International Journal of Networking and Computing – www.ijnc.org
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 3, Number 1, pages 55–74, January 2013

Distributed Algorithms for TDMA Link Scheduling in Sensor Networks

Thamer Alsulaiman
Computer Science Department, GSU
Atl, GA, USA
thamer.mohsen@gmail.com

Sushil K. Prasad
Computer Science Department, GSU
Atl, GA, USA
sprasad@gsu.edu

Alexander Zelikovsky
Computer Science Department, GSU
Atl, GA, USA
alexz@cs.gsu.edu

Received: July 25, 2012
Revised: October 27, 2012
Accepted: December 4, 2012
Communicated by Akihiro Fujiwara

Abstract

The paper is devoted to Time Division Multiple Access Link Scheduling Protocols in wireless sensor networks for full duplex (two-way) communication, where each sensor is scheduled on an incident link as a transmitter and as a receiver in two different time slots. We formulate the full duplex link scheduling problem (FDLSP) as distance-2 edge coloring in bi-directed graphs and prove tighter lower and upper bounds for the FDLSP problem. We formulate the FDLSP problem as an integer linear program (ILP). Then, we present two Δ-approximation distributed algorithms for growth bounded graphs (GBG), for modeling the sensor networks, and for general graphs, Δ being the maximum node degree in the network. The first algorithm is a synchronous Δ-approximation algorithm based on finding maximal independent sets. The second is an asynchronous Δ-approximation depth first search (DFS) based algorithm. The maximal independent set based algorithm requires only $O(\Delta log^* n)$ communication rounds (where n is the number of processors in the network) in growth bounded graphs. For general graphs, the maximal independent set based algorithm requires $O(\Delta^4 + \Delta^3 log^* n)$ communication rounds, improving upon the previous best known algorithm with $O(n\Delta^2 + n^2 m)$ communication rounds (where m is the number of links in the network). The asynchronous DFS based algorithm requires only $O(n)$ communication rounds for both general and growth bounded graphs. The simulations show that the proposed algorithms assign on average equal or fewer number of time slots compared to the best known distributed algorithm while being significantly faster.

Keywords: Distributed Algorithms, Scheduling, TDMA, Sensor Networks

[0]This is an extended version of our APDCM/IPDPS workshop paper [1].

1 Introduction

Our model of a sensor network consists of a set of stationary sensors which cooperate together in gathering data from their surrounding environment, the data then are sent to a base station or forwarded to other sensors for further processing. Sensors are small processing units with a sensing unit, a transceiver to transmit/receive data, and equipped with a battery as its power source, but the battery has limited lifetime. The processing in sensor devices consumes much less energy than transmitting data to a base station or to other sensors. In order to maximize utilization of the battery life time, instead of sending data directly to the base station or to other designated receipts, the sensors form a multi hop network where data is sent over a series of intermediate sensors to reach the intended receiver in timely and energy efficient manner.

TDMA MAC Link Scheduling. Sensor networks use the wireless medium to transmit/receive data, thus a Medium Access Control (MAC) protocol is required to ensure transmission of data. Due to collision in dynamic MAC protocols such as ALOHA, retransmission may be required to deliver data. Retransmission results in decreasing the battery life time of the sensors. Krumke et al. in [12] proposed to use Time Division Multiple Access protocol (TDMA MAC) to avoid collision. In TDMA MAC protocols, time is divided into equal intervals, called frames, and each time frame is divided into equal time slots. *Broadcast scheduling protocol* assigns time to sensors, on the other hand *link scheduling protocol* assigns time to links, i.e., a pair of sensors to send and receive. To allow flow of information in two directions in the network, from sensors to base station/sensors and from base station/sensors back to sensors, full duplex communication is required.

As noted in [8], link scheduling has the following advantages over broadcast scheduling. In broadcast scheduling, none of the distance-2 neighbors can transmit during the same time slot, while it is possible in link scheduling. Also link scheduling better conserves power since each sensor in broadcast scheduling switches on its transceiver even if it is not the intended receiver of its neighbor's message.

Hidden Terminal Problem. Both link and broadcast scheduling in sensor networks must take into account the *hidden terminal problem.* The hidden terminal problem exists when distance-2 neighbors are transmitting during the same time slot, where the intermediate sensor between the transmitting sensors receives interfering signal even if the receiving sensor is supposed to receive only from one of the transmitting sensors. In Figure 1, sensor v receives a signal from sensor u, but since sensor w is also transmitting during the same time, v also receives another signal from w, therefore, v receives two interfering signals. In broadcast scheduling, distance-2 neighbors are not allowed to transmit during the same time slot to avoid the hidden terminal problem; only distance-3 neighbors can transmit simultaneously. In link scheduling, not only distance-3 neighbors are permitted to transmit simultaneously, even distance-1 and distance-2 neighbors also can transmit simultaneously in certain situations: the hidden terminal problem does not exists if distance-1 neighbors are both transmitting simultaneously or receiving simultaneously; also distance-2 neighbors can transmit simultaneously if the intermediate node between them is not intended to be a receiver in that time slot. This advantage of link scheduling allows for concurrent transmission during a single time slot more than broadcast scheduling.

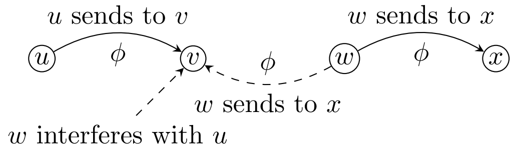

Figure 1: The Hidden Terminal Problem

Full Duplex Link Scheduling Problem (FDLSP): Given a sensor network, FDLSP problem is to find a full duplex link scheduling taking in account the hidden terminal problem using minimum number of time slots.

Network Model. Unit disk graphs (UDG) is one of the most common models for modeling sensors networks. In unit disk graphs, the sensors are points in the Euclidean plane and two sensors are neighbors (share a link) if their Euclidean distance is less than or equal to 1. This implies that sensors have identical transmission range. But, this does not reflect the reality since there are factors such as mobility and available battery power which affect the transmission range of the sensors. Thus, recently, new geometrical models have been studied in sensors network community that better model sensor networks in realty. These models include quasi unit disk graphs and unit ball graphs. Such models capture more realistic scenarios concerning connectivity based on distance [21]. All of the aforementioned models are members of a larger graph family called *growth bounded graphs* (GBG) [21]. Growth bounded graphs assume only a limit on the number of independent nodes in a neighborhood (an independent node is a node that all of its neighbors are not independent). More formally, a graph G is growth bounded if there is a polynomially bounding function $f(r)$, such that the number of independent nodes that are at shortest distance r from a node is given by $f(r)$ [21]. In this paper, we model and analyze the sensor network as growth bounded graphs. We use UDG for our experiments, because it is not known how to generate GBG in practice, while theoretical analysis of the algorithms remains valid for UDG.

Communication Model. In this paper, we develop algorithms for two distributed communication models, namely synchronous and asynchronous message passing models. In both models, the communication network is modeled as an undirected graph (V, E), where the set of nodes V represents processors of the network and the set of links E represents bidirectional non-interfering communication channels, where we assume the existence of a protocol to carry out the communication among the nodes in the network correctly. Each node has a distinct identity. In a single round a node carries out some computations, sends message to its neighbors and receives messages from of its neighbors. In the *synchronous message passing model* for distributed computing [5, 11], each node sends/receives to/from all of its neighbors in every communication round, and the communication rounds are synchronized. If we assume that each round takes one time unit, then the time complexity of the algorithm is the number of time units taken from start to completion. In the *asynchronous message passing model* for distributed computing, communication rounds are not synchronized, i.e., the time complexity of algorithm is the worst-case number of time units from start to completion [4].

