$, we have $t(e_j) \neq t(e_i)$, and (ii) for each $e_j$ such that $e_i, e_j \in E_T^k$ (i.e., they are assigned the same frequency), we have $t(e_i) \neq t(e_j)$. It can now be shown that the number of slots needed is at most $\max_{e,f_k} Z_{ek}$.

The above lemma implies that we should find a frequency assignment that minimizes $\max_{e,f_k} Z_{ek}$. We formulate this by the following ILP.

**Minimize** $\lambda$

subject to:

$$\forall e = (u,v), \forall f_k : \sum_{v'} n(e,v') X_{v'k} \leq \lambda \hspace{1cm} (13)$$
$$\forall e = (u,v) : \sum_{f_k} J_k X_{ek} = 1, \hspace{1cm} (14)$$
$$\forall e = (u,v), \forall f_k : X_{ek} \in \{0,1\} \hspace{1cm} (15)$$

The second constraint guarantees that each of the receivers is assigned a single frequency. Since solving this ILP is NP-hard, we first find the solution to the linear programming (LP) relaxation of it, which is obtained by modifying the third constraint to include fractional values for the indicator variables as $X_{ek} \in [0,1]$. Let the optimum (fractional) values for the indicator variables $X_{ek}$ obtained by solving the LP relaxation be $X_{ek}^\star$, and let $\lambda^\star$ be the corresponding objective value. We now construct integral random variables $Y_{ek}$ by rounding the fractional values $X_{ek}$ in the following manner: for each $v$ and $f_k$, we choose $Y_{ek} = 1$ with probability $X_{ek}^\star$; this rounding is done in a mutually exclusive manner, so that for each $v$, there is exactly one frequency $f_k$ with $Y_{ek} = 1$; this can be done since $\sum_k X_{ek}^\star = 1$.

**Lemma 7.** Let $Y_{ek}$ be the rounded solution, as described
above. Then,
\[
\max_{e=(u,v), f_k} \left\{ \frac{1}{2} \sum_{e'=(u',v') \in I(e)} Y_{e'k} \right\} = O \left( \Delta(T) \log n \cdot \lambda^* \right),
\]
with probability at least \(1 - 1/n\).

**Proof:** Because of our randomized rounding strategy,
\[
E[Y_{ik}] = Pr[Y_{ik} = 1] = X_{ik}^*.
\]

Let \(\hat{Z}_{ek} = \sum_{e'=(u',v') \in I_e(e)} Y_{e'k} = \sum_{e'=(u',v')} n(e, v') Y_{e'k}\).

Therefore, by linearity of expectation,
\[
E[\hat{Z}_{ek}] = \sum_{e',v'} n(e, v') E[Y_{e'k}] = \sum_{e',v'} n(e, v') X_{e'k}^* \leq \lambda^*.
\]

Next, note that \(n(e, v') \leq \Delta(T)\) for any \(e, v'\). Therefore, by the weighted version of the Chernoff bound [23], it follows that for any edge \(e\) and frequency \(f_k\), we have
\[
Pr[\hat{Z}_{ek} \geq \lambda^* \Delta(T) \log n] \leq 1/n^3.
\]

Since the number of edges in \(T\) and the number of frequencies are both \(O(n)\), we have
\[
Pr[\max_{e,k} \hat{Z}_{ek} \geq \lambda^* \Delta(T) \log n] \leq \sum_{e,k} Pr[\hat{Z}_{ek} \geq \lambda^* \Delta(T) \log n] \leq 1/n,
\]
where the first inequality follows from the union bound.

The above rounded solution, along with the scheduling algorithm from Lemma 6, leads to the following approximation guarantee.

**Theorem 6.** The schedule constructed by Lemma 6, along with the frequency assignment using the above randomized rounding procedure results in a schedule of length \(O(\Delta(T) \log n)\) times the optimum.

### VI. Evaluation

In this section, we evaluate the performance of our algorithms through simulations on UDG. We generate connected networks by randomly placing nodes in a square region of maximum size 200 x 200 unit\(^2\) and connecting any two nodes that are at most 25 units apart (scaled up for convenience).

#### A. Frequency Bounds

Fig. 4 compares the number of frequencies needed as a function of density to remove all the secondary interfering links on shortest path trees (SPT) as calculated from the upper bound \(\Delta(G_C) + 1\) with that from a Largest Degree First (LDF) assignment (where a node is assigned the first available frequency in non-increasing order of degrees). Here, the number of nodes \(N\) is fixed at 200 and the length \(l\) of the square region is varied from 200 to 20 so the density \(d = N/l^2\) varies from 0.005 to 0.5.

