The HKU Scholars Hub The University of Hong Kong 香港大學學術庫

Title	Scheduling wireless links with SINR constraints
Author(s)	Hua, Qiangsheng; 華強勝
Citation	
Issue Date	2009
URL	http://hdl.handle.net/10722/56036
Rights	unrestricted

SCHEDULING WIRELESS LINKS WITH SINR CONSTRAINTS

by

Hua Qiangsheng

华强胜

B. Eng.; M. Eng. C.S.U., P. R. China

A thesis submitted in partial fulfillment of the requirements for

the Degree of Doctor of Philosophy

at The University of Hong Kong.

July 2009

香港大學圖書館 HKU Libraries

Abstract of thesis entitled

SCHEDULING WIRELESS LINKS WITH SINR CONSTRAINTS

Submitted by

Hua Qiangsheng

For the degree of Doctor of Philosophy

at The University of Hong Kong

in July 2009

This dissertation investigates three link scheduling problems under the physical interference model, or the SINR model. The first problem is called minimum frame length link scheduling for arbitrary link topologies (MFSAT): Given a set of arbitrarily constructed links over arbitrarily located nodes on a plane, schedule all these links with the minimum number of timeslots such that each link appears in at least one timeslot. The requirement for this problem is that concurrently scheduled links must satisfy the SINR constraints. The second problem is called minimum length link scheduling for arbitrary link topologies (MLSAT): Different from the MFSAT problem where each link has only a unit traffic demand (one packet to transmit), each link in the MLSAT problem may have non-unit traffic demands, namely, we need to schedule all the links with the minimum number of timeslots such that each link is scheduled at least the number of times as specified by its traffic demands. The third problem is called minimum frame length link scheduling for a data gathering tree topology (MFSTT): Given a set of arbitrarily located

香港大學圖書館 HKU Libraries

nodes on a plane, connect these nodes as a data gathering tree towards the sink node. The objective of this problem is to construct the tree such that all the links in this topology can be scheduled using a minimum number of timeslots. The requirement for this problem is the same as that for MFSAT.

We have developed heuristic, exact and approximate link scheduling algorithms for the MFSAT problem. For the heuristic algorithm, we have designed a novel maximum directed cut based scheduling framework called MDCS. Both theoretical analyses and simulation results have shown that the MDCS scheduling framework significantly outperforms all the sate-of-the-art heuristic link scheduling algorithms in terms of the scheduling lengths. By applying an exact algorithm for the set cover problem, we have designed an exact algorithm called ESA_MFSAT for the MFSAT problem. Finally, based on the ESA_MFSAT algorithm, we give the first polynomial time polynomial space approximate link scheduling algorithm for the MFSAT problem with approximation ratio $O(n/\log n)$ where n is the number of the links.

For the MLSAT problem, we first transform it into the set multi-cover problem, and then we give a first known exact algorithm for the set multi-cover problem. This exact algorithm can solve the MLSAT problem in $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space where t means the maximum traffic demand. Based on the proposed exact algorithm, we present the first polynomial time polynomial space approximation algorithm for the MLSAT problem with an approximation ratio independent of the links' lengths.

Finally, for the MFSTT problem, we have generalized a nonlinear power assignment based link scheduling algorithm to cover also wideband networks. We prove that the asymptotic poly-logarithmic scheduling length is achieved

香港大學圖書館 HKU Libraries

at the expense of the exponential total power consumption in the number of the nodes. Then, by using the MDCS scheduling framework, we show that connecting the nodes with a minimum spanning tree algorithm rather than an iterative nearest component connector algorithm can significantly reduce the scheduling length.

(An abstract of exactly 501 words)

香港大學圖書館 HKU Libraries

Declarations

I Declare that this thesis represents my own work, except where due acknowledgements is made, and that it has not been previously included in a thesis, dissertation or report submitted to this University or to any other institution for a degree, diploma or other qualifications.

Hua Qiangsheng

July 2009

_____

香港大學圖書館 HKU Libraries

Acknowledgements

First of all, this thesis could not have been completed without the continuous support and guidance of Professor Francis C.M. Lau. I'm profoundly grateful for the invaluable patience and freedom that Francis offers for my research during the past five years. I really appreciate his kind understanding, encouragement and help in both my studies and personal life. It's my fortune to have Prof. Francis as my PhD supervisor.

Second, I'd like to thank Dr. Choli Wang for his kind care in my SRG life. Thanks also go to Prof. Richard B. Tan and Prof. Roger Wattenhofer for their kind suggestions in my research. I'm also grateful for all the professors and researchers whom I have email correspondences with.

Third, I'd like thank my co-author Yu Dongxiao. Part of our collaborated work appears in Chapter 5 of the thesis.

Fourth, I'd like to thank all the staff from HKU library, the computer science department General Office and Technical Support Office. It's their dedicated and diligent work that makes HKU and the department a comfortable and pleasant place for doing research.

Last, I'm very lucky to know many colleagues and friends in HKU. They are: Wang Rui, Wenzhang, Simai, Zheng Yuan, Luo Zhi, Xiangli, Xiaolei, Chen Lin, Tianqi, Tianchi, Roy, Fangwei, Luo Yang, Haisheng, Wenxia, Hongxing, Dongxiao, Di Sheng, Haoyu, Chen Zhuo, Wang Zhen, Yan Li, Yongjian and many others. Thanks to all of you!

香港大學圖書館 HKU Libraries

Contents

香港大學圖書館 HKU Libraries

香港大學圖書館 HKU Libraries

香港大學圖書館 HKU Libraries

List of Figures

香港大學圖書館 HKU Libraries

List of Tables

香港大學圖書館 HKU Libraries

Chapter 1 Introduction

In this thesis, we will investigate three related minimum (frame) length wireless link scheduling problems which are to use the minimum number of timeslots to schedule all these links such that the simultaneously scheduled links must fulfill some interference model. Here by the interference model, we mean the criteria for determining the links that can be scheduled in the same timeslot. Obviously the interference model plays a fundamental role in the minimum length link scheduling problems. So in order to further explore our research topics, we need to give a brief survey of the interference models that arise in various literatures.

1.1 Interference Models

1.1.1 Graph-based interference models

In this section, we will introduce six graph-based interference models. Here by a graph-based interference model, we mean that each constraint only involves two wireless links, i.e., it is a binary constraint model. Among the six interference models, only the first imposes constraints on a single wireless node, while the other five models impose constraints on each pair of wireless links.

The first binary constraint model is called the primary interference model [49,111,119], or the node-exclusive interference model [71]. This model restricts that a wireless node can not perform two operations at the same time, such as receiving from two transmitters, transmitting to two receivers or receiving and sending at the same time. These constraints are due to the

香港大學圖書館 HKU Libraries

following two facts: The first is the half-duplex constraint for the single radio transceiver and the second is for the point-to-point traffic requirement which means that each packet is addressed to a single receiver. In this model, only non-adjacent links which form a link matching can be concurrently scheduled.

The second binary constraint model arises due to the so called secondary interference caused by the broadcast nature of the wireless medium [111]. Given two single hop wireless transmissions one is from node *i* to node *j* and the other is from node *k* to node *l*, we can tell that these two transmissions can not be simultaneously scheduled if at least one of the receivers is within the transmission range of another link's sender. Obviously, the secondary interference model prevents the capture effect of the wireless transceiver. Here by the capture effect, we mean the ability of the wireless transceiver that can correctly receive the strong signal from one transmitter despite the interferences caused by the other transmitters. From this we can see that capture effect is beneficial in physical reality since it can increase the network throughput by allowing more potential transmissions in each timeslot.

