A Simple Greedy Algorithm for Link Scheduling with the Physical Interference Model

Dejun Yang, Xi Fang, Nan Li and Guoliang Xue

***Abstract*—In wireless networks, mutual interference prevents wireless devices from correctly receiving packages from others and becomes one of the challenges in the design of protocols for wireless networks. Spatial-reuse Time Division Multiple Access (STDMA) has been used to cope with this problem. In this scheme, links are assigned to several time slots and in each slot all the links can transmit simultaneously. In this paper, we propose a greedy link scheduling algorithm to find a short schedule for a problem instance in the physical interference model. Our scheduling algorithm is inspired by the k-MAX-CUT algorithm in [13]. Experimental results show that our greedy algorithm can give a better schedule compared with the greedy algorithm in [3], with an improvement about 20%-30% when the density of links is high.**

1. Introduction

Wireless multi-hop radio networks such as ad hoc, mesh, or sensor networks attract considerable attentions in recent years due to their potential applications in various areas. Since these networks use a shared communication medium, one of the main problems in wireless networks is the decrease of capacity due to interference among multiple simultaneous transmissions [4], as shown in Fig. 1. To this end, STDMA-based link scheduling has been extensively studied [3][5][7][8][9][10][11][15]. The problem is to schedule a set of communication requests, represented by wireless links in general, into a number of time slots. In each time slot, only a subset of the required communication links can be scheduled. Clearly, it is ideal to find a schedule of minimum length in order to maximize the network throughput.

In this paper, we assume that time is slotted and synchronized(i.e. time division multiplexing). A link scheduling is to assign each link a set of time slots in which it will transmit. This schedule can guarantee all links in each slot can transmit simultaneously without causing unaccepted mutual interference.

In the literature, two main interference models have been proposed [4]: the *protocol* interference model and the *physical* interference model. In the former model, a communication between nodes u and v is successful if no other node within a certain interference range from v (the receiver) is simultaneously transmitting. In the latter one, a communication between u and v is successful if the Signal to Interference and Noise Ratio (SINR) at v is above a certain threshold, whose value depends on the desired channel characteristics. In this paper, we consider the physical interference model because it is more practical than the protocol interference model: two links which are far away from each other can interfere with each other. In addition, this paper considers two scenarios: unidirectional transmission and bidirectional transmission. In the former scenario, unaccepted interference should not be induced on directed links. In the latter one, both directions of a undirected link should not experience unaccepted interference.

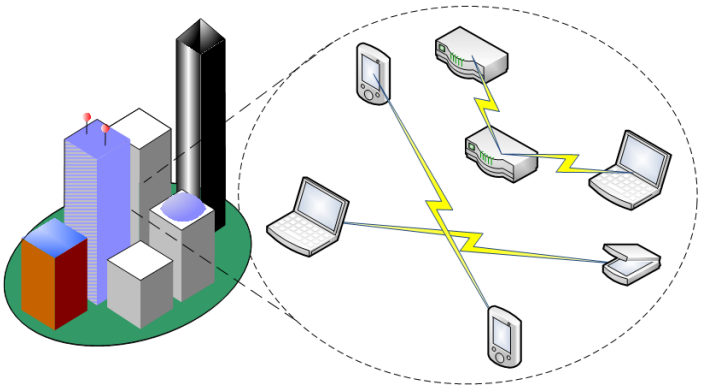


Fig. 1. A typical wireless network where there exist multiple simultaneous communication requests

Link scheduling in the context of spacial-reuse time division multiple access (STDMA) under the physical interference model has been shown to be an NP-hard problem [5]. Thus, scheduling algorithms often rely on heuristics that approximately optimize the throughput. In this paper, we present a polynomial time heuristic called k-**M**ax-**C**ut-based **G**reedy (**MCG**) Algorithm to solve this problem. Specifically, the main contributions are as follows:

1) **MCG** Algorithm can be applied to both homogeneous and heterogeneous networks.
2) **MCG** Algorithm can be applied to both unidirectional and bidirectional transmission scenarios.
3) **MCG** Algorithm is simple, and therefore suitable for implementation in real protocols. The complexity is $O(n^3 \log n)$, where n is the number of links.

The rest of the paper is organized as follows. In Section 2, we give an overview of related work in the literature. Then, we formally describe the network model and define the problem in Section 3. In Section 4, we present an efficient greedy algorithm for link scheduling problem. The performance of our algorithm is evaluated in Section 5. Section 6 concludes this paper.