Contribution. In this paper, our contribution includes the following:

- Formulating FDLSP as distance-2 edge coloring in bi-directed graphs.
- Proving a new lower bound for the number of time slots required for FDLSP, which is max(2(δ+largest cluster size+ no.of edges in largest joint clique size)) that is tighter than previous lower bound of 2Δ, Δ being the maximum node degree in the network and δ being a given node degree.
- Proving an upper bound on the number of time slots, which is $2\Delta^2$, and the existence of Δ-approximation algorithm for the FDLSP problem.
- Problem formulation of FDLSP as an integer linear program.
- A synchronous Δ-approximation distributed maximal independent set based fast algorithm for FDLSP, which requires $O(\Delta log^* n)$ communication rounds in growth bounded graphs, Δ being the maximum node degree, and $O(\Delta^4 + \Delta^3 log^* n)$ communication rounds in general graphs, compared to the best known algorithm for FDLSP problem which requires $O(n^2 m + nm\Delta)$ communication rounds.
- An asynchronous Δ-approximation distributed DFS-based algorithm for FDLSP which improves the communication rounds from previous a previous algorithm [8] with $O(n^2 m + nm\Delta)$ communication rounds to $O(n)$ and reduces the average number of time slots in practice.
- Implementation and experimental comparison with a distributed edge coloring algorithm from [8] showing the same number of time slots on average for the maximal independent set based algorithm while significantly reducing the communication complexity.

In this extended version of our paper [1], we prove tighter lower bounds for the number of time slots required for FDLSP problem. Also, we formulate the FDLSP problem as a linear program and provide experiments for testing the ILP with our algorithms. Also, we prove that our algorithms are Δ-approximation for the FDLSP problem.

The rest of the paper is organized as follows: Section 2 presents related work in channel/link assignment. Section 3 formulates FDLSP problem, and prove bounds on the number of required time slots for the problem. In Section 4, we present an ILP for FDLSP problem. In Section 5, we present a maximal independent set based distributed algorithm for FDLSP problem. In Section 6, an analysis of the algorithm is presented. Section 7 presents an asynchronous DFS based algorithm for FDLSP. Section 8 describes our simulation settings and the experimental results. Section 9 provides conclusions and future work.

2 Related Work

In this section, we present some proposed solutions in the literature that are related to channel/link assignment, and we show how those solutions are different from our algorithms. Both centralized and distributed TDMA algorithms are proposed in the literature. We first present the centralized algorithms. In [20] the author provides a unified framework for a set of broadcast and link scheduling protocols. For link scheduling protocol, the author presents a centralized half duplex edge coloring solution such that the hidden terminal problem is taken into account. The author also considers full duplex problem in the sense that a sensor can transmit and receive at the same time using multiple frequencies. In contrast, our solution considers the full duplex problem over a single frequency in the network. In [22], the proposed solution assumes that sensors can operate on different frequencies. In [23], the proposed solution is for broadcast scheduling based on distance-1 edge coloring. As mentioned earlier, broadcast scheduling allows for less concurrency of transmission as compared to link scheduling [8]. In [6], full duplex link scheduling assignment in centralized setting is presented, where the network is modeled as a bi-directed graph and time slots are assigned to links based on coloring. Although authors consider the link scheduling problem, but their solution permits multiple outgoing links to be assigned the same color. Observe that coloring more than one outgoing link with the same color reduces the number of colors in a single time frame, but a sensor transmits over only one outgoing link in a single time frame, and each of the remaining outgoing links with the same color is scheduled in another time frame.

For distributed systems, in [8], the closest to our work, a distributed algorithm is presented that solves the full duplex channel assignment problem. The algorithm first edge-colors the graph with at most $\Delta + 1$ colors. Then, a distributed DFS algorithm assigns directions to edges to obtain the full duplex schedule. Converting a coloring of an undirected graph to a valid coloring with directions while considering the hidden terminal problem is not possible directly. So, the algorithm then injects more colors in the graph to obtain a valid link scheduling. The $O(n\Delta^2 + n^2 m)$ communication rounds of the algorithm in [8] is high because of the first phase of distributed coloring. In [9], authors solve the full duplex link scheduling problem for acyclic networks only, where the time complexity of their algorithms is $O(polylog(n))$. Fault tolerant distance-2 vertex coloring algorithms is proposed in [15] and [10] for the half duplex channel assignment problem.

There are different models for sensor networks, such as unit disk graphs [7], quasi unit disk graphs [13] (both belong to growth bounded graphs family), and signal-to-interference-plus-noise ratio (SINR) model [3, 18, 17] (it belongs to the *physical* or *fading channel* models [19]). All the mentioned models are geometric models, where they better model the sensor network connectivity in practice. According to [3], SINR model is the best model to represent sensor networks, but yet the SINR model has not been studied sufficiently from algorithmic point of view. In SINR model, a message is received correctly by a receiver only if the received signal strength divided by the interfering strength of current simultaneous transmissions is above the reception threshold. We can observe that considering the SINR model for distributed algorithms is not viable *currently*, where sensors must have full knowledge of the network to communicate correctly. In [17], the authors present a distributed algorithm that enables any unit disk graph algorithm to operate under the

SINR model correctly by emulation. We, thus, consider the FDLSP under growth bounded graphs model because of the aforementioned reasons.

3 Graph Theoretical Formulation of FDLSP

In this section, we define distance-2 edge coloring problem and prove bounds for required number of time slots and the existence of an approximation algorithm for FDSLP problem. Also, we formulate the full duplex link scheduling problem (FDLSP) as a distance-2 edge coloring. Our proposed solution is to edge-color the graph taking the hidden terminal problem into account. Since we need to provide a valid schedule for full duplex communication, we start by representing the undirected sensor network graph $G = (V, E)$ as a bi-directed graph. The following definitions are needed for problem formulation:

Definition 1. A *bi-directed* graph is a directed graph in which for each arc (u,v) there exists its opposite arc (v,u).

If arc (u, v) is assigned color c, then, in the time slot c, node u transmits and node v receives from node u.

Definition 2. Given a bi-directed graph G, a *distance-2 edge coloring* assigns colors to edges such that no two edges have the same color if they share an endpoint or if one's head is adjacent to the tail of another.

Coloring in Figure 2 is feasible since v and w can send messages u and x or receive from u and x respectively. But, (u, v) and (w, x) can not have the same color since w will interfere with u's message.

The full duplex link scheduling problem (FDLSP) is equivalent to the following:

Bi-directed Distance-2 Edge Coloring Problem. Given a bi-directed graph $G = (V, E)$, find a distance-2 edge coloring with the minimum number of colors.

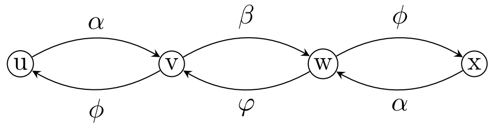

Figure 2: Feasible distance-2 edge coloring

We know that any graph can be edge-colored by Δ or $\Delta + 1$ colors by Vizing's theorem. In order to edge-color a bi-directed graph, we need 2Δ or $2(\Delta + 1)$ colors. But, since we need to avoid the hidden terminal problem in FDLSP, more than $2(\Delta + 1)$ colors may be needed for a feasible assignment for FDLSP.

The link scheduling problem is an NP-complete problem [2]. Thus, FDSLP is NP-complete, and we can hope only for an approximation algorithms. We now prove new lower and upper bounds on the number of colors in any FDLSP solution.

3.1 Lower Bound

In this subsection we provide a new tighter lower bound for the FDLSP problem that is not presented in our paper [1]. As discussed earlier and due to [8], the minimum number of colors required to distance-2 color a bi-directed graph G is 2Δ at least. While this holds true for trees, note that if we have a clique of size 3 or more, or cycles, then 2Δ colors are not sufficient. We prove a tighter lower bound for the FDLSP problem. First, we need the following definitions to facilitate the proofs.