The third binary constraint model we want to introduce is called the protocol interference model that is given in [23]. For any wireless transmission which is from node *i* to node *j*, in order to make the receiver *j* successfully receive the packet from *i*, the distance from any transmitter *k* of the other simultaneously scheduled links (transmissions) to node j must be at least a factor $(1+\Delta)$ higher than the distance from node *i* to node *j* (the link's length). Here the positive parameter Δ is a specified guard zone value to prevent the neighboring nodes from transmitting at the same time.

香港大學圖書館 HKU Libraries

Another protocol interference model has been introduced in [64]. In this model, in order to make the receiver j successfully receive the packet from i, the distance from any transmitter k of the other concurrently scheduled links to node j must be at least a factor $(1+\Delta)$ larger than the distance from node k to its corresponding receiver (node l).

The fifth binary constraint model which has been used in [21] is called the transmitter interference model (Tx-model). For two transmissions with transmitter i and j respectively , in order to make sure the intended recipient of node i correctly receive the packet, the distance between i and j must be at least a factor $(1+\Delta)$ larger than the sum of the transmission ranges of sender i and sender j.

The sixth binary constraint model is called the Distance-K interference model (K is a positive integer) [21,22,112]. This model requires that two links must be at least distance-K apart to ensure simultaneous transmissions. Here by the distance of two links, we mean the least number of hops between an incident node of the first link and an incident node of the second link. Depending on different K values, this model can incorporate a large class of interference models. For example, when K=1, it becomes the primary interference model; when K=2, it becomes the transmitter-receiver interference model (Distance-2 interference model) which has been used in many network protocols, such as 802.11 DCF (Distributed Coordinated Function).

香港大學圖書館 HKU Libraries

1.1.2 SINR models

In this section, we will discuss the interference models that are based on the signal-to-interference-plus-noise ratios (SINR). This means that only all the SINR values at the links' receivers are above some threshold values can these links be successfully scheduled in the same timeslot. Different from graph-based interference models, since each constraint in the physical interference models covers any subset of all the links, they are called global constraints models. It is commonly believed that the SINR models are more realistic than the graph-based interference models but using these models also pose much more challenges on the link scheduling problems due to the cumulative interferences effect from all the other transmitters [50-52,54,80].

The SINR ratio at the receiver of a link i can be represented as [49,121]

$$SINR_i = \frac{g_{ii} \cdot p_i}{n_i + \sum_{j=1, j \neq i} g_{ij} \cdot p_j} \geq \frac{\beta}{m} \tag{1.1}$$

To delve into the details of this model, an explanation of the used parameters is in order: p_i denotes the transmission power of link i 's transmitter i_s; n_i is the background noise at link i ' s receiver i_r; g_{ii} and g_{ij} are the link gain (wireless signal propagation attenuation) from i_s to i_r, and that from the transmitter j_s of link j to i_r, respectively; β is the SINR threshold which is larger than or equal to 1; m stands for the processing gain which equals the ratio of the chip rate to symbol rate or the information bit rate. Here each message or information consists of symbols and each symbol is encoded (spreaded) into a pseudorandom sequence of chips. Thus the chip rate is normally larger than the symbol rate. The processing gain can be

香港大學圖書館 HKU Libraries

regarded as the signal's ability to fight the interferences. So the larger the processing gain, the more links can be tolerated in the same timeslot. The processing gain is larger than 1 in (ultra-)wideband networks, and it equals 1 in narrowband networks. Throughout this thesis, we assume m=1 (narrowband networks) except Chapter 6.

Since we do not consider fading effects and possible obstacles in wireless transmissions, the link gain can be represented by an inverse power law model of the link length, i.e., $g_{ii} = 1/d^{\alpha}(i_s, i_r)$ and $g_{ij} = 1/d^{\alpha}(j_s, i_r)$. Here $d(,)$ is the Euclidean distance function, and α represents the path loss exponent which is equal to 2 in free space, and varies between 2 and 6 in urban areas. By plugging into these equations and the m value, the SINR model becomes:

$$SINR_i = \frac{p_i / d^{\alpha}(i_s, i_r)}{n_i + \sum_{j=1, j\neq i} p_j / d^{\alpha}(j_s, i_r)} \geq \beta \qquad (1.2)$$

This is the same as the physical interference model proposed in [23].

1.2 Relationships between Graph-based Interference Models and the SINR Model

In this section, we will discuss some interesting relationships between the two protocol interference models and the physical interference model. We will use these results in the following chapters.

We consider two links, one is called link i with transmitter i_s and receiver i_r, the other is called link j with transmitter j_s and receiver j_r. In order to successfully schedule link i, according to inequality (1.2), we have

香港大學圖書館 HKU Libraries

$$SINR_i = \frac{p_i / d^{\alpha}(i_s, i_r)}{n_i + p_j / d^{\alpha}(j_s, i_r)} \geq \beta$$

From this inequality, we can obtain

$$\frac{d(j_s, i_r)}{d(i_s, i_r)} > \beta^{\frac{1}{\alpha}} \cdot (\frac{p_j}{p_i})^{\frac{1}{\alpha}} \tag{1.3}$$

Now if $p_i = p_j$ (we call this as constant power assignment), inequality (1.3) becomes

$$d(j_s, i_r) > \beta^{\frac{1}{\alpha}} \cdot d(i_s, i_r) \tag{1.4}$$

This is the same as the first protocol interference model introduced in [23]. Similarly, if $p_i = \rho \cdot d^{\alpha}(i_s, i_r)$ and $p_j = \rho \cdot d^{\alpha}(j_s, j_r)$ (we call this as linear power assignment), inequality (1.3) becomes

$$d(j_s, i_r) > \beta^{\frac{1}{\alpha}} \cdot d(j_s, j_r) \tag{1.5}$$

This is the same as the second protocol interference model introduced in [64].

1.3 Reasons to Choose the SINR Model

In this section, we will give some reasons for choosing the physical interference model rather than the graph-based interference models.

The first reason is that, compared with the SINR model, all the graph-based interference models did not take the cumulative interferences effect into account. This can be seen in the following example of Figure 1-1. In this example, there are seven links whose lengths are all 1. In addition, all the distances from the other six links' transmitters to the transmitter i_s and the receiver i_r of link i are 3 and 4, respectively. We also set the distances of link

香港大學圖書館 HKU Libraries

i's transmitter i_s to all the other six links' receivers are 2.5. Now according to the five graph-based interference models (the secondary interference model, the two protocol interference models, the transmitter interference model and the Distance-K interference model), these seven links can be simultaneously scheduled in the same timeslot (We suppose these seven links are distance-K apart in the Distance-K interference model). However, if we assume all the six links' (link j,k,l,p,q and h) transmitters employ the same transmission power which equals to $6 \cdot p_i$, and we set the path loss exponent α =3, the threshold β =2 and all the background noises values are 0, then the SINR value at the receiver i_r of link i is:

$$SINR_i = \frac{p_i / 1^3}{0 + 6 \cdot 6 \cdot p_i / 4^3} \doteq 1.78 < 2$$

From this we can conclude that link i can not be successfully scheduled. This example indicates that the power assignment strategies and the aggregate interferences effect of simultaneous transmissions may subvert a communication request which might otherwise appear successful under the graph-based interference models.