2. Related Work

The problem of scheduling communication links in wireless networks in order to achieve maximum throughput has gained much interest in the research community. Since the spatial-reuse time division multiple access (STDMA) was first proposed in [12], STDMA-like algorithms have been studied for the problems under both pro-

Yang, Fang, Li and Xue are all with the CSE Dept. at Arizona State University, Tempe, AZ 85287. Email: {dejun.yang, xi.fang, nan.li.3, xue}@asu.edu. This research was supported in part by NSF grants CNS-0721803 and CCF-0830739. The information reported here does not reflect the position or the policy of the federal government.

tocol interference model [7][8][15] and physical interference model [3][5][7][9][10][11].

In the protocol model, a transmission from node x_i to node y_i is successful if and only if there are no other nodes within a certain range R of y_i transmitting at the same time. In [7], Jain *et al.* proposed a conflict graph based method for computing upper and lower bounds on the optimal throughput for given networks. The conflict graph is constructed by having a node represent each link. If two links conflict with each other in the original graph, we connect the corresponding nodes in the conflict graph with an edge. Clearly, the scheduling problem is closely related to coloring problem in the conflict graph. In [8], Kumar *et al.* obtained a constant-approximation algorithm by assigning the links in a first-fit manner. By using fractional coloring, Wang *et al.* [15] presented a $2\lceil 2\pi/arcsin\frac{c-1}{2c}\rceil$ approximation algorithm.

Unlike in the protocol model, the conflict relation is not binary in the physical interference model, that is, two distinct links might not corrupt each other, but the coexistence of a third one might make at least one link fail in transmission. This property makes link scheduling problems under physical interference model much more challenging.

To handle this special property, Jain *et al.* [7] constructed a weighted conflict graph, where the weight of a directed edge from vertex l_i to l_j is the fraction of the maximum permissible noise at the receiver of link l_j. They derived the lower and upper bounds on the optimal throughput by finding all the independent sets and cliques, which could have exponential time complexity.

Brar *et al.* proposed a computationally efficient scheduling algorithm referred to as *GreedyPhysical* in [3]. They also proved it is at most a factor $O(n^{1-\frac{2}{\psi(\alpha)+\epsilon}}(\log n)^{\frac{2}{\psi(\alpha)+\epsilon}})$ away from the optimal schedule, where ϵ is an arbitrary constant and $\psi(\alpha)$ is a constant depending on path loss exponent α. In *GreedyPhysical*, all the links are first sorted in decreasing order according to *interference number*. The *interference number* of link l_i is defined as the number of links, none of which can transmit simultaneously with l_i without conflict. Then, all the links are scheduled in a greedy manner. Each link is scheduled into the first w_i time slots such that each time slot is feasible, where w_i is the traffic demand on link l_i. If there are no such time slots, new empty ones are added at the end of the schedule. Though *GreedyPhysical* takes into account the measurement of the amount of interference generated by a link using *interference number*, it only considers binary relationship.

In [9], Moscribroda and Wattenhofer derived an upper bound $O(\log^4 n)$ on the number of time slots needed for scheduling a set of strongly connected communication requests in an arbitrary network by assigning different power levels to the links. Along the same line, they improved the complexity to $O(\log^3 n)$ in [10] and further to $O(\log^2 n)$ in [11].

The NP-completeness of link scheduling under physical interference models is formally proved in [5] by giving a reduction from *Partition* problem. To the best of our knowledge, this paper is also the only one that presented an approximation polynomial algorithm with a proved meaningful bound $O(g(L))$, where $g(L)$ is the number of the magnitudes of link lengths. When all the lengths of links are within a factor of 2, the approximation ratio becomes a constant. However, the interference model used in that paper is a modified physical interference model, where the noise is neglected. Moreover, the links in their model must be unidirectional.

TABLE I
NOTATIONS

$\mathcal{L}$	a set of communication requests
$\mathcal{L}'$	a subset of $\mathcal{L}$
$\mathcal{L}(i)$	a subset of links in $\mathcal{L}$ that are transmitting simultaneously with link l_i
N_0	ambient noise power level
n	number of links in $\mathcal{L}$
β	SINR threshold of links
α	path loss exponent
$P_r(x_i, y_j)$	received power at node y_j of link l_j from node x_i of link l_i
$P_t(x_i)$	transmission power of node x_i of link l_i
$I_l(l_j, l_i)$	interference induced on l_i by l_j
$I_s(\mathcal{L}', l_i)$	interference induced on l_i any set $\mathcal{L}'$ of links
$\mathcal{S}$	feasible schedule
S_t	set of links in t-th time slot
k_i	key value of link l_i
K	schedule length
$w(S_t, l_i)$	weight between link l_i and time slot t

3. Network Model and Problem Definition

In this section, we first describe the network model and notations. Then, we formally define the problem studied in this paper.