The trend shows that the number of frequencies initially increases with density, reaches a peak, and then steadily goes down to one. This happens because of two opposing factors: as the density goes up, the parents link up with more and more new nodes, thus increasing the number of secondary interfering links; however, at the same time the number of parents (on the SPT) gradually decreases because the deployment region gets smaller in size. As we go right in the graph, the latter effect starts dominating until the network finally turns into a single hop network with the sink as the only parent. We also observe that for sparser networks there is a significant gap between the upper bound and the LDF scheme as compared to that in denser networks. This is because in sparser settings there are many parents, resulting in higher \(\Delta(G_C)\), and assigning a distinct frequency to the largest degree parent according to LDF removes more interfering links at every step than it does for denser settings when the parents are fewer and have similar degrees.

![Fig. 4. Number of frequencies required to remove all the secondary interfering links as a function of network density for shortest path trees.](image)

#### B. Schedule Length

We evaluate the performance of our greedy algorithm \(\mathcal{G}\) of Theorem 4 for \(l = 200\) on shortest path trees and Minimum Interference Trees (MIT). We note that the constant factor approximations in our algorithms depend on the parameter \(\mu_{\alpha}\), which decreases with decreasing \(\alpha\). However, the smallest \(\alpha\) for which Lemma 4 holds is 2. Thus, in our experiments we chose \(\alpha = 50\), which is again scaled up 25 times, as is the UDG. We do not present our evaluation results on general disk graphs due to lack of space.

1) Shortest Path Tree: Fig. 5 shows the schedule length of \(\mathcal{G}\) with different number of nodes on SPT for different number of frequencies. We observe that multiple frequencies
help in reducing the schedule length, and this effect dominates with increasing network size as the curve for single frequency and those for multiple frequencies diverge from each other. We also note that the schedule lengths with three or more frequencies do not differ much, implying that for larger networks the schedule length is constrained by the number of children any receiver has rather than the number of secondary interfering links.

2) Minimum Interference Tree: Since interference is one of the limiting factors in minimizing the schedule length, we study the performance of our approximation algorithms on interference-optimal trees. We use an existing greedy algorithm LIFE [4] to construct minimum interference spanning trees. LIFE uses a particular interference model, in which the outgoing edge interference \( I_{out}(e) \) for an edge \( e = (u, v) \) is defined as the number of nodes covered by the union of the two disks centered at \( u \) and \( v \), each of radius \( \|uv\| \), where \( \|uv\| \) denotes the Euclidean distance between \( u \) and \( v \). The interference \( I_{out}(G) \) of a graph \( G \) is defined as the maximum edge interference over all edges. The greedy strategy in LIFE is to construct a minimum spanning tree with the weight of an edge \( e \) equal to \( I_{out}(e) \).

Fig. 6 shows the schedule length computed by algorithm \( G \) on SPT with one frequency, and on MIT with one and three frequencies, for different network sizes. As expected, we observe a significant reduction in the schedule length for larger networks on MIT. Comparing Fig. 5 and 6, we note that the curve for MIT with even one frequency is lower than those for SPT with multiple frequencies, implying that interference-optimal trees can also give benefits similar to multiple frequencies in terms of reducing the schedule length. The increasing gain in larger networks is due to smaller maximum node degree on MIT compared to that of SPT. For this particular plot with one frequency, the average maximum node degree on MIT is between 4 and 9, whereas on SPT it is between 8 and 34, with more than 20 beyond 450 nodes.

VII. CONCLUSIONS

We proved two NP-completeness results on the problem of minimizing the schedule length of aggregated convergecast in sensor networks and proposed algorithms that achieve constant factor approximation on UDG and \( \Delta(T) \log n \) approximation on general disk graphs. We also evaluated some of our algorithms through simulations and showed various trends in performance for different network parameters. Even though we considered a graph-based network and interference model as opposed to the SINR-based model [15] as a first step in this paper, the results presented in [12] show that graph-based models provide a decent approximation to SINR-model behavior. Studying scheduling protocols utilizing multiple frequencies under the SINR model remains as part of our future work. From the simulation results, we observed that the schedule length improved significantly for minimum interference trees; however the trees are not guaranteed to be degree-bounded, which is a necessary condition for Theorem 5 to hold. Exploring the problem of constructing interference-optimal, degree-bounded trees is also part of our future work.

REFERENCES


[22] B. Han, V. S. Anil Kumar, M. Marathe, S. Parthasarathy, and A. Srinivasan, “Distributed Strategies for Channel Allocation and Scheduling in Software-Defined Radio Networks”, *Proceedings of Infocom’09*, Rio de Janeiro, Brazil.