Definition 3. A *Cluster* node v is the set of all cliques of size 3 containing node v such that all the cliques share a common edge. Node v is called the *cluster center*.

For example, in Figure 3, there are two clusters for cluster center v. The first cluster contains three cliques: vwx, vwr, vwz. The second cluster contains two cliques: vxw, vxr. The first cluster's

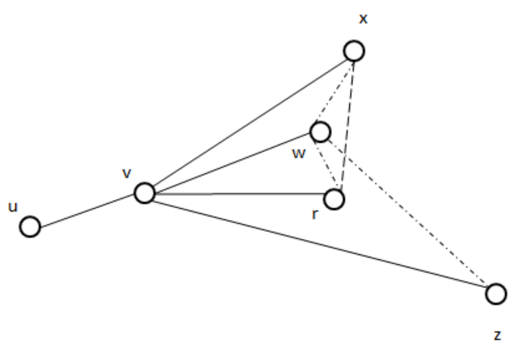

Figure 3: Clusters of node V

common edge is (v, w), and the second cluster's common edge is (v, x). The first cluster size is 3; the second cluster size is 2.

Note that any node w may belong to at most two clusters of a specific cluster center v. A *cluster size* is the number of cliques of size 3 in the cluster, and a node's cluster size is the size of the largest cluster of that node.

Definition 4. An *outer* edge of a clique belonging to a cluster of node v is the edge that does not have node v as an end point.

For example, in Figure 3, the outer edges of the largest cluster of cluster center v are (x, w), (w, r), and (w, z).

Definition 5. A *joint edge* of a given cluster of node v is an edge that connects two cluster nodes, but the formed clique do not belong to the given cluster.

For example, in Figure 3, joint edge (x, r) connects nodes x and r, but the clique vxr do not belong to the largest cluster which contains three other cliques.

Definition 6. A *joint clique* of cluster of node v is a clique that is formed by joint edges of the given cluster.

Lemma 1. *An outer edge of a clique in a cluster can not be colored with a color used by an edge incident on the cluster center.*

Proof. Without loss of generality, consider outer edge (w, x) of clique (v, x, w) in the cluster of node v in Figure 4. Clearly, edge (w, x) can not be colored with a color used by an edge incident on both x and v, or on w and v. Also, edge (w, x) can not be colored with color used by edge (v, u), because node x will receive two interfering signals from nodes w and v. Moreover, edge (w, x) can not be colored with color used by edge (u, v), because node v will receive two interfering signals from nodes w and u. Similar arguments will prevent edge (x, w) to be colored with a color used any edge incident on node v. Hence, the proof holds. □

Lemma 2. *Outer edges of two cliques belonging to the same cluster can not be colored with the same color.*

Proof. By definition, two cliques belonging to the same cluster share an edge; thus, their outer edges must share an endpoint. Hence, they can not be colored with the same color. □

Lemma 3. *Two outer edges of two cliques that do not belong the same cluster of some node can be colored with the same color.*

Proof. If two cliques do not belong to the same cluster of some node, then their directed distance is more than 2; thus, they can be colored with the same color. □

Lemma 4. *A joint edge can not be colored by a color that is used by an edge incident on a cluster center, nor an outer edge of the same cluster.*

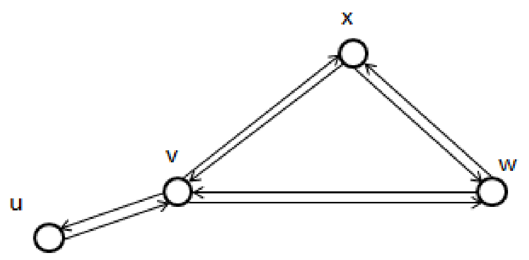

Figure 4: Cluster of node V

Proof. Note that a joint edge forms a clique of size 3 with two edges incident on the cluster center. Also, a joint edge forms a clique of size 3 with two outer edges. The proof then is similar to the proof of Lemma 1 since we get the same configuration as in Figure 4. □

Lemma 5. *A joint edge can not be colored by a color that is used by a joint edge that belongs to the same joint clique.*

The proof is obvious, and hence is omitted.

Theorem 1. *A lower bound for a bi-directed graph G to be distance-2 edge colored is max(2(δ+largest cluster size+ no.of edges in largest joint clique size)) colors.*

Proof. The proof follows by Lemmas 1 through 5. □

The lower bound provided here is not tight; to see why consider the graph in Figure 5. Assume that the edges of the graph are joint edges (we excluded cluster edges for simplicity.) Note that the number of edges in the largest joint clique size is 3. But, this graph requires more than 6 colors (we double the number of colors when we consider the bi-directed graph), it requires 10 colors to take FDLSP problem into account.

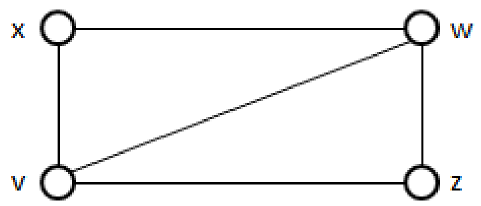

Figure 5: Two joint cliques of size 3

3.2 An Upper Bound

Lemma 6. *Any bi-directed graph G with the maximum out-degree Δ requires at most $2\Delta^2$ colors to distance-2 edge-color graph G.*

Proof. Let directed edge (u, v) from bi-directed graph G has color c (see Fig 6). Count the number of edges that if colored with color c, then a conflict in distance-2 edge coloring exists. Observe that

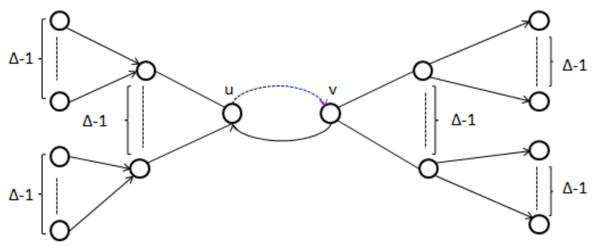

Figure 6: Maximum number of colors

the edges that have conflicts in color with edge (u, v) belong to three categories, edges incident on both nodes u and v, outgoing edges from the neighbors of node v and incoming edges to neighbors of node u. We know that the sum of the edges of both nodes u and v equals to $4\Delta - 3$ at most (there are two common directed edges belong to both u and v, and we subtract the edge (u, v) also). Also, there are $(\Delta - 1)(\Delta - 1) = \Delta^2 - 2\Delta + 1$ outgoing edges at most from any neighbor of node v and $\Delta^2 - 2\Delta + 1$ incoming edge at most to any neighbor of node u (note that we exclude edges incident on u and v because they are considered previously). Summing up those quantities gives $2\Delta^2 - 1$ edges in total.

Now, construct conflict graph $G' = (V', E')$ where the set of vertices V' represents the set of directed edges in G, and there is an edges between two vertices in G' if there is a distance-2 edge-coloring conflict between the two directed edges represented by the two vertices in G'. Clearly, the maximum degree in G' is $2\Delta^2 - 1$. A greedy vertex coloring algorithm exists which assigns at most $2\Delta^2$ colors to vertices of the conflict graph G' [16]. That is, a bi-directed graph G requires at most $2\Delta^2$ colors for distance-2 edge coloring. □

Theorem 2. *For any bi-directed graph G with the maximum out-degree Δ, there exists a deterministic Δ-approximation algorithm that distance-2 edge color the graph G.*

Proof. Any bi-directed graph can be greedily distance-2 edge colored by at most $2\Delta^2$ colors by Lemma 6. Since the lower bound is at least 2Δ, dividing the upper by the lower bound, such a greedy algorithm achieves Δ-approximation. □

Note. In complete graphs, observe that since every node is connected to every other node, then, if more than one node transmits in the graph during the same time slot, the hidden terminal problem exists. Thus, in complete graphs, the feasible assignment assigns a unique color for each link in the bi-directed graph, where the number of edges in bi-directed complete graph is $\Delta^2 + \Delta$. On the other hand, edges in even cycles requires only 4 colors for the FDSLP and edges in odd cycles requires 6 colors only [8].