We consider the problem of scheduling communication requests of wireless nodes randomly distributed in the Euclidean plane. Request and link are used interchangeably in this paper. We assume the network is static, that is all the nodes are stationary. The set of links is denoted by $\mathcal{L} = \{l_1, l_2, \cdots, l_n\}$, where $l_i = (x_i, y_i)$ represents the communication request between nodes x_i and y_i. For unidirectional scenario, x_i is the transmitter and y_i is the receiver of link l_i. For bidirectional scenario, x_i and y_i are two end nodes of link l_i. We do not assume any specific direction of the link, because our algorithm can be applied on both unidirectional and bidirectional transmission scenarios. Also, the network model could be either homogeneous or heterogeneous, that is, we allow nodes to use different transmission powers and have different SINR thresholds. We assume each node is equipped with a single radio and there is only one available channel for all the links. Thus, simultaneous transmissions along two distinct links would interfere with each other. Each link l_i is associated with a weight $w_i \geq 1$, which indicates the traffic demand on the link. Let $d(x_i, y_j)$ be the Euclidean distance between nodes x_i and y_j. To be specific, $d(x_i, y_i)$ denotes the length of link l_i. Transmission power at node x_i is denoted by $P_t(x_i)$. The received power $P_r(x_i, y_j)$ at node y_j of a signal transmitted by node x_i is

$$P_r(x_i, y_j) = \frac{P_t(x_i)}{d^\alpha(x_i, y_j)}$$

where α is the path loss exponent, whose value is between 2 and 4 usually [4].

Choosing interference model has a strong impact on the complexity of link scheduling problem. For the $primary$ interference model, where two links interfere with each other only when they share a common endpoint, Hajek and Sasaki [6] proposed polynomial time algorithms using LP formulation. For the K-hop interference model defined by Sharma *et al.* in [14], the authors proved that the scheduling problem can be solved in polynomial time when $K = 1$. On the other hand, it has been proved that this problem is NP-hard under the K-hop model (when $K > 1$) [14], the $protocol$ model [7], and the $physical$ model [5]. In order to capture important aspects of real wireless networks, we adopt the Physical Interference Model (also called Signal-to-Interference-plus-Noise-Ratio (SINR) model sometimes) [4] in this paper. In this model, a message received by a node y_i can be correctly decoded if and only if the following condition is satisfied:

$$\frac{P_r(x_i, y_i)}{N_0 + \sum_{l_j \in \mathcal{L}(i)} P_r(x_j, y_i)} \geq \beta \tag{3.1}$$

where N_0 is the ambient noise power level, $\mathcal{L}(i)$ is a subset of links in $\mathcal{L}$ that are transmitting simultaneously with link l_i, and β is the minimum SINR required for a successful message decoding.

This paper considers two scenarios:

1) Unidirectional transmission: This transmission mode is often used in sensor networks and some video/voice service systems. This mode is also used in [5][9][10][11]. For this mode, the interference induced on l_i by l_j is given by $I_l(l_j, l_i) = P_r(x_j, y_i)$, and the total interference induced on l_i by all the links in $\mathcal{L}'$ is given by $I_s(\mathcal{L}', l_i) = \sum_{l_j \in \mathcal{L}'} I_l(l_j, l_i)$.
2) Bidirectional transmission: This is the most common transmission mode. Bidirectional application data transmission or reverse feedback (such as ACK and automatic retransmission request) requires systems to support bidirectional transmission. This mode is also used in [3]. For this mode, the interference induced on l_i by l_j is given by $I_l(l_j, l_i) = \max\{P_r(x_j, y_i), P_r(y_j, y_i), P_r(x_j, x_i), P_r(y_j, x_i)\}$, and the total interference induced on l_i by all the links in $\mathcal{L}'$ is given by $I_s(\mathcal{L}', l_i) = \max\{\sum_{l_j \in \mathcal{L}'} \max\{P_r(x_j, y_i), P_r(y_j, y_i)\}, \sum_{l_j \in \mathcal{L}'} \max\{P_r(x_j, x_i), P_r(y_j, x_i)\}\}$

Now we formally define the problem to be studied in this paper.

Definition 3.1. *A schedule is represented by $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_T\}$, where $\mathcal{S}_t$, $1 \leq t \leq T$, is a subset of links in $\mathcal{L}$ that are assigned into time slot t. We say a schedule $\mathcal{S}$ is* ***feasible*** *if and only if the following conditions are satisfied:*

- *For each $l_i \in \mathcal{L}$, it appears in at least w_i time slots and at most one time in each set $\mathcal{S}_t$.*
- *For each link $l_i \in \mathcal{S}_t$, the constraint (3.1) is satisfied.*

The cardinality $|\mathcal{S}|$ of a schedule $\mathcal{S}$ is called schedule length. A schedule length K is feasible if we can guarantee that there exists a feasible schedule of length K.