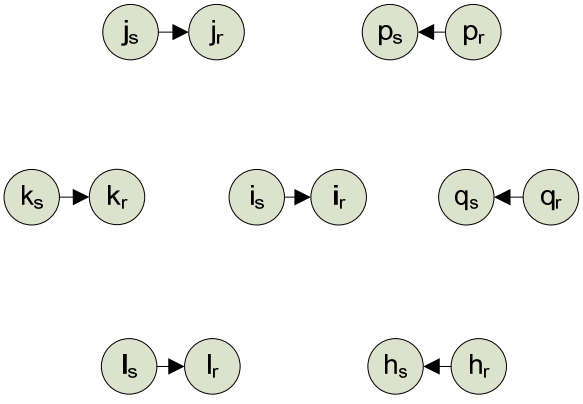


Figure 1-1: An example of seven links centered at link i

香港大學圖書館 HKU Libraries

The second reason to choose the SINR model instead of the graph-based interference models is derived from an observation in [57]. Specifically, the authors in [57] proved that the scheduling algorithm under the SINR model can achieve the throughput that can surpass the theoretically attainable throughput upper bound under the graph-based interference models. The simulation results to compare the throughput by using these two kinds of interference models can be found in [10,11].

From the above analyses, we can conclude that we should employ the physical interference model rather than the simple graph-based interference models in terms of both improving the network throughput capacity and guaranteeing correct packet receptions for all the wireless transmissions.

1.4 System Model and Problem Definitions

1.4.1 System model

Throughout the thesis, we have the following assumptions of the wireless network: (1) All the wireless nodes are arbitrarily located on a plane, and each node is equipped with an omni-directional antenna; (2) We assume all the nodes are stationary; (3) we assume a single channel which means all the simultaneously scheduled links interfere with each other; (4) The wireless transceivers work on a half-duplex mode, which means each node can not send to or receive from more than one node, nor to receive and send at the same time (this corresponds to the primary interference model); (5) we assume the link capacity is fixed, which means increasing the transmission power only increases the sender's transmission range but not its capacity; (6)

香港大學圖書館 HKU Libraries

we assume time is slotted with equal durations which means that each packet can not be further divided into smaller units to transmit.

1.4.2 Problem definitions

To begin this section, we first give an explanation of some terms: As mentioned earlier, by a wireless link, we mean a wireless transmission comprised by a source node (transmitter) and a destination node (receiver); If we regard the wireless link as an edge in a graph, then a 'link matching' is just a matching in the graph; Similarly, a 'link maximum (maximal) matching' or a 'maximum (maximal) link matching' is just a maximum (maximal) matching in the graph. For the brevity of presentation, we can also use (maximum) (maximal) matching directly since it will not cause any confusion. By a link independent set, we mean a set of links that can be concurrently scheduled in the same timeslot under some interference model. Also any subset of a link independent set is also a link independent set.

Now we define the minimum length wireless link scheduling (MLS) problem. First we assume each link can transmit at most one packet in each timeslot. Second, by the scheduling length, we mean the totally used timeslots to schedule all the packets. Third, if all the links have the same number of packets to be transmitted, we call it uniform traffic (link) demands (traffic requirements), otherwise we call it non-uniform traffic (link) demands. Now by the minimum length wireless link scheduling problem, we mean that, for a set of wireless links with given traffic demands, we need to use the minimum number of timeslots to schedule all the packets subject to the interference constraints.

香港大學圖書館 HKU Libraries

With the above definitions, we can define the minimum frame length link scheduling problem. First, by the frame length T, we mean the scheduling length to schedule all the links such that each link has transmitted one packet. Then the minimum frame length link scheduling problem is just the minimum length link scheduling problem with a unit traffic demand. In the minimum frame length link scheduling scenario, we just need to repeat the scheduling sequences in the subsequent frames, i.e., $X_{i,t} = X_{i,t+kT}$ ($0 < t \leq T$; k is a positive integer; $X_{i,t}$ equals 1 if link i transmits in timeslot t and 0 otherwise).

From these definitions, we can see that minimum frame length link scheduling problem is only a special case for the minimum length link scheduling problem. In addition, repeating the minimum frame length link scheduling results in each frame can not guarantee the minimum length link scheduling result with uniform traffic demands. Finally, even iteratively applying the minimum frame length link scheduling algorithm to schedule all the links with uniform or non-uniform traffic demands can not guarantee the minimum length link scheduling results for the given traffic requirements. For example, given four links {1,2,3,4} where the maximum link independent sets are {1,2,3},{1,2,4} and {3,4}, the minimum frame length link scheduling result could be to schedule {1,2} in the first timeslot and then to schedule {3,4} in the second timeslot. Now suppose each link has two packets to transmit. Repeating the minimum frame length link scheduling results would lead to the scheduling length with four timeslots. However, the minimum length scheduling result for these eight packets is three timeslots {{1,2,3},{3,4},{1,2,4}}. Moreover, we can first apply the minimum frame length

香港大學圖書館 HKU Libraries

link scheduling algorithm to schedule four packets and then to apply this algorithm again to schedule the remaining four packets. The results could be to use two timeslots ({1,2,3},{4}) in the first minimum frame length link scheduling and then to use another two timeslots ({1,2,4},{3}) in the second minimum frame length link scheduling. So this result is not optimal for minimum length link scheduling, either.

We now give the definitions of the three studied problems in this thesis. The first is called the minimum frame length link scheduling problem for arbitrary link topologies (MFSAT); the second is called the minimum length link scheduling problem for arbitrary link topologies (MLSAT); the third is called the minimum frame length link scheduling problem for a data gathering tree topology (MFSTT).

The MFSAT problem: Given n links which are arbitrarily constructed over arbitrarily located nodes on a plane and suppose each link has a unit traffic demand, we need to assign each link's transmitter a power level and a timeslot, such that all the links scheduled in the same timeslot satisfy the SINR constraints and the total number of used timeslots for transmitting all the packets is minimized.

The MLSAT problem: The only difference between the MLSAT and the MFSAT problem is that, each link in the MLSAT problem may have non-unit traffic demands. In this case, in order to minimize the totally used timeslots for transmitting all the packets, we need to assign each link's transmitter a set of power levels and timeslots such that each power level corresponds to a timeslot.

香港大學圖書館 HKU Libraries

The MFSTT problem: Given n nodes arbitrarily located on a plane, we need to connect these nodes to form a data gathering tree towards the sink node such that the number of timeslots used to schedule all the links under the SINR model is minimized.

From the three problem definitions, we can easily see that: (1) the MFSAT problem is a special case for the MLSAT problem; and (2) if the tree topology has been constructed, the MFSTT problem becomes a special case for the MFSAT problem which has been identified as a prominent open problem in [52]. However, as we will see in Chapter 7, how to construct this tree topology plays a very important role in the scheduling length.

1.5 Thesis Organization

The structure of this thesis is organized as follows. We will first review the state-of-the-art heuristic minimum (frame) length link scheduling algorithms and analyze their time complexities and inefficiencies in Chapter 2. In Chapter 3, we will give a maximum directed cut based scheduling framework for the MFSAT problem. Then we will present both exact and approximate link scheduling algorithms for the MFSAT problem in Chapter 4. In Chapter 5, we give both an exact and an approximate algorithm for the MLSAT problem. We then turn to investigating the MFSTT problem in Chapters 6 and 7. Specifically, we will discuss an elegant nonlinear power assignment based link scheduling algorithm together with its total power consumption analysis in Chapter 6, then a joint topology construction and link scheduling algorithm using the MDCS scheduling framework is given in Chapter 7. We finally conclude this thesis with some future work in Chapter 8.

香港大學圖書館 HKU Libraries

Note that most of the results in this thesis have been published or are in press. The details are as follows: Chapters 2, 3 and 7 have been summarized as a brief survey chapter in “Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice” [138]. Chapter 4 has been published in the DIALM-POMC 2008 workshop [50]. Chapter 5 is now in press for “Theoretical Computer Science” [139]. Chapter 6 has been published in MSWiM 2006 [49].