4 ILP Formulation

We present an integer linear program (ILP) for the FDLSP problem, which was not presented in our previous paper [1]. ILP is helpful to test small size instances of the FDLSP problem to get the optimal solution. Our objective in the ILP is to minimize the used colors in the graph, so we maximize the concurrent transmissions carried on in a single time slot, i.e., minimize the required number of time slots.

Variables:

$$C_j = \begin{cases} 1 & \text{if color } j \text{ is used} \\ 0 & \text{otherwise} \end{cases}$$

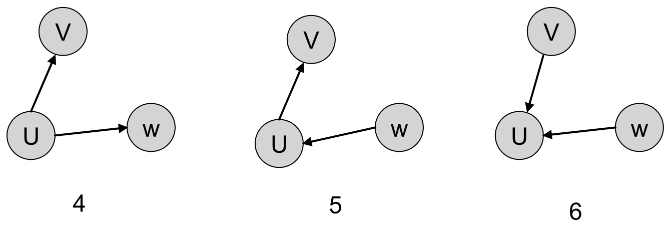

Figure 7: ILP Constraints 4, 5 and 6

$$X_{(u,v)j} = \begin{cases} 1 & \text{if arc } (u,v) \text{ has color } j \\ 0 & \text{otherwise} \end{cases}$$

Objective:

$$\sum C_j \rightarrow \ Minimize$$

Constraints:

$$\begin{array}{ll}
(1)\ \forall j, (u,v) \in E: & C_j \geq X_{(u,v)j} \\
(2)\ \forall j, (u,v), (w,u), (v,z) \in E: & X_{((w,u),j)} + X_{((v,z),j)} \leq 1, \\
(3)\ \forall (u,v) \in E: & \sum_{j=1}^{N} X_{((u,v),j)} = 1, \\
(4)\ \forall u \in V, (u,v), (u,w) \in E: & X_{((u,v),j)} + X_{((u,w),j)} \leq 1, \\
(5)\ \forall u \in V, (u,v), (w,u) \in E: & X_{((u,v),j)} + X_{((w,u),j)} \leq 1, \\
(6)\ \forall u \in V, (v,u), (w,u) \in E: & X_{((v,u),j)} + X_{((w,u),j)} \leq 1,
\end{array}$$

Constraint (1) requires j^{th} color, C_j, to be counted in the objective only if it is used. Constraint (2) takes into account the hidden terminal problem where it prevents a node to be transmitter while its adjacent node is receiver in the same time slot (color). Constraint (3) ensures that each edge is colored with a single color. The last three constraints are to ensure that each incident edge to the same node gets a unique color, where constraint 4 prevents two outgoing links originating from the same node to have the same color, constraint 5 prevents one outgoing link and another incoming link adjacent to the same node to have the same color, and constraint 6 prevents two incoming links to the same node to have the same color. See Figure 7 for constraints 4, 5, and 6.

5 Distributed Maximal Independent Set Based Algorithm (DistMIS)

In this section we present a synchronous maximal independent set (MIS) based distributed algorithm for FDLSP. Our algorithm approach is based on a distributed node-coloring algorithm presented in [21]: our algorithm computes iteratively an MIS, and then another secondary MIS; nodes in secondary MIS then colors their incident edges. A detailed explanation follows. A set of nodes $S \subseteq V$ is said to be independent in graph G if no two nodes $u, v \in S$ are neighbors in G. An independent set S is maximal if S is not a subset of any other MIS set, i.e., if $u \in V - S$, then $S \cup \{u\}$ is not an independent set. We use the terms independent and dominant for independent nodes, and dominated for non-independent nodes. We define the following to facilitate the description of the algorithm:

$dist_G(u, w)$ is the length of the shortest path between nodes u and w in graph G.

$N^r(v)$ be the *node neighborhood*, which is the set of nodes that can be reached by a shortest path of length at most r from node v.

To compute the MIS, our algorithm can use any existing synchronous distributed algorithm for computing MIS. To the best of our knowledge, the fastest algorithm for computing MIS in GBG is due

Algorithm 1 Distributed Maximal Independent Set Based Algorithm (DistMIS)

```
Require: $G(V, E)$: bi-directed graph
 1: for all $u \in V$ in parallel do
 2:     repeat
 3:         compute an MIS set $S$
 4:         $H \leftarrow S$
 5:         repeat
 6:             $G' := (S, E')$ where $E' = \{(u, v) | u, v \in S \wedge dist_G(u, v) \leq 3\}$
 7:             compute secondary MIS set $S'$ in graph $G'$
 8:             if $u \in S'$ then
 9:                 color edges incident on u greedily
10:                 broadcast color assignment to its neighbors
11:             end if
12:             $S \leftarrow S - S'$
13:             $S' \leftarrow \phi$
14:         until $S$ is empty
15:         $G \leftarrow G - H$
16:     until $G$ is empty
17: end for
```

to Schneider et. al [21], which is optimal, that it computes MIS in GBG in $O(log^*n)$ communication rounds. The MIS algorithm in [21] is uniform and local, i.e., it only requires knowledge of local neighborhood processes. Our algorithm requires that each node has distance-2 knowledge to compete for an MIS and to color its incident edges with valid color. Distance-2 knowledge of node v implies that node v have knowledge of the colors of each edge incident on any node $u \in N^2(v)$, and the status of nodes incident on those edges. When a node gets the turn to color its edges, it does so in a greedy manner. After a node edge-colors its incident edges, it broadcasts the edge color assignment to its distance-2 neighborhood. Note, if an incident edge on a node is already colored by a neighbor, the node does not change the color of the edge, it proceeds and colors the edges that are not yet colored taking into account the colors of all its incident edges.

Our algorithm consists of two consecutive phases that are repeated till all the nodes in the graph get the turn to color their incident edges. In the first phase, a maximal independent set S from graph G is formed. Before the second phase, the nodes in S form a new graph $G' = (S, E')$ where $E' = \{(u, v) | u, v \in S \wedge dist_G(u, v) \leq 3\}$. In the second phase a new MIS S' from graph G' is formed. (We call set S' a *secondary MIS* to facilitate the description of the algorithm). Nodes in S will compete to form secondary MIS S', then nodes in secondary MIS S' color their incident edges simultaneously. Note that if two nodes on distance-3 or less are coloring their incident edges, then there may be a conflict in coloring, that is why we form secondary MIS S'. The nodes in S' are excluded from S, a new graph G' is formed and a new secondary MIS S' is formed, and the nodes in the new secondary MIS S' color their incident edges. This process is repeated till all the nodes in S join some secondary MIS S' and color their incident edges. After that, the nodes in MIS S are excluded from G, i.e., $G \leftarrow G - S$. The previous phases are repeated till all the nodes in new G join some MIS S and color their incident edges. Algorithm *distMIS* is given in Algorithm 1.

It may seem that the required distance-2 knowledge for coloring is relatively high, but distance-2 knowledge is required to color an edge with feasible color in FDSLP, i.e., distance-2 knowledge is lower bound on the knowledge to produce feasible schedule for FDSLP. It is possible to bypass distance-2 knowledge requirement and color with distance-1 knowledge only by randomization. We have attempted a randomized algorithm for the FDSLP, but it produced longer schedule with speed that is close to the independent set based algorithm.