Definition 3.2. *Scheduling Problem aims to find a feasible schedule $\mathcal{S}^*$ such that it has the minimum schedule length among all the feasible schedules.*

For purpose of clarification, we assume that the traffic demand w_i is 1 for any link $l_i \in \mathcal{L}$. We will show that our algorithm can be easily extended to the case where the traffic demand is greater than 1. Table I lists all the frequently used notations in this paper.

4. Greedy Scheduling Algorithm

In this section, we present a computationally efficient heuristic algorithm referred to as k-**M**ax-**C**ut-based **G**reedy Algorithm (**MCG**) for *Scheduling Problem* under the physical interference model. This algorithm uses bisection scheme to find a feasible scheduling length of as small value as possible. For each possible value K, **MCG** Algorithm tests if K is feasible using Algorithm $Test(\mathcal{L}, K)$, which is listed as Algorithm 1. If we can get a feasible schedule of length K following our algorithm, we know that there must exist a feasible schedule of length less than or equal to K. Our Algorithm $Test(\mathcal{L}, K)$ outputs YES. Otherwise, it outputs NO. However, in this case, we cannot say that there does not exist a feasible schedule such that its length is less than or equal to K. Based on the returned value of $Test(\mathcal{L}, K)$, our algorithm refines the upper bound or the lower bound, and recalculates a tentative schedule length. This bi-section operation repeats until the tentative length is equal to either the upper bound or the lower bound. Note that this upper bound is always the upper bound of the optimal schedule length, while this lower bound is not always the lower bound of the optimal schedule length. The reason for the latter is that even if a feasible schedule can not be found by Algorithm 1, it does not necessarily indicate this length is infeasible.

To make this paper self-contained, we give the definition of k-Max-Cut in the following.

Definition 4.3 (k-Max-Cut)**.** *Given an undirected graph $G(V, E, w)$, where V denotes the set of vertices in the graph, E denotes the set of edges and w is an edge weight function so that $w_{uv} \geq 0$ is the weight of edge (u, v) for any $(u, v) \in E$. k-Max-Cut problem is to find k disjoint sets, $\{V_1, \ldots, V_k\}$, such that $\bigcup_{i=1}^{k} V_i = V$ and $\sum_{(u,v)\in E, u\in V_i, v\in V_j, i<j} w_{uv}$ is maximized.*

Scheduling problem with a given schedule length is similar to k-Max-Cut to a certain extent. In the k-Max-Cut problem, the total weight of all the edges is a constant. Maximizing the edge weight among the k disjoint sets is equivalent to minimizing the edge weight within the vertex sets. Scheduling links into k time slots also implicitly makes the interference within each time slot as small as possible. Nevertheless, the interference is not necessarily minimized. The similarity above inspires our **MCG** Algorithm.

Before we formally describe our greedy algorithm, we need the following definitions.

Definition 4.4. *A time slot S_t is feasible for a link l_i, if and only if after we add link l_i into $\mathcal{S}_t$, all the links can transmit successfully at the same time. In other words, the constraint (3.1) is satisfied for any link $l_j \in \{l_i\} \cup \mathcal{S}_t$.*

Definition 4.5 (Link Tolerance)**.** *The tolerance τ_i of a link l_i indicates how much interference can be tolerated before the SINR falls below the threshold β. It can be calculated by*

$$(\textit{Unidirection}) \quad \tau_i = \frac{P_r(x_i, y_i)}{\beta} - N_0 \tag{4.1}$$

or

$$(\textit{Bidirection}) \quad \tau_i = \frac{\min\{P_r(x_i, y_i), P_r(y_i, x_i)\}}{\beta} - N_0 \tag{4.2}$$

Definition 4.6 (Residual Link Tolerance)**.** *The residual tolerance τ_i' of a link l_i indicates that in the case link l_i is experiencing interference from all the links in $\mathcal{L}'$, how much interference can be tolerated before the SINR falls below the threshold β. It can be calculated by*

$$(\textit{Unidirection}) \quad \tau_i' = \tau_i - \sum_{l_j \in \mathcal{L}(i)} I_l(l_j, l_i) \tag{4.3}$$

or

$$(\textit{Bidirection})$$
$$\tau_i' = \min\{\tfrac{P_r(x_j,y_i)}{\beta} - \textstyle\sum_{l_j \in \mathcal{L}(i)} \max\{P_r(x_j,y_i), P_r(y_j,y_i)\} - N_0,$$
$$\tfrac{P_r(y_j,x_i)}{\beta} - \textstyle\sum_{l_j \in \mathcal{L}(i)} \max\{P_r(x_j,x_i), P_r(y_j,x_i)\} - N_0\} \tag{4.4}$$

Now we are ready to present the greedy algorithm in this paper.