香港大學圖書館 HKU Libraries

Chapter 2 Literature Review

In this chapter, we will review and analyze state-of-the-art link scheduling algorithms whose objectives are to minimize the scheduling lengths. Although the objectives of some link scheduling algorithms are not on scheduling lengths minimization (throughput maximization), their algorithms can also be easily adapted to the minimum length link scheduling problems. So we will also cover some of these algorithms. Now we will first discuss the hardness of the minimum length link scheduling problems.

2.1 The Hardness of the MFSAT and the MLSAT Problems

By regarding the wireless link as an edge in the graph, many researchers have claimed that the MFSAT problem is NP-hard through the reduction from the graph coloring problem. For example, this kind of reductions have been used in [2,14]. All of these reductions assume that the link gains between any pair of links are arbitrary values. However this is not true under the physical interference model used in most of the link scheduling problems. Since the link gains are determined by the distances among different links, thus the triangle inequalities must not be violated. So these direct reductions from graph coloring problems are problematic under the physical interference model. Recently, under the assumption that power control is not allowed, the MFSAT problem has been proven to be NP-hard [54]. In addition, even if we allow arbitrary power assignment, the MFSAT problem is still NP-hard [137]. For the MLSAT problem, until the time we are writing the thesis, there is still no rigorous hardness proof. However, some researchers believe that it

香港大學圖書館 HKU Libraries

remains NP-hard [15]. In this case, most of the researchers seek to solve the minimum (frame) length link scheduling problems with heuristic algorithms. All the existing heuristics can be largely classified as either a top-down or a bottom-up approach (cf. Figure 2-1). In a top-down approach, if the given links are not a link matching, the heuristic algorithm would first try to pick the maximal number of links which do not violate the half-duplex constraint (a matching), and then to find a maximal link independent set which does not violate the SINR constraints by removing one link at a time. This process will continue until all links have been scheduled. In a bottom-up approach, the heuristic would pick each link incrementally to see if the union of the selected links satisfies the half-duplex and SINR constraints; if not, the link is discarded. This process continues until it finds a maximal link independent set, and until all the links have been scheduled. Since the top-down approach is based on removing one link at each step, it can also be called a link removal based scheduling approach; similarly, since the bottom-up approach is based on incrementing one link at each step, it can also be called a link incremental based scheduling approach. We now first discuss the link removal based link scheduling algorithms.

2.2 The Top-Down Approaches

To begin this section, we will go to further details in the SINR model. In particular, we will discuss the link gain matrix and some useful properties on it. Based on the SINR model given in Chapter 1, we define a normalized non-negative link gain matrix $H = (h_{ij})$ such that $h_{ij} = \beta \cdot g_{ij} / g_{ii} = \beta \cdot d^{\alpha}(i_s, i_r) / d^{\alpha}(j_s, i_r)$, for $i \neq j$, and $h_{ij} = 0$, for $i = j$. Now if we construct an associated directed

香港大學圖書館 HKU Libraries

graph of matrix H as follows: for each element $h_{ij} > 0$, we add a directed edge from node i to node j. From this we know this directed graph is strongly connected. Then by using a theorem in [141, p.20] which asserts that a matrix is irreducible if and only if its associated directed graph is strongly connected, we know H is an irreducible matrix. We also define a normalized noise vector $\eta = (\eta_i)$ such that $\eta_i = \beta \cdot n_i / g_{ii} = \beta \cdot n_i \cdot d^{\alpha}(i_s, i_r)$. With these definitions, we can rewrite the SINR inequality as $p_i = \sum_{j=1}^{Q} h_{ij} \cdot p_j + \eta_i$. Now by using the power vector $P = (p_i)$ and the normalized noise vector $\eta = (\eta_i)$, the SINR inequality becomes $P \geq HP + \eta$, or $(I - H)P \geq \eta$. If there is only one transmitting link, i.e., no interferences from other links, the SINR model degenerates into the SNR (Signal to Noise Ratio) model, which is shown below:

$$p_i \geq \beta \cdot n_i \cdot d^{\alpha}(i_s, i_r) \tag{2.1}$$

Obviously, the right hand side of Inequality (2.1) is the minimum power of link i's transmitter i_s to use such that the receiver i_r can successfully decode the packet. We now define the spectral radius $\rho(H)$ of the H matrix as $\rho(H) = \max_i |\lambda_i(H)|$ where $\lambda_i(H)$ stands for the ith eigenvalue of H. Now according to the Perron-Frobenius Theorem [92], since H is a non-negative irreducible matrix, we know that $\rho(H)$ is positive and the corresponding eigenvector has strictly positive components. Let r_i and c_j represent the ith row sum and jth column sum of H, and we have: $r_i = \sum_j h_{ij}$ and $c_j = \sum_i h_{ij}$. The following is a compilation of the useful propositions of the H matrix shown in [42,46,47,92]:

香港大學圖書館 HKU Libraries

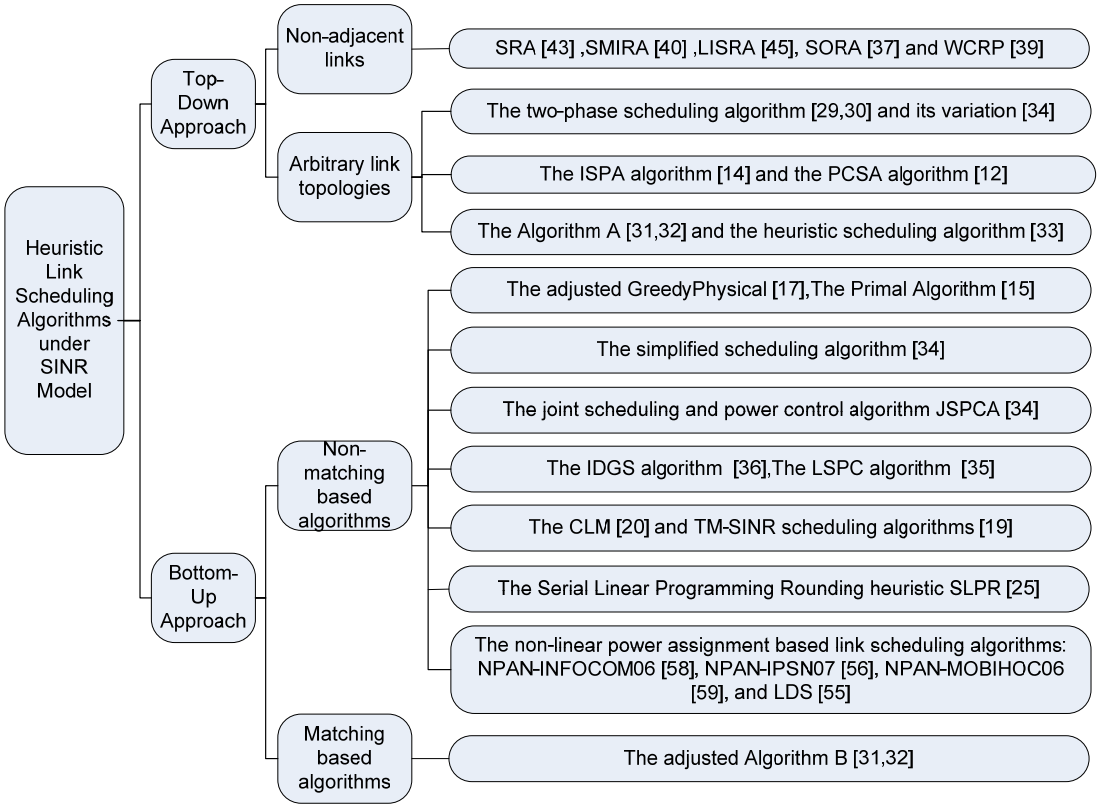


Figure 2-1: Categorization of existing heuristic link scheduling algorithms under the SINR model

Proposition 2.1: $\rho(H)$ increases when any entry of H increases.