Distance-3 nodes will be competing in the second phase of the algorithm to join the secondary MIS S'. But, only the independent nodes from set S will be active in the second phase of the algorithm, and will exchange information concerning computing the secondary MIS set, not coloring. Other nodes will be just bridges to carry out the communication among the competing nodes. In

GBG, number of independent nodes in a neighborhood is bounded, hence distance-3 knowledge is of constant size for the second phase.

6 Algorithm Analysis

In this section we provide algorithm analysis and correctness proofs.

Theorem 3. *Algorithm distMIS computes feasible link assignment for FDSLP.*

Proof. Note that the algorithm terminates only after each node joins some MIS set S and then joins a secondary MIS set S' and color its incident edges. Also, since any two nodes coloring their incident edges simultaneously must belong to the same secondary MIS, thus they are at least distance-4 apart from each other; observe that any edges that are colored simultaneously maintains correct FDSLP coloring even if they are colored with the same color. Moreover, each node requires distance-2 knowledge, and it is sufficient to color any edge with feasible color (avoiding the hidden terminal problem). Thus, from the previous arguments the theorem follows. $\square$

Lemma 7. *For any undirected graph $G = (V, E)$ with maximum degree Δ, there is at most $\Delta + 1$ disjoint maximal independent set such that the union of the sets is the set of nodes V of graph G.*

Proof. Let sets $S_1, S_2, ..., S_i, ..., S_k$ be disjoint maximal independent sets of graph G, such that $1 \leq i \leq k$ and $S_i \cap S_j = \phi$ for any $i \neq j$. We now prove that $\bigcup_{i=1}^{k} S_i = V$ and $k \leq \Delta + 1$. Without loss of generality, for any node $v \in V$ with maximum degree Δ, we know that node v is either an independent node or incident to at least one independent node in any maximal independent set S_i. Thus, if node v belongs to S_i, then, for any disjoint maximal independent set S_j $(i \neq j)$, S_j must contain at least one of node v's neighbors. And since node v has Δ neighbors, and each disjoint maximal independent set for graph G must contain at least one of node v's neighbors, then there is at most Δ set of such disjoint maximal independent sets. Summing up the quantities, we get at most $\Delta + 1$ disjoint maximal independent set for any graph G with maximum node degree Δ and the union of the sets is V. Hence, the proof follows. $\square$

Recall that our algorithm computes first a maximal independent set S, then computes secondary maximal independent set S'. After the nodes in S' color their incident edges, the nodes of S' are excluded, and another secondary MIS is computed. This is repeated till all the nodes in S join some secondary MIS S'. We now conclude and prove the number of required secondary MIS such that every node in MIS S belong to one of the secondary MIS sets.

Lemma 8. *For any maximal independent set S for graph G, the number of required disjoint secondary MIS sets to contain all the nodes in S is $O(1)$ in growth bounded graphs.*

Proof. We know that in growth bounded graphs the number of independent nodes in a neighborhood $N^r(v)$ is given by the function $f(r)$, where $f(r)$ is a polynomial bounding function [21]. Since, in distMIS algorithm, secondary MIS is computed among distance-3 neighborhood, i.e., $r = 3$, observe that $f(r)$ is a constant; i.e., the number of required disjoint secondary MIS sets to contain all the nodes in S is constant ($O(1)$). Hence, the proof follows. $\square$

Theorem 4. *Algorithm distMIS requires $O(\Delta log^* n)$ communication rounds in growth bounded graphs.*

Proof. Algorithm distMIS computes at most $O(\Delta)$ disjoint MIS by Lemma 7. And it computes $O(1)$ secondary MIS for each disjoint MIS set S by Lemma 8. And, since we use the MIS distributed algorithm in [21] which requires $O(log^* n)$ rounds, distMIS algorithm requires $O(\Delta log^* n)$ rounds. Hence the proof follows. $\square$

We now show that our algorithm *distMIS* is Δ-approximation for the FDLSP problem, which was not presented in our previous paper [1]. In order to do so, we first present a sequential $\Delta + 1$ greedy node-coloring algorithm. Then, we show that our algorithm *distMIS* imitates the greedy algorithm operation in coloring the line graph obtained from the original graph. Recall that the line graph is constructed with nodes as the links of the original graph and the links between two nodes exist if there is a conflict between the edges if colored with the same color in the original graph, as shown in Lemma 6. Thus we show that our algorithm *distMIS* is a Δ-approximation for the FDLSP problem, as shown by Theorem 2.

Let algorithm *greedyColor* be a greedy algorithm that works as follows: given a graph G with maximum node degree Δ and $\Delta+1$ distinct colors, greedily pick any uncolored node v from G, color node v with a color c such that no neighbor of node v is colored with c. Continue till all the nodes in graph G are colored and terminate.

Lemma 9. *Given a graph G with maximum node degree Δ, algorithm greedyColor node-colors graph G with at most $\Delta+1$ colors and terminates correctly.*

Proof. Since the maximum node degree in graph G is Δ, any picked uncolored node v in graph G can be colored with color c from the given $\Delta+1$ colors such that no other neighbor of v is colored with, because if each neighbor of v is colored with a distinct color among the $\Delta+1$ given colors, then there is one available color that is not used yet and can be used to color v. Hence, the proof follows. □

Lemma 10. *Algorithm distMIS is Δ-approximation for the FDLSP problem.*

Proof. First, we show that at some round r, if all colored edges in previous rounds are colored such that FDLSP problem does not exist, then for any two distinct non-adjacent edges e_1 and e_2 that are colored in round r, if edges e_1 and e_2 are colored with color c, there is no conflict in distance-2 edge coloring. Note that any two nodes v and u in some secondary MIS S' are at distance 4 at least, and thus edges e_1 and e_2 are at distance 3 at least from each other. Thus, coloring e_1 and e_2 with the same color does not violate distance-2 edge-coloring.

Since any two non-adjacent edges that are colored simultaneously in the same round do not have conflict in coloring, i.e., we can say that coloring non-adjacent edges takes place sequentially, and thus coloring the line graph obtained from the original graph is carried sequentially just as the greedyColor sequential algorithm. Also, we showed in Lemma 6 that $2\Delta^2$ colors at most are required to distance-2 edge-color a bi-directed graph. Hence, by Theorem 2, algorithm distMIS is Δ-approximation. □

We conclude and prove the time complexity of our MIS based distributed algorithm in general graphs. Recall that the algorithm computes secondary MIS set among distance-3 nodes; assume that the time complexity of the algorithm is some polynomial function $O(q(G))$. We can reduce the complexity $O(q(G))$ by factor Δ as follows: instead of computing secondary MIS among distance-3 nodes, compute the secondary MIS among distance-2 nodes; then nodes in the secondary MIS colors their outgoing edges only; observe that the color assignment in this algorithm is still correct since there will be no conflict in simultaneous coloring. Note that we reduced the number of competing nodes in secondary MIS by Δ (because we excluded distance-3 nodes from the competition). Our simulations on general graphs are conducted on the second proposed version.

Theorem 5. *Algorithm distMIS requires $O(\Delta^4+\Delta^3 log^* n)$ communication rounds in general graphs.*

Proof. Algorithm distMIS computes at most $O(\Delta)$ disjoint MIS by Lemma 7. And it computes at most Δ^3 secondary MIS for each disjoint MIS set, since there is at most Δ^3 MIS node at distance-2 from any node v. Moreover, the fastest MIS distributed algorithm [14] requires $O(\Delta + log^* n)$ communication rounds in general graphs, distMIS algorithm requires $O(\Delta^4 + \Delta^3 log^* n)$ communication rounds. Hence the proof follows. □

Space complexity is $O(\Delta^2 * log\Delta^2)$; this may seem high space complexity, but it is unavoidable since the upper bound on colors is $(2\Delta^2)$ and a color is represented using $log\Delta^2$ bits. Note that in practice the space required is much less than $O(\Delta^2 * log\Delta^2)$. Message complexity is $O(m\Delta log^* n)$ for GBG, and $O(m * (\Delta^4 + \Delta^3 log^* n))$ for general graphs, since each node sends to and receives from each of its neighbors and since the model is synchronous.