Algorithm 1 Test($\mathcal{L}$, K)

1: $Input$: A set of communication requests $\mathcal{L}$ and the number of time slots K.
2: $Output$: Feasibility of a schedule $\mathcal{S}$ of length K.
3: $Initialize$:
4: $S_t = \emptyset$ for each time slot t;
5: **for** each link $l_i \in \mathcal{L}$ **do**
6: Compute $I_s(\mathcal{L} \setminus \{l_i\}, l_i)$;
7: Compute the tolerance τ_i using Equation (4.1) or (4.2);
8: Compute $k_i = \tau_i / \log(1 + I_s(\mathcal{L} \setminus \{l_i\}, l_i))$;
9: **end for**
10: Add all links in a queue Q in nondecreasing order;
11: **while** Q is not empty **do**
12: Pop the head link l_i in Q ;
13: **for** $t = 1$ to K **do**
14: **if** S_t is *feasible* for link l_i **then**
15: $w(S_t, l_i) = I_s(S_t, l_i)$;
16: **else**
17: $w(S_t, l_i) = \infty$;
18: **end if**
19: **end for**
20: **if** not all $w(S_t, l_i) = \infty$ **then**
21: Add link l_i into the time slot S_t with least $w(S_t, l_i)$;
22: **else**
23: **RETURN** NO;
24: **end if**
25: **end while**
26: **RETURN** YES.

The basic idea of Algorithm 1 is as follows. First, given a set $\mathcal{L}$ of communication requests and a tentative schedule length K, Line 3 to 9 initialize the interference induced by all the other links in $\mathcal{L}$, the tolerance and key value for all links. The fraction in the calculation of key's value comes from the intuition that a smaller tolerance or a larger interference indicates the link is more vulnerable. Additionally, we use $\log$ function to prevent the key from being too large when the interference is small. Note that $\frac{1}{\log(1+x)}$ is a monotonically decreasing function on the range $(0, \infty)$. Based on their key values, Line 10 puts all links in a queue Q in nondecreasing order, i.e. links which are more vulnerable are scheduled first. Then Line 12 pops the head link l_i from this queue. Line 14 checks whether link l_i can be added into any of these K time slots. If time slot S_t is *feasible* for link l_i, we set the weight between S_t and l_i to the interference induced by all the links in S_t. Otherwise, we let the weight be equal to infinity. If there exists at least one feasible time slot, Line 21 adds link l_i into time slot S_t with least $w(S_t, l_i)$. Otherwise, it returns NO at Line 23. We repeat this procedure from Line 11 to 25 for each link until Q is empty or one link cannot be scheduled. After all the links are scheduled, Line 26 returns YES.

Algorithm 2 MCG Algorithm

1: $Input$: A set of communication requests $\mathcal{L}$.
2: $Output$: A feasible schedule $\mathcal{S}$ under physical interference model.
3: $Initialize$: $LB = 1$, $UB = n$, $K = UB/2$;
4: **while** $K \neq LB$ and $K \neq UB$ **do**
5: **if** Test($\mathcal{L}, K$)=YES **then**
6: $UB = K$;
7: **else**
8: $LB = K$;
9: **end if**
10: $K = (LB + UB)/2$;
11: **end while**
12: **RETURN** the last feasible schedule $\mathcal{S}$.

Based on Algorithm 1, Algorithm 2 uses bisection method to find a feasible schedule length K with as small value as possible. At first, it sets $LB = 1$, $UB = n$, and $K = UB/2$. Then, Lines 4-11 search for the smallest value of K. If $Test(\mathcal{L}, K) = YES$, it means the upper bound of the optimal number of slots should be less than or equal to K. Thus we set UB to K. Otherwise LB is set to K. Now, we set $K = \frac{LB+UB}{2}$. We repeat this test until $K = LB$ or $K = UB$, since that means K is the smallest value.