Since $h_{ij} = \beta \cdot d^{\alpha}(i_s, i_r) / d^{\alpha}(j_s, i_r)$, we can see that $\rho(H)$ can be reduced by either reducing the threshold value β, the length of any links or by selecting the links which can result in larger $d(j_s, i_r)$ values.

Proposition 2.2: $\rho(H)$ is lower bounded by either the minimum row sum or the minimum column sum, and it is upper bounded by either the maximum row sum or the maximum column sum.

$\min_i(r_i) \le \rho(H) \le \max_i(r_i)$; $\min_j(c_j) \le \rho(H) \le \max_j(c_j)$.

Proposition 2.3: $(I - H)^{-1} > 0$ if and only if $\rho(H) < 1$.

香港大學圖書館 HKU Libraries

Proposition 2.4: The power vector $P^* = (I - H)^{-1} \cdot \eta$ is Pareto-optimal in the sense that $P^* \leq P$ component-wise for any other nonnegative P vector satisfying $(I - H)P \geq \eta$.

After having introduced the link gain matrix and its useful propositions, we will discuss the link removal based scheduling algorithms one by one. According to Figure 2-1, we can further partition the top-down approaches into two branches: the first is to consider non-adjacent links (a link matching), the second is to consider arbitrary link topologies. We first consider the case where all the links form a link matching.

2.2.1 Link removal algorithms for non-adjacent links

The first link removal based scheduling algorithm called SRA (Step-wise Removal Algorithm) is proposed by Zander [43]. For a set of non-adjacent links, this algorithm defers the link which has the maximum value $\max(r_i, c_i)$. The rationale behind this algorithm is based on Proposition 2.2, i.e., the spectral radius of the link gain matrix is bounded by the maximum value of the row sum r_i or the column sum c_i. So the SRA algorithm aims to minimize the upper bound of the spectral radius in each removal step. Note that the CSCS (Combined Sum Criterion Selection) algorithm presented in [36] is actually the same as SRA. Instead of minimizing the upper bound of the spectral radius, the SORA (Step-wise Optimal Removal Algorithm) proposed by Wu [37] defers the link whose removal can minimize the spectral radius directly in each step. However, different from SRA which needs only $O(n)$ eigenvalue computations, the SORA algorithm

香港大學圖書館 HKU Libraries

needs $O(n^2)$ eigenvalue computations where n is the number of links. Aiming at removing the link which can cause the maximum interference, Zander [45] proposed another algorithm called LISRA (Limited Information Stepwise Removal Algorithm). In this algorithm, assuming all the links employ the same transmission powers, the link with the minimum SINR value is excluded in each step. For each link in SMIRA (Step-wise Maximum Interference Removal Algorithm) [40], the algorithm first computes the larger interference value between the received cumulative interferences from other links and the interferences it caused to all the other links, and then it postpones the link which has this largest interference value. For each link in the WCRP algorithm (named with the four initial letters of the four authors' family names) proposed by Wang et al.[39], it first computes a so called MIMSR (Maximum Interference to Minimum Signal Ratio) value, and then all the links whose MIMSR values exceed some pre-determined threshold is removed in each step.

2.2.2 Link removal algorithms for arbitrary topologies

Having covered the link removal algorithms for non-adjacent links, we now turn to the algorithms for the set of arbitrarily constructed links. To our current knowledge, the two-phase link scheduling algorithm in [29,30] is the first solution to the joint link scheduling and power control problem for ad-hoc networks. In the first phase, this algorithm uses a separation distance to find a "valid" link set. This links in the 'valid' link set must first guarantee that all the links are non-adjacent; second the Euclidean distances between any pair of links must be larger than the separation distance (the protocol interference

香港大學圖書館 HKU Libraries

models). From this we can deduce that these links must form a subset of some maximal matching of the original links. Here, the larger the separation distance, the fewer the number of links in the found 'valid' link set. In the second phase, this algorithm tries to find an "admissible" link independent set satisfying the SINR constraints by using the LISRA algorithm in each link removal step. A variation of the two-phase link scheduling algorithm has been presented in [34]. This algorithm first defines a link metric which is a combination of the link's queue length and the number of blocked links (the number of links sharing either a transmitter node or a receiver node of the current link). Then it finds a maximal matching by greedily selecting a link with the longest queue length and the fewest blocked links (the lowest link metric value). There are two differences between the two-phase scheduling and its variation algorithm: the first is that the variation algorithm sets the separation distance value as zero, which means it tries to find a maximal matching but not its subset; the second difference is that, in order to find an admissible link independent set, the variation algorithm defers the link with the largest link metric, i.e., the link with the shortest queue length and the maximum number of blocked links. So if we do not consider the backlogged system, which means we do not consider the links' queue lengths, the link with the maximum number of blocked links rather than the link which has the lowest SINR value is removed.

The PCSA (Power Controlled Scheduling Algorithm) presented in [12] behave similarly as the ISPA (Integrated link Scheduling and Power control Algorithm) proposed in [14]. Both of these two algorithms first construct a (generalized) power-based interference graph. This kind of interference

香港大學圖書館 HKU Libraries

graph is constructed as follows: First, we take all of these links as vertices in the interference graph; second, for each pair of links we check the spectral radius $\rho(H)$ of the link gain matrix consisting of the two links, then according to Proposition 2.3, if $\rho(H) \geq 1$, we add an edge between these two links which correspond to two nodes in the interference graph; third, even if $\rho(H) < 1$ but if any power component in the Pareto-optimal power vector P^* (Proposition 2.4) is larger than the maximum allowable power, we need to add an edge between these two links. From this we can conclude that the links in this graph also form a subset of some maximal matching of the original links. When this interference graph is established, by using the minimum degree greedy algorithm (MDGA), the ISPA algorithm finds a maximal number of links which satisfy the SINR constraints pair-wisely. Finally, they use the SMIRA algorithm as the pruning method to find a maximal number of links that satisfy the SINR constraints. The difference between the ISPA and the PCSA algorithm is that, a "maximality stage" is added after the link removal step in the ISPA algorithm. This step is to find more links to be added to the link independent set.

Different from all the previously mentioned link removal based scheduling algorithms, the Algorithm A in [31,32] first defines each link's effective interference as the corresponding column sum (c_i) in the link gain matrix, and then it finds a maximum matching of the links directly instead of finding a maximal matching or even a subset of the maximal matching. If the maximum matching does not satisfy the SINR constraints, the link with the maximum effective interference is discarded in each link removal step. This process is

香港大學圖書館 HKU Libraries

repeated until all links have been scheduled. Also for the set of non-adjacent links, the heuristic algorithm given in [33] first finds a link matching, not necessarily the maximum matching, second it discards the link with the maximum row sum value r_i in the link gain matrix.

2.3 The Bottom-Up Approaches

As mentioned earlier, the bottom-up approach is based on scheduling each link incrementally. The main difference between the top-down and bottom-up scheduling approaches is that, for a set of non-adjacent links, the top-down approach always consists of two phases, i.e., the link matching searching phase (either a maximum matching, a maximal matching or even just a matching) and the link removal based scheduling phase. The bottom-up approach, however, can directly schedule the links one by one without first finding a link matching. So we can largely classify the bottom-up approach into two categories: matching based scheduling and non-matching based scheduling. We will first study the non-matching based algorithms since most state-of-the-art link incremental based scheduling algorithms directly schedule the links one by one without first finding a link matching.