We provide now a general overview of the distributed algorithm D-MGC presented in [8], and a proof sketch of the time complexity in order to compare our algorithm with D-MGC algorithm, which is, to the best of our knowledge, the fastest algorithm for the FDLSP problem. Note that the D-MGC algorithm is asynchronous. The D-MGC algorithm consists of two phases, namely distributed edge-coloring and sign (direction) assignment. In the edge-coloring phase, edges are colored first greedily using available colors, which are $(\Delta+1)$ colors. A node colors its incident edges exclusively when all of its 2-hop neighbors with higher ID are finished edge-coloring their incident edges; some edges will be uncolored since $\Delta+1$ colors are available only. In order to color the uncolored edges, each node constructs a data structure called *fan* (we omit its description). Then each node constructs a path called a *cd-path*, where a cd-path is a maximal path of alternating colors c and d along the path with some specific properties. If a cd-path is found, the path is inverted, i.e., the colors along the path are alternated. Inverting cd-paths helps in coloring uncolored edges with the available $\Delta+1$ colors. The length of the longest cd-path is $n-1$, thus, the number of communication rounds required to invert a cd-path is $O(n)$. If more than one cd-path to be inverted are overlapping, then one cd-path inversion only proceeds, the rest are locked. Locks are acquired over all the paths; to acquire locks over all overlapping paths, up to $O(n^2)$ communication rounds are required (since the underlying model is asynchronous). Hence, inverting a single cd-path requires up to $O(n^2)$ communication rounds in the worst case. Also, since there is at most $O(m)$ cd-path to be inverted (each edge may be the starting point of a cd-path), there are at most $O(n^2m)$ communication rounds required for first phase at most. Note that we omitted the description and the proof sketch of the edge-coloring part of the first phase because it requires less communication rounds, that is $O(n^2\Delta)$ at most.

Only $\Delta+1$ colors are used in the first phase of algorithm D-MGC. In order to obtain a distance-2 edge-color direction assignment, each edge in the undirected graph of the underlying network is assigned a valid direction that does not violate distance-2 edge-color requirements. Thus, each node is considered either a sender or a receiver over each of its incident edges. Then, the number of colors obtained are doubled and the directions are reversed to get a full duplex distance-2 edge color assignment. In order to assign direction for each edge, each node starts a DFS tree to assign direction for each specific edge with color c. Each DFS tree requires $O(m)$ communication rounds. Since each node may have an edge with specific color c, then there is at most n DFS trees evoked for color c by the communication network nodes. But, only one DFS tree for each color succeeds and the rest are ignored based on highest ID. Hence, there are at most $O(nm)$ communication rounds required to find a valid direction assignment for each color. Hence, there are at most $O(nm\Delta)$ communication rounds required to find a valid direction assignment for all the colors in th communication network. Note that because of the existence of cycles, a valid direction assignment may not be possible with the given $\Delta+1$ colors, thus, more colors are injected to color the edges with valid distance-2 edge-color assignment in the communication graph and more communication rounds are required. By the previous discussion, the number of communication rounds required by algorithm D-MGC are $O(n^2m+nm\Delta)$ at most. Thus, our algorithm requires significantly less communication rounds in general graphs.

7 A DFS-Based Algorithm

In this section we present a (DFS) based algorithm for FDLSP. We assume the asynchronous message passing model, where the algorithm works also for the synchronous model (but the message complexity will be higher in the synchronous model). In the DFS-based algorithm, presented in Algorithm 2, a designated node in the network starts the computation, this designated node will be the root of the DFS traversal tree. The root starts coloring its incident edges, then it passes the token to one of its unvisited neighbors with maximum degree (each node keeps a record of previously visited nodes). When a node receives the token, it asks its neighbors to send their distance-2 edge color assignment; colors its incident edges; informs all of its neighbors about its color assignment; then passes the token to one of its unvisited neighbors with the maximum degree afterward. This continues till the token reaches a node that has no unvisited neighbor, this node sends the token back to the sender after coloring its edges. If a node receives the token back, it sends the token to one of its unvisited neighbors with the maximum degree. This continues till all the nodes in the graph get the token and color their incident edges. Note, if an incident edge to a node is already colored by a neighbor, the node does not change the color of the edge, it proceeds and colors the edges that are not yet colored taking into account the colors of all its incident edges. Observe that the network nodes are visited in a depth first search order.

Algorithm 2 Distributed DFS Algorithm (DFS)

```
Require: $G(V, E)$: bi-directed graph
 1: wait for token from parent
 2: ask neighbors for distance-2 edge color assignment
 3: receive edge color assignment of distance-2 neighbors
 4: distance-2 edge-colors incident edges
 5: broadcast edge color assignment
 6: while this node has at least one unvisited neighbor do
 7:     send(token) to unvisited neighbor v with max degree
 8:     wait for token from neighbor v
 9: end while
10: send token back to parent
```

The correctness of the algorithm is obvious, since only one node is allowed to color exclusively in the network to avoid conflicts and since each node has a full knowledge about its distance-2 neighbors' color assignment before coloring. The communication complexity is $O(n)$ since a distributed DFS tree can be constructed in $O(n)$ rounds by keeping record of visited neighbors and removing a neighbor from the record in case that neighbor asks about edge color assignment (it means that it has received the token to color already). The communication complexity is much less than the D-MGC algorithm in [8] which is $(n^2m + nm\Delta)$ communication rounds. Space complexity is $O(\Delta^2 * log\Delta^2)$ messages as discussed in Section 6. On the other hand, message complexity for DFS is $O(m)$ message since the algorithm works in an asynchronous environment and each edge carries constant number of messages in total.

In Section 5, we proved that our DistMIS algorithm is Δ-approximation by showing that our DistMIS algorithm imitates a greedy sequential coloring algorithm. It is straight forward to show that our DFS algorithm also imitates the sequential greedy coloring algorithm, thus it is Δ-approximation algorithm for the FDLSP problem.

8 Simulation Settings and Experimental Results

We first present and compare the results obtained for input graphs using the ILP and our depth first based algorithm (DFS). We do not compare the ILP with our distributed maximal independent set based algorithm (distMIS), because on such small instances the MIS based algorithm operates sequentially, just like the DFS. Both the ILP and the DFS algorithm assign 2Δ colors for input tree graphs. For complete bipartite graphs $K_{n,m}$, the optimal number of colors for the link scheduling problem is more than 2Δ colors for most cases as shown using the ILP. On the other hand the DFS algorithm assigns the optimal colors in some cases, and assigns more colors on some other tested instances of the complete bipartite graphs. See Table 1.

Our distributed MIS-based (distMIS) and depth first search based (DFS) algorithms are compared to the algorithm from [8] (D-MGC). In our simulations, we test the three algorithms on both unit disk graphs and general graphs to show the effectiveness of our approach in both of the cases. Recall that unit disk graphs belong to bounded growth graphs family. We generated UDG for our simulations because, to the best of our knowledge, it is unknown how to generate general growth bounded graphs; also our theoretical analysis of the algorithm still applies on UDG.