To illustrate the idea on how the **MCG** Algorithm works, we present a simple example of 5-link network. We only consider unidirectional transmission in this example. For ease of exposition, we assume all the links have uniform power and same length. Set both N_0 and β to be 1. Let the received power at the receiver of each link be 6. Thus, we know that link tolerance for each link is 5. The interference between each pair of links is shown in the following matrix:

$$IM = \begin{pmatrix} 6 & 2 & 1 & 3 & 1 \\ 4 & 6 & 5 & 1 & 1 \\ 2 & 2 & 6 & 3 & 1 \\ 1 & 6 & 1 & 6 & 1 \\ 5 & 1 & 2 & 1 & 6 \end{pmatrix}$$

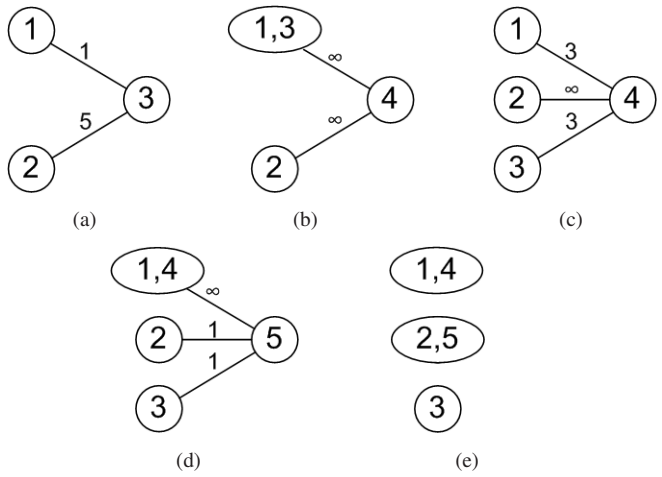


Fig. 2. A 5-link example

where $IM_{ij} = I_l(l_i, l_j)$, if $i \neq j$; $IM_{ij} = P_r(x_i, y_j)$, otherwise. We can easily compute $I_s(\mathcal{L} \setminus \{l_i\}, l_i)$ for each link, which are 12, 11, 9, 8 and 4 for links l_1, l_2, l_3, l_4 and l_5, respectively. Having all these values calculated, we sort the links in nondecreasing order in terms of the key value k_i. We get l_1, l_2, l_3, l_4 and l_5. Now, we show how our algorithm works step by step. We initialize LB to be 1 and UB to be 5. First, we set K to be $5/2 = 2$. Obviously, l_1 and l_2 should be scheduled into time slots S_1 and S_2 respectively. For link l_3, we compute the weights $w(S_1, l_3)$ and $w(S_2, l_3)$ as in Fig. 2(a). We assign link l_3 into time slot S_1. Next, we note that S_1, which has links l_1 and l_3, is not feasible for link l_4, because the SINR of link l_4 is $\frac{6}{1+(3+3)} < 1$. If we put link l_4 into time slot S_2, the SINR of link l_2 becomes $\frac{6}{1+6} < 1$, which is not enough to decode the message. Both the edge weights are set to ∞ as shown in Figure 2(b). $Test(\mathcal{L}, 2)$ returns NO. LB is updated to 2. We need to increase the value of K to $\frac{LB+UB}{2} = 3$. The first three links, l_1, l_2 and l_3, are assigned to time slots S_1, S_2 and S_3, respectively. To schedule link l_4, we compute the weights $w(S_1, l_4)$, $w(S_2, l_4)$ and $w(S_3, l_4)$ as shown in Figure 2(c). Since there is a tie between time slots S_1 and S_3, we arbitrarily schedule l_4 into S_1. Similarly, the weights $w(S_1, l_5)$, $w(S_2, l_5)$ and $w(S_3, l_5)$ are shown in Figure 2(d). Finally, we schedule link l_5 into time slot S_2. We have a feasible schedule $\{\{l_1, l_4\}, \{l_2, l_5\}, \{l_3\}\}$. This time, UB is updated to 3. Since $K = \frac{LB+UB}{2} = 3 = UB$, our algorithm terminates.

Theorem 4.1. *Given a set $\mathcal{L}$ of communication links, let n be the number of links. Then the time complexity of* **MCG** *Algorithm is $O(n^3 \log n)$.* □

PROOF. In Algorithm 1, it takes $O(n)$ time to initialize S_t, τ_i and k_i respectively, and $O(n^2)$ time to initialize I_i. $O(n \log n)$ time is required to sort links in Line 10. Now consider the time complexity of Line 4-11. A vector is used to hold all τ_i's. During each execution of the *while-loop*, it checks at most n time slots. For each time slot, Line 14 takes at most $O(n)$ operations to test whether some τ_i' falls below 0. Thus, each execution of the *while-loop* takes $O(n^2)$ time. Obviously, the *while-loop* is executed at most n times, as there are n links. Therefore, the time complexity of Algorithm 1 is $O(n + n^2 + n \log n + n^3) = O(n^3)$.