2.3.1 Non-matching based link incremental scheduling

The first polynomial time approximated link scheduling algorithm called GreedyPhysical is given in [17]. The approximation bound of this algorithm, however, is proved under the assumption that the set of nodes are uniformly distributed at random in a square of unit area or a disk of unit area. This means that the approximation bound can not be generalized to arbitrarily

香港大學圖書館 HKU Libraries

constructed links over arbitrarily located nodes on a plane. Moreover, the algorithm does not use packet-level power control, which means that all the links in the same timeslot employ the same transmission powers. Since this algorithm is designed for links with non-unit traffic demands, i.e., different links may have different number of packets to be transmitted, it can be easily applied to the unit traffic demand case. Moreover, as we will see, since constant power assignment can result in very undesirable scheduling length, we can modify this algorithm by allowing power control at the packet-level, which means the links scheduled in the same timeslot can employ different transmission powers. The original algorithm first sorts all the links in the decreasing order of their interference numbers. Here by the interference number of a link, it means the number of links which do not share a common node with the current link and can not be concurrently scheduled with it under constant power assignment (cf. Inequality (1.4)). But since we allow packet-level power control, we modify the definition of interference degree as the number of links which do not share a common node with the current link and can not be concurrently scheduled with it SINR model (cf. Proposition 2.3). Second, the GreedyPhysical algorithm proceeds as greedily schedules these links from the link with the largest interference number to the link with the fewest interference number.

The Primal Algorithm proposed in [15] is designed originally for some kind of “superincreasing” link demands, which means when we sort the link demands in a non-increasing order, each link with a higher demand is greater than or equal to the sum of all the links with lower demands. This algorithm first finds the link with the largest link demand, and then all the other links

香港大學圖書館 HKU Libraries

which can be pair-wisely scheduled with the current link under the SINR model. After that the algorithm schedules these two link sets with the duration of the link with a lower link demand. And then the algorithm checks how many packets have not been transmitted for the link with the largest link demand and schedules this single link packet by packet. The algorithm repeats these steps until all the packets have been transmitted. The authors of this paper have proven that this polynomial time greedy algorithm is optimal for these 'superincreasing' link demands. We can adapt the algorithm to general non-uniform link demands by first sorting the links in a decreasing order of their link demands, and then picking each link in order using the bottom-up approach. Obviously, this method can not guarantee the optimal scheduling length for general non-uniform link demands cases.

Also designed for non-uniform link demands, the IDGS (Increasing Demand Greedy Scheduling) algorithm presented in [36] first sorts the links in an increasing order of their link demands; and then in each timeslot it first picks the link with the lowest link demand, and then it switches to pick the links in a reversed order, i.e., selecting the link with the highest link demand using a bottom-up approach.

We now review the two non-matching based scheduling algorithms proposed in [34]. The simplified scheduling algorithm first sorts the links in an increasing order of their link metrics, and then picks each link in order while giving it a power level which is the smaller value of its linear power assignment (a power assignment proportional to its link length to the power of the path loss exponent) and its maximum allowable power level. If any SINR constraints are violated then it defers it to the next timeslot. The second joint

香港大學圖書館 HKU Libraries

link scheduling and power control algorithm (JSPCA) behaves similarly with the simplified scheduling algorithm with the difference that the former one assigns the power levels with the values calculated from the Pareto-optimal power vector P^* (Proposition 2.4) rather than the pre-determined power assignments. Compared with the two-phase link removal algorithm and the simplified scheduling algorithm, the authors have shown that the JSPCA algorithm can greatly improve the network performance in terms of throughput and delay. The link scheduling and power control algorithm (LSPC) proposed in [35] first constructs a conflict graph which is based on the node-exclusive interference model (links sharing a common node can not be concurrently scheduled), and then sorts the links either in an increasing order or in a decreasing order of the node degrees. Finally it schedules the links in order using the bottom-up approach. Note that if we employ the increasing order and if we do not consider a backlogged system (without considering the links' queueing lengths), the LSPC algorithm becomes the same as the JSPCA algorithm presented in [34].

For the throughput maximization problem for single hop links, i.e., to compute the maximum number of packets transmitted on these links in a fixed frame length, Tang et al. [25] first formulate it as a mixed integer linear programming (MILP), and then they relax it as a linear programming. In order to generate a link's ordering for the proposed serial linear programming rounding algorithm (SLPR), the authors also relax the SINR requirement. Then by solving the linear programming, they sort the links in a decreasing order of the fractional values of the scheduling variables. Finally the greedy SLPR algorithm incrementally schedules these links using the bottom-up

香港大學圖書館 HKU Libraries

approach. The intuitive idea of this link ordering is that, the larger the fractional value of the scheduling variable calculated from the relaxed SINR model, the higher the probability of this link satisfying the original SINR requirement. Note that although this is a polynomial time algorithm, it suffers from an extremely high worst case computational complexity $O(n^8 \cdot M_{LP})$, where n is the number of the links and M_{LP} is the number of binary bits required to store the data.

We now turn to reviewing another class of non-matching based scheduling algorithms which feature a kind of nonlinear power assignment. Informally, nonlinear power assignment is a kind of strategy between constant and linear power assignments. This power assignment can overpower the short links, which means that on one hand, compared with constant power assignment, long links can use larger powers; on the other hand, short links can receive relatively larger power compared with linear power assignment. The nonlinear power assignment is first introduced in an algorithm (we call it NPAN-INFOCOM06) for the MFSTT problem [58] and has subsequently been used for the MFSAT problem (Here NPAN stands for Nonlinear Power Assignment for Narrowband Networks). The NPAN-INFOCOM06 algorithm can schedule all the links in a tree topology constructed by the nearest neighbor forest algorithm with $O(\log^4 n)$ timeslots where n is the number of the links. Aiming for the MFSAT problem, also by using the nonlinear power assignment, the authors present an algorithm (we call it NPAN-MOBIHOC06) [59] that studies the relationship between the graph-based interference model (called the in-interference degree) and the

香港大學圖書館 HKU Libraries

SINR model. Here by the in-interference degree of a node, we mean the number of other transmitters whose transmission ranges cover this node. And the largest in-interference degree of a node is called the in-interference degree of the topology. This paper concludes that the scheduling length of the MFSAT problem is upper bounded by the in-interference degree of the topology times the square of the logarithmic function of the number of the links. From this, we can see that a lower in-interference degree greatly shortens the scheduling length. In a subsequent paper [55], the authors propose a low disturbance scheduling algorithm called LDS. This algorithm can generate a poly-logarithmic scheduling length for a topology with low disturbances. Here low disturbance is characterized by a parameter called ρ-*disturbance* which can also be regarded as the density of the links' distribution. For a link's ρ-*disturbance* , the algorithm first computes the number of other links' transmitters (receivers) located in the current link transmitter's (receiver's) range (the link's length divided by the value ρ which is greater than or equal to 1), and then the larger value is the link's ρ-*disturbance* . The maximum ρ-*disturbance* of all the links becomes the ρ-*disturbance* of the topology. With this parameter, the authors prove that the scheduling length of the MFSAT problem is upper bounded by the ρ-*disturbance* of the topology times the product of the square of the logarithmic function of the number of the links and the square of the ρ value. From this, we know that a sparse link topology with a lower ρ-*disturbance* can significantly reduce the scheduling length. Similar to the NPAN-INFOCOM06 algorithm [58], the algorithm proposed in [56] (we call it NPAN-IPSN07) is

香港大學圖書館 HKU Libraries

also aimed for the MFSTT problem. This algorithm also employs the nonlinear power assignment, but it can reduce the scheduling length for all the links in a tree topology constructed by the nearest component connector algorithm to $O(\log^2 n)$ where n is the number of the links.