We generated four different sets of random unit disk graphs; each set has 75 random unit disk graphs with 50, 100, 200 and 300 nodes count each. In order to generate UDG graphs, we first specify the side length of the squared area plan, and then we place the nodes in the squared plan

Input Graph	No. of Colors	
	ILP	DFS Algorithm
$K_{2,2}$	4	4
$K_{3,3}$	9	10
$K_{4,4}$	15	18
K_4	12	12
K_5	20	20

Table 1: ILP vs. Distributed DFS Algorithm on Complete bipartite and Complete graphs

randomly. After that we connect the nodes with links based on the distance between each two nodes; if distance is less than or equal to some distance d (transmission radius), then there is link between the two nodes. Note that it is not possible to fix the number of edges in UDG, but we can only fix the side length of the squared area plan and the transmission radius. Of each set with fixed number of nodes, we generated three different sets with different side length of the squared area plan (15, 17, and 20), and if two nodes are at distance less than or equal to 0.5 from each other ($radius \leq 0.5$), then there is a link between the two nodes (the unit length in our sample is 0.5). With multiple varying side length of the squared area plan, we generate both sparse and dense graphs.

For the general graph case, we generated 50 random graphs with multiple node counts, where for each node count we alter the number of edges in order to get sparse to dense graphs. For the sake of brevity, we present only two node counts, 200 and 500 nodes, and we vary the number of edges for each node count; our distMIS algorithm for general graphs case is the algorithm proposed in Section 6.

In the UDG case, the results are presented in Figure 8 with squared area plan's side length 15, in Figure 9 with squared area plan's side length 17, and in Figure 10 with squared area plan's side length 20. In Figures 8, 9 and 10, the horizontal axis represents the number of nodes of the generated UDG graphs and the average degree of the nodes in the undirected graph separated by comma; the vertical axis represents number of generated time slots for FDLSP problem in *logarithmic scale* of base 10 in order to contain upper and lower bounds properly in the plot; we provide the lower bound and the upper bound of time slots in the plots drawn from the theoretical analysis in order to show the effectiveness of our algorithms in practice. We found that generally our distMIS and DFS algorithms produced slightly fewer time slots for each graph size than D-MGC in most of the cases in UDGs.

In the general graphs case, the results are presented in Figures 11 with 200 nodes, and in Figure 12 with 500 nodes. The horizontal axis represents the number of edges of the generated general graphs and the average degree of the nodes in the undirected graph separated by commas; the vertical axis represents number of generated time slots for FDLSP problem in logarithmic scale of base 10. Also, we provide the lower bound and the upper bound of time slots in the plots, beside the algorithms time slot numbers, drawn from the theoretical analysis in order to show the effectiveness of our algorithms in practice. Note that our DFS algorithm consistently produces 25% fewer time solts than D-MGC, and our distMIS algorithm also produces fewer time solts than algorithm D-MGC.

Although the number of time slots assigned by our algorithms distMIS and DFS compared to D-MGC algorithm are very close in the UDG case, our distMIS (DFS) algorithm has lower time complexity in both general graphs, which is $O(\Delta^4 + \Delta^3 log^* n)$ ($O(n)$), and in UDG, which is $O(\Delta log * n)$ ($O(n)$) compared to $O(n\Delta^2 + n^2 m)$ for D-MGC algorithm in general graphs (authors do not provide analysis in UDG). This is illustrated by Figure 13 for UDG case, where the x-axis represents the number of edges of a graph with a fixed number of nodes (100, 200 and 300 nodes), and the y-axis represents the number of rounds taken by our algorithm distMIS. Note that although nodes in distMIS algorithm need to communicate with neighbors on distance-3 (this requires 3 rounds instead of 1 for the competition phase to compute secondary MIS set, where 3 rounds is counted in the presented number of rounds), our distMIS algorithm requires much less than n rounds taking into consideration the hidden constants of the communication complexity. Also,the communication complexity for the general graphs case is represented in Figure 14 and in Figure 15, where the x-

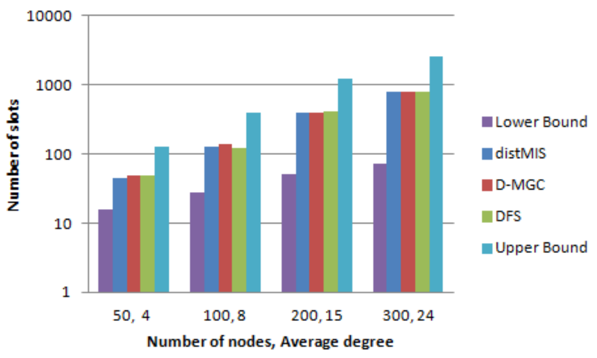

Figure 8: Time slot assignment in UDG: plan area = 15x15

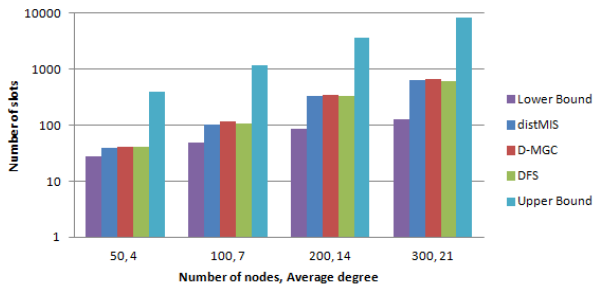

Figure 9: Time slot assignment in UDG: plan area = 17x17

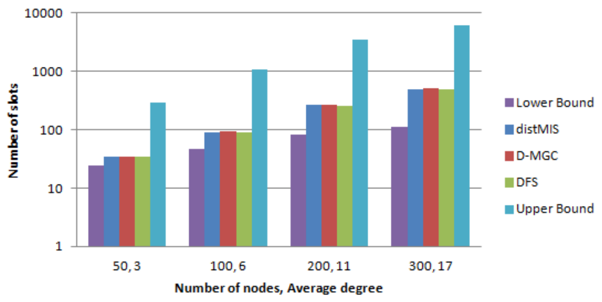

Figure 10: Time slot assignment in UDG: plan area = 20x20

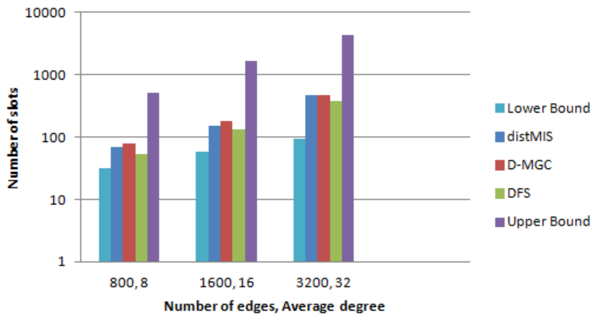

Figure 11: Time slot assignment in general graphs: 200 nodes

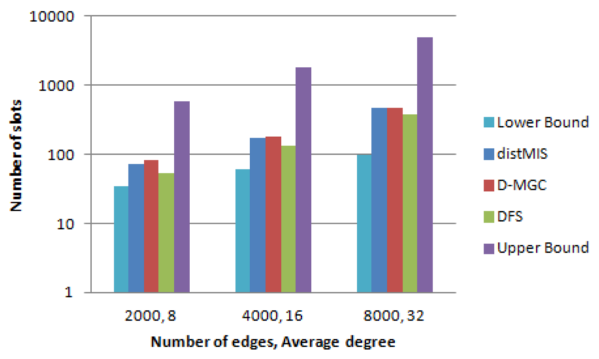

Figure 12: Time slot assignment in general graphs: 500 nodes

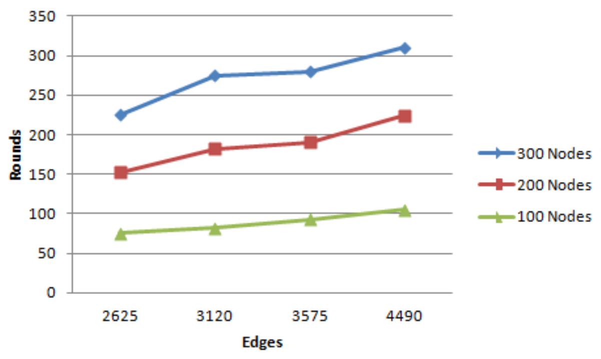

Figure 13: Number of communication rounds required by distMIS with varying number of edges in UDG.