Algorithm 2 uses bisection scheme to test whether a tentative schedule length is feasible by invoking Algorithm 1, which executes at most $O(\log n)$ iterations. Thus the complexity of **MCG** Algorithm is bounded by $O(n^3 \log n)$. ■

Remark. Note that in the case where the traffic demand w_i is greater than one, our algorithm can be easily extended to accommodate this change. That is, we can simply make another $w_i - 1$ copies of each original link and then apply **MCG** Algorithm.

5. SIMULATION RESULTS

Since the *GreedyPhysical* Algorithm [3] is the only efficient algorithm for *Scheduling Problem* under physical interference model, we evaluate our algorithm by comparing it with GreedyPhysical in several sets of simulations.

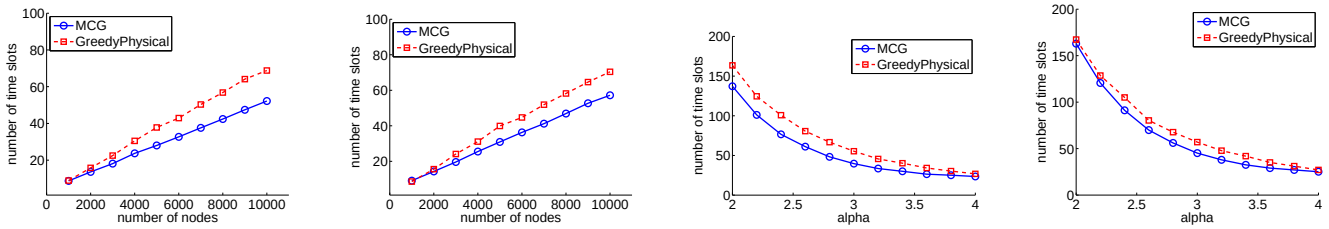
A. Simulation setup

In this section we evaluate the performance of **MCG** Algorithm. We set up the simulations for both unidirectional transmission mode and bidirectional transmission mode. Though *GreedyPhysical* is not originally designed for unidirectional mode, we note that it can be easily extended to apply on this mode. For each mode, we present four sets of simulation results. Two of them compare the schedule lengths obtained from the two algorithms on homogeneous networks and heterogeneous networks as the density of links increases. The other two compare the schedule lengths as the value of α varies. The links were generated in the following way. All the links were randomly distributed in a rectangular region of 1000 by 1000. The length of each link is randomly chosen between 1 and 30. The number of nodes (twice the number of links) varies from 1000 to 10000 with step size 1000. The SINR threshold β was set to 10 and the environment noise was 10^{-9}w. The path loss exponent α was set to 3.5 for simulations of increasing node density, and varied from 2.0 to 4.0 with step size 0.2 for simulations of increasing value of α. For homogeneous networks, the transmit power was 200w. For heterogeneous networks, it was one of the three values 150w, 200w and 250w. For each set of simulations, we ran simulations ten times and averaged the results.

B. Simulation results

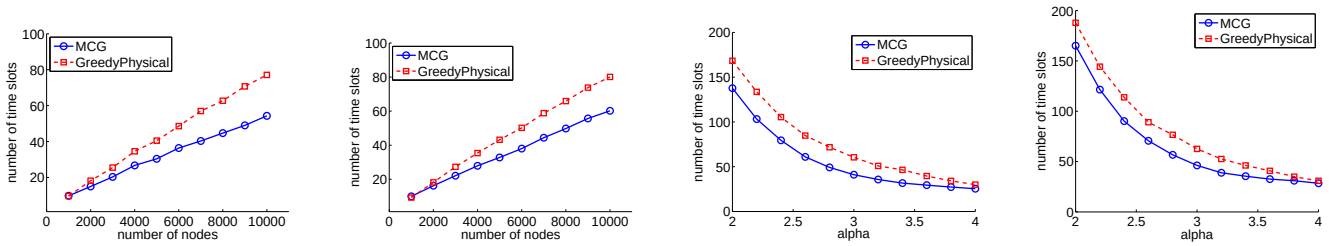
Figure 3(a) and 3(b) show the average schedule lengths in the unidirectional transmission mode with the increase in the density of nodes. When the number of nodes is greater than 2000, the lengths obtained by **MCG** Algorithm are less than those by *GreedyPhysical* Algorithm [3] by 20%-30%. This is because *GreedyPhysical* Algorithm uses interference number as metrics. When the node density is large, this kind of metrics will result in significant inaccuracy. In **MCG** Algorithm, the interference of all nodes is considered together. This allows **MCG** to choose a potentially better time slot.