The cross-layer latency minimization problem (CLM) and throughput maximization problem (TM-SINR) for multi-hop flows have been studied in [20,19]. Here a multi-hop flow consists of several links where each packet is passed from the first link in the flow to the last link in the flow. The algorithms proposed in these two papers also belong to the Bottom-Up approach category since they all schedule each link one by one. These algorithms take the routing issues into account, but their scheduling parts behave similarly with the nonlinear power assignment based link incremental scheduling algorithms in [55,56,58,59] except that they use either constant power assignment or linear power assignment.

2.3.2 Matching based link incremental scheduling

In this section, we will introduce a link incremental scheduling algorithm which is based on first finding a link matching. This algorithm is called Algorithm B [31,32], and it is the only matching based link incremental scheduling algorithm we've found in the literature. The Algorithm B, however, is originally designed for minimizing the total power consumption, but it can be adapted for the minimum length link scheduling problem with a few modifications. Similar to Algorithm A given in the same paper which uses a top-down approach, the Algorithm B first finds a maximum matching of the unscheduled links; second, it sorts all the links in the maximum matching in a

香港大學圖書館 HKU Libraries

decreasing order of their effective interferences; third, the algorithm can then be adjusted to pick each link in order using the bottom-up approach. The authors have shown that Algorithm B can schedule more links in a timeslot than the top-down approach based Algorithm A.

2.4 Time Complexities of the Heuristic Link Scheduling Algorithms

In this section, we will briefly summarize the time complexities of the various scheduling algorithms just reviewed. First based on Proposition 2.3, since most of the heuristic link scheduling algorithms reduce the problem of finding whether there are positive power assignments that satisfy the SINR constraints to the spectral radius checking problem, the time complexities of these algorithms are dominated by the matrix eigenvalue computation. The time complexity for the $n \times n$ matrix eigenvalue computation and matrix inversion problem is $O(n^3)$ [95]. Based on this result, we then briefly review the worst case time complexities for all the link removal and link incremental based algorithms which need matrix eigenvalue computations.

We first review the time complexities for the link removal based algorithms. In the worst case, any link removal based algorithms can only schedule one link in each timeslot. This means that these algorithms will do $O(n)$ eigenvalue computations in each timeslot (Here n means the number of links). Since there are n timeslots, the total numbers of engenvalue computations is $O(n^2)$. Now since each eigenvalue computation takes time $O(n^3)$, we know the overall time complexity is $O(n^5)$. Most of the current

香港大學圖書館 HKU Libraries

link removal based algorithms, including SRA, SMIRA, LISRA, WCRP, the PCSA algorithm and the ISPA algorithm belong to this group (cf. Section 2.2). However, as mentioned in Section 2.2.1, the SORA algorithm is an exception which needs more eigenvalue computations in each timeslot. Specifically, each link removal in the SORA algorithm needs $O(n^2)$ eigenvalue computations. Thus in the worst case, the SORA algorithm needs $O(n^3)$ total eigenvalue computations. As a result, the overall time complexity for any link removal based algorithm which uses SORA as a link removal algorithm takes $O(n^6)$ time.

Second, we review the time complexity of the link incremental based scheduling algorithms. Similarly, in the worst case, any link incremental based algorithms can only schedule one link in each timeslot. This means that these algorithms will do $O(n)$ eigenvalue computations in each timeslot. Since there are n timeslots, the total numbers of engenvalue computations is $O(n^2)$. Now since each eigenvalue computation takes time $O(n^3)$, we know the overall time complexity is $O(n^5)$. The link incremental based scheduling algorithms, including the GreedyPhysical, JSPCA, LSPC, IDGS and the simplified scheduling algorithms belong to this category (cf. Section 2.3).

Now we review the time complexities of the scheduling algorithms which do not need costly eigenvalue computations. All the nonlinear power assignment based link incremental scheduling algorithms belong to this category. Similarly, in the worst, these algorithms can only schedule one link in each timeslot. Thus the overall time complexities of all the nonlinear power

香港大學圖書館 HKU Libraries

assignment based algorithms, including NPAN-INFOCOM06, NPAN-MOBIHOC06, NPAN-IPSN07 and LDS, are $O(n^2)$.

2.5 Algorithms Inefficiency Analyses

In this section, we will give some inefficiency results for both top-down and bottom-up based link scheduling algorithms. These results generalize the wireless link scheduling algorithms inefficiency results in [55]. Before giving more details, we need to give a theorem for any link scheduling algorithm which employs either constant or linear power assignment. The proof of this theorem was first given in our published paper [49].

2.5.1 Inefficiency of constant and linear power assignments

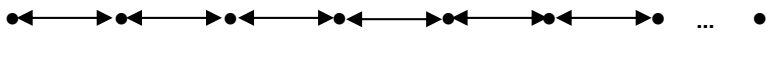
We first give an exponential node chain which is shown in Figure 2-2. In this chain, there are n nodes (X) starting from the leftmost node x_1 and end at the rightmost node x_n. All the nodes are placed on a straight line with exponentially increasing distances between them. For every node $x_i \in X$, we require it can successfully send at least one packet to its nearest neighbor. Now we want to prove a theorem for the scheduling length to schedule all of these n links for any scheduling algorithms which employ either constant or linear power assignments. We have given the mathematical formulations of the constant and linear power assignments in Inequality 1.4 and Inequality 1.5, and here we first formally define the constant and linear power assignments.

香港大學圖書館 HKU Libraries

Constant (Uniform) Power Assignment: If all the concurrently scheduled links employ the same transmission power, we call it a constant (uniform) power assignment.

Linear Power Assignment: If each link in the concurrently scheduled links employs the transmission power which is proportional to the corresponding link's length (the distance from the transmitter to the receiver) to the power of the path loss exponent, we call it a linear power assignment.

THEOREM 2.1: Under the SINR model given in Inequality 1.1, for both constant and linear power assignments, no matter what link scheduling algorithm we use, the scheduling length for all the links in the exponential node chain is at least $n \cdot \beta / (m \cdot 2^{\alpha} + \beta) \in \Omega(n / m)$, even in the absence of ambient noise, where n is the number of the nodes, and m is the processing gain.



2^0 2^1 2^2 2^3 2^4 2^5 2^n

Figure 2-2: Exponential node chain, where 2^i is the distance between nodes x_{i-1} and x_i

PROOF: a) Constant power assignment

In this case, for all nodes, transmission power $P_i = P_k$ =P. Now consider the example in Figure 2-2; we assume there are at most L simultaneous transmissions in a scheduling timeslot. Suppose node x_s is the right-most transmitter in this timeslot, and node x_r is its receiver. The other (L-1) simultaneous transmissions will cause aggregate interferences to node x_r.

香港大學圖書館 HKU Libraries

According to the property of the exponential node chain, if node x_r is on the left side of node x_s, the distance from every other simultaneous transmitter to the receiver x_r is $d(x_i, x_r) \le d(x_s, x_r)$; and if node x_r is on the right side of node x_s, the distance from every other simultaneous transmitter to the receiver x_r is $d(x_i, x_r) \le 2 \cdot d(x_s, x_r)$. Therefore the aggregate interferences caused by these (L-1) simultaneous transmitters are at least $(L-1) \cdot P / (2 \cdot d(x_s, x_r))^{\alpha}$. According to the SINR inequality 1.1 and by plugging in the link gain values, we have:

$$\frac{P / d(x_s, x_r)^{\alpha}}{N + (L-1) \cdot P / (2 \cdot d(x_s, x_r))^{\alpha}} \ge \frac{\beta}{m}$$

From the above inequality, it follows that the maximum number of simultaneous transmissions L in each timeslot is $(\beta + m \cdot 2^{\alpha}) / \beta$. Therefore, the constant power assignment method requires at least $n \cdot \beta / (\beta + m \cdot 2^{\alpha})$ timeslots to schedule all nodes at least once.

b) Linear power assignment

With linear power assignment, the sender x_s will send to its receiver x_r with power $P_s = \rho \cdot d(x_s, x_r)^{\alpha}$, where ρ denotes the minimum received power to decode the message. Similar to the constant power assignment analysis, we assume there are at most L simultaneous transmissions in a scheduling timeslot. According to the property of the exponential node chain, for all nodes x_i, it will cause at least the interference $\rho / 2^{\alpha}$ to its left side nodes [58]. Now suppose x_r is the left-most receiver,

香港大學圖書館 HKU Libraries

and x_s is some transmitter in the L simultaneous transmissions. The other (L-1) simultaneous transmissions will cause at least the aggregate interferences $(L\text{-}1)\cdot\rho/2^{\alpha}$ to this left-most receiver x_r. According to the SINR inequality 1.1, we have

$$\frac{\rho \cdot d(x_s, x_r)^{\alpha} / d(x_s, x_r)^{\alpha}}{N + (L-1)\cdot\rho/2^{\alpha}} \geq \frac{\beta}{m}$$

From the above inequality, it follows that the maximum number of simultaneous transmissions L in each timeslot is $(\beta + m\cdot 2^{\alpha})/\beta$. And therefore the linear power assignment method requires also at least $n\cdot\beta/(\beta + m\cdot 2^{\alpha})$ timeslots to schedule all nodes at least once.

By combining the two results for constant and linear power assignments, we finish the proof for Theorem 2.1. Now if we consider narrowband networks, i.e., the processing gain m=1, we have the following corollary. Note that this result has been proved separately in [58].

COROLLARY 2.2: For both constant and linear power assignments in narrowband networks, no matter what link scheduling algorithm we use, the scheduling length for all the links in the exponential node chain is $\Omega(n)$, where n is the number of the nodes.

Now in order to show the inefficiency for both constant and linear power assignment, we give another theorem which states that the links constructed over the exponential node chain can be scheduled in $O(\log^2 n)$.

THEOREM 2.3: In narrowband networks, all the links in the exponential node chain can be scheduled in $O(\log^2 n)$ with a nonlinear power assignment algorithm proposed in [55].

香港大學圖書館 HKU Libraries

PROOF: The algorithm in [55] proves that the scheduling length for any arbitrary link topologies is $O((\rho - disturbance) \cdot \rho^2 \cdot \log n \cdot (\log n + \rho))$. Since each node sends a packet to its nearest neighbor in the exponential node chain topology, if we set ρ=1, then $\rho - disturbance$ is a constant value (cf. p27 for the definition of $\rho - disturbance$). Thus we complete the proof.

By comparing Theorem 2.1 and Theorem 2.3, we can see that any link scheduling algorithm which employs either constant or linear power assignments are inefficient since it leads to exponentially longer scheduling length than the scheduling algorithm based on a nonlinear power assignment.

2.5.2 Inefficiency of top-down based scheduling algorithms

THEOREM 2.4: The following top-down based link scheduling algorithms, i.e., the two phase scheduling algorithm [29,30], the variation of the two phase scheduling algorithm [34], the PCSA and the ISPA algorithms [12,14], the Algorithm A [31,32] and the heuristic link scheduling in [33] have a worst case scheduling length lower bound of $\Omega(n)$.

PROOF: Since the two phase scheduling algorithm, the PCSA and the ISPA algorithm use either LISRA or SMIRA as their link removal algorithms, the inefficiency results of the four link removal algorithms (SRA, SMIRA, LISRA and WCRP) (Theorem 5.2 in [55]) can be directly applied here. For the other three scheduling algorithms, we can also take the same co-centric exponential node chain given in that paper as a worst case link topology. In this topology, all the links' transmitters and receivers are located on the same line with link i 's transmitter coordinate as $(-2^{(i-1)},0)$ and link i 's receiver

香港大學圖書館 HKU Libraries

coordinate as $(2^{(i-1)},0)$ (i is from 1 to n). In this case, the associated link gain matrix of this link topology is $h_{ij} = \beta \cdot (2^i/(2^{i-1}+2^{j-1}))^\alpha$ if $i \neq j$ and $h_{ij} = 0$ if $i = j$ (cf. Section 2.2, p15). Similarly we can also set the path loss exponent $\alpha = 3$, the background noise $n_i = 0$ and the threshold $\beta = 2$. For the variation of the two phase scheduling algorithm, since all the links have the same number of blocked links (zero), the links removed in each step is link 1 to link n-1, so only one link (link n) can be scheduled in the first timeslot. These removal steps will be repeated in the following n-1 timeslots. For the Algorithm A and the heuristic link scheduling, since they either use the link gain matrix column sum or row sum as their link removal metrics, the links removed in each step are either in an increasing order of their links' lengths or in a decreasing order of their links' lengths. However, both orders will result in $\Omega(n)$ scheduling lengths. This completes the proof.

Now since the co-centric exponential node chain topology can be scheduled in $O(\log n)$ timeslots by a nonlinear power assignment based link scheduling algorithm in [55], we can see that the top-down based link scheduling algorithms shown in Theorem 2.4 are inefficient.

2.5.3 Inefficiency of bottom-up based scheduling algorithms

THEOREM 2.5: The two bottom-up based link scheduling algorithms, i.e., the simplified scheduling algorithm in [34] and the GreedyPhysical algorithm in [17] have a worst case scheduling length lower bound $\Omega(n)$.

PROOF: We can also take the co-centric exponential node chain as an example (cf. the proof in Theorem 2.4). Since all the links form a matching,

香港大學圖書館 HKU Libraries

the algorithm can schedule the links in a decreasing order of their lengths. So depending on the value of maximum allowable transmission power, the corresponding power assignments can be either linear power assignments, constant power assignments, or the long links employing constant power assignments while the remaining short links employing linear power assignments. According to Theorem 2.1, we can complete the proof for the simplified scheduling algorithm. Similarly since the GreedyPhysical algorithm does not employ packet-level power control, which means that all the links in the same timeslot use the same transmission powers (the links in different timeslots may use different powers), Theorem 2.1 can be directly applied here. This completes the proof for the GreedyPhysical algorithm.

Now before we introduce the inefficiency results for some other link incremental scheduling algorithms, we need to introduce a pair-wise link conflict (infeasible) graph. This graph is based on the following theorem and is similar to the (generalized) power based interference graph introduced in the PCSA scheduling algorithm and in the ISPA scheduling algorithm (cf. Section 2.2.2).

THEOREM 2.6: In narrowband networks, for any two transmissions (x_s, x_r) and (y_s, y_r), if $d(x_s, y_r) \cdot d(y_s, x_r) \leq \beta^{2/\alpha} \cdot d(x_s, x_r) \cdot d(y_s, y_r)$, then there exists no feasible power assignment for simultaneous transmissions (infeasible link independent set); otherwise, there always exists a feasible power assignment to have a simultaneous schedule.

PROOF: If the two transmissions can be successfully scheduled, based on SINR model (inequality 1.1) with processing gain equal to 1, the following two inequalities must follow:

香港大學圖書館 HKU Libraries

$$\frac{P_x / d(x_s, x_r)^\alpha}{N + P_y / d(y_s, x_r)^\alpha} \geq \beta \qquad \frac{P_y / d(y_s, y_r)^\alpha}{N + P_x / d(x_s, y_r)^\alpha} \geq \beta$$

From these inequalities, we have

$$\beta \cdot P_y \frac{d(x_s, x_r)^\alpha}{d(y_s, x_r)^\alpha} < P_x < \frac{1}{\beta} \