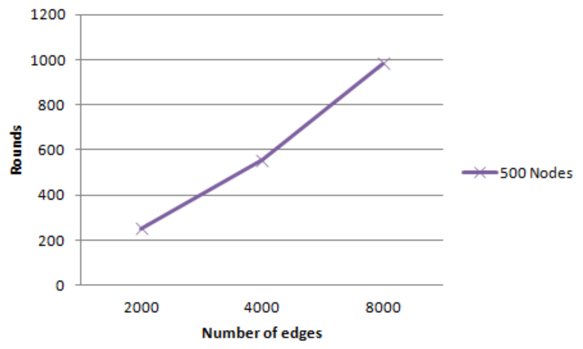

Figure 14: Number of communication rounds required by distMIS with varying number of edges in general graphs.

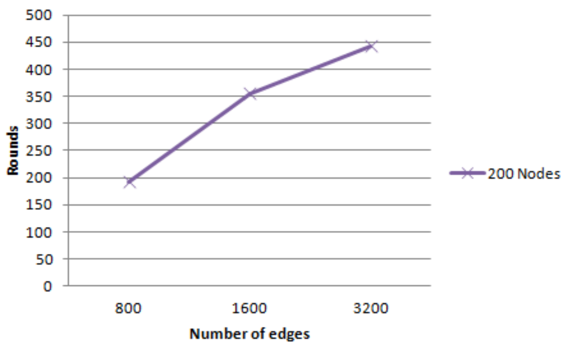

Figure 15: Number of communication rounds required by distMIS with varying number of edges in general graphs.

axis represents the number of edges of a graph with a fixed number of nodes (500 and 200 nodes respectively), and the y-axis represents the number of rounds taken by our algorithm distMIS.

Due to [2], the FDSLP problem is NP-complete; hence, we can only provide an upper bound, $2\Delta^2$, and an approximation ratio. But no precise formula for the number of colors in the sparse or the dense case can be provided accurately. To the best of our knowledge, we provide a tighter upper bound for the problem (due to [20] it is of order $O(\Delta^2)$ but not exact as we show). Also, the lower bound provided in [8] applies for acyclic graphs only; we provide a tighter lower bound that applies to all instances of graphs.

9 Conclusion

Full duplex link scheduling problem in wireless sensor networks FDLSP has been considered. The problem is formulated as distance-2 edge coloring a bi-directed graph such that the hidden terminal problem is avoided. Then, two distributed algorithms for FDSLP are presented. The first is a synchronous MIS based distributed algorithm and the second is an asynchronous distributed DFS based algorithm. Both of the algorithms have lower time complexity than the best known algorithm for the FDSLP problem in both growth bounded graphs and general graphs. Theoretical analysis and simulation results show the effectiveness of our approach compared to the best known algorithms for FDSLP problem.

In future work we consider fault-tolerance algorithms for DFSLP problem, where new sensors may join the network, or existing sensors may move or fail. And as a result, some of the links may disappear or new links may be established among the sensors in the network. Developing a fault tolerant algorithm for DFSLP is quite challenging task, since low energy consumption is crucial for the life time of the sensor network. In other words, the fault tolerant algorithm must incur low communication and computation cost to make the FDLSP assignment schedule feasible in the light of the new changes in the network topology.

References

[1] Thamer Alsulaiman, Sushil K. Prasad, and Alexander Zelikovsky. Distributed algorithms for tdma link scheduling in sensor networks. In *IPDPS Workshops*, pages 839–847. IEEE Computer Society, 2012.

[2] E. Arikan. Some complexity results about packet radio networks (corresp.). *Information Theory, IEEE Transactions on*, 30(4):681 – 685, July 1984.

[3] C. Avin, Y. Emek, E. Kantor, Z. Lotker, D. Peleg, and L. Roditty. Sinr diagrams: towards algorithmically usable sinr models of wireless networks. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 200–209. ACM, 2009.

[4] Baruch Awerbuch. Complexity of network synchronization. *J. ACM*, 32(4):804–823, 1985.

[5] C.L. Barrett, G. Istrate, V.S.A. Kumar, M.V. Marathe, S. Thite, and S. Thulasidasan. Strong edge coloring for channel assignment in wireless radio networks. In *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, pages 5 pp. –110, March 2006.

[6] M. Cheng and Li Yin. Transmission scheduling in sensor networks via directed edge coloring. In *Communications, 2007. ICC '07. IEEE International Conference on*, pages 3710 –3715, 2007.

[7] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Math.*, 86:165–177, January 1991.

[8] S. Gandham, M. Dawande, and R. Prakash. Link scheduling in sensor networks: distributed edge coloring revisited. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 4, pages 2492 – 2501 vol. 4, March 2005.

[9] T. Herman, S. Pemmaraju, and I. Pirwani. Oriented edge colorings and link scheduling in sensor networks. In *Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on*, pages 1 –6, 2006.

[10] Ted Herman and Sébastien Tixeuil. A distributed TDMA slot assignment algorithm for wireless sensor networks. In Sotiris Nikoletseas and José D. P. Rolim, editors, *Algorithmic Aspects of Wireless Sensor Networks*, volume 3121 of *Lecture Notes in Computer Science*, pages 45–58. Springer Berlin / Heidelberg, 2004.

[11] Christos Koufogiannakis and Neal E. Young. Distributed and parallel algorithms for weighted vertex cover and other covering problems. In *PODC '09: Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 171–179, New York, NY, USA, 2009. ACM.

[12] Sven O. Krumke, Madhav V. Marathe, and S. S. Ravi. Models and approximation algorithms for channel assignment in radio networks. *Wirel. Netw.*, 7:575–584, November 2001.

[13] F. Kuhn and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 joint workshop on Foundations of mobile computing*, pages 69–78. ACM, 2003.

[14] Fabian Kuhn. Weak graph colorings: distributed algorithms and applications. In *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures*, SPAA '09, pages 138–144, New York, NY, USA, 2009. ACM.

[15] Sandeep S. Kulkarni and Umamaheswaran Arumugam. Collision-free communication in sensor networks. In *Proceedings of the 6th international conference on Self-stabilizing systems*, SSS'03, pages 17–31, Berlin, Heidelberg, 2003. Springer-Verlag.

[16] L and Lovsz. Three short proofs in graph theory. *Journal of Combinatorial Theory, Series B*, 19(3):269 – 271, 1975.

[17] E. Lebhar and Z. Lotker. Unit disk graph and physical interference model: Putting pieces together. 2009.

[18] Z. Lotker and D. Peleg. Structure and algorithms in the sinr wireless model. *ACM SIGACT News*, 41(2):74–84, 2010.

[19] Kaveh Pahlavan and Allen H. Levesque. *Wireless information networks.* Wiley-Interscience, New York, NY, USA, 1995.

[20] S. Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wirel. Netw.*, 5:81–94, March 1999.

[21] Johannes Schneider and Roger Wattenhofer. A log-star distributed maximal independent set algorithm for growth-bounded graphs. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*, PODC '08, pages 35–44, New York, NY, USA, 2008. ACM.

[22] K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie. Protocols for self-organization of a wireless sensor network. *Personal Communications, IEEE*, 7(5):16 –27, October 2000.

[23] H. Tamura, K. Watanabe, M. Sengoku, and S. Shinoda. On a new edge coloring related to multihop wireless networks. In *APCCAS 2002*, volume 2, pages 357 – 360 vol.2, 2002.