Figure 4(a) and 4(b) show the average schedule lengths in the bidirectional transmission mode with the increase in the density of nodes. Similar with unidirectional transmission mode, when the number of nodes is greater than 2000, the schedule lengths produced by **MCG** Algorithm are less than those by *GreedyPhysical* Algorithm [3] by 20%-25%.



(a) Homogeneous networks with increasing nodes density (b) Heterogeneous networks with increasing nodes density (c) Homogeneous networks with increasing alpha (d) Heterogeneous networks with increasing alpha

Fig. 3. Comparison of schedule length for unidirectional transmission mode



(a) Homogeneous networks with increasing nodes density (b) Heterogeneous networks with increasing nodes density (c) Homogeneous networks with increasing alpha (d) Heterogeneous networks with increasing alpha

Fig. 4. Comparison of schedule length for bidirectional transmission mode

Comparing Fig. 3 with Fig. 4, we can find the schedule lengths in Fig. 3 are less than those in Fig. 4. This is because according to the definition in Section 3, we know that the mutual interference between two links in unidirectional transmission scenario is smaller than that in bidirectional transmission scenario. Thus fewer time slots are required in the former scenario.

As shown in Fig. 3(c)(d) and Fig. 4(c)(d), we observe that, with the increase in the value of α, the gap between these two algorithms reduces. This is because a smaller α indicates more interference on every link. However, considering interference number as metrics, *GreedyPhysical* Algorithm introduces more inaccuracy when the interference is large.

6. Conclusion

In this paper, we have studied the Link Scheduling problem under the physical interference model with the goal of minimizing the schedule length. We analyzed the similarity between link scheduling with fixed schedule length K and K-Max-Cut problem. More important, we presented a simple and efficient k-Max-Cut-based Greedy Algorithm (**MCG**). Experimental results show that the improvement is about 20%-30% compared with the Greedy Algorithm proposed in [3].

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci; Wireless sensor networks: a survey; *COMNET*; Vol. 38(2002), pp. 393–422.

[2] M. Alicherry, R. Bhatia and L.E. Li; Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh network; *Mobicom'05*, pp. 58-72, 2005.

[3] G. Brar , D. M. Blough and P. Santi; Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks;*Proceedings of the 12th annual international conference on Mobile computing and networking*, September 23-29, 2006, Los Angeles, CA, USA.

[4] P. Gupta and P.R. Kumar; The Capacity of Wireless Networks; *IEEE Trans. Info. Theory*, Vol. 46, No. 2, pp.388-404, 2000.

[5] O. Goussevskaia, Y.A. Oswald and R. Wattenhofer; Complexity in geometric SINR; *MobiHoc '07*, pp. 100-109, 2007.

[6] B. Hajek and G. Sasaki; Link scheduling in polynomial time; *IEEE Transactions on Information Theory*, Vol. 34, 1988.

[7] K. Jain, J. Padhye, V.N. Padmanabhan and L. Qiu; Impact of interference on multi-hop wireless network performance; *MobiCom '03*, pp. 66-80, 2003.

[8] V.S.A. Kumar, M.V. Marathe, S. Parthasarathy and A. Srinivasan; Algorithmic aspects of capacity in wireless networks; *SIGMETRICS Perform. Eval. Rev.*, pp. 133-144, 2005.

[9] T. Moscribroda and R. Wattenhofer; The complexity of connectivity in wireless networks; *Infocom'06*, pp. 1-13, 2006.

[10] T. Moscribroda, R. Wattenhofer and A. Zollinger; Topology control meets SINR: the scheduling complexity of arbitrary topologies; *Mobihoc'06*, pp. 310-321, 2006.

[11] T. Moscribroda, Y.A. Oswald and R. Wattenhofer; How optimal are wireless scheduling protocols?; *Infocom'07*, pp. 1433-1441, 2007.

[12] R. Nelson and L.Kleinrock; Spatial-TDMA: a collison-free multihop channel access protocol; *IEEE Trans. on Communiction*, Vol. 33, pp. 934-944, 1985.

[13] S. Sahni and T. Gonzales; P-Complete approximation problem; *In Proc. 32nd Ann. ACM Symp. on the Theory of Computing*.

[14] G. Sharma, R.R. Mazumdar and N.B. Shroff; On the complexity of scheduling in wireless networks; *Mobicom'06*, pp. 227-238, 2006.

[15] W. Wang, Y. Wang, X. Li, W. Song and O. Frieder; Efficient interference-aware TDMA link scheduling for static wireless networks; *MobiCom '06*, pp. 262-273, 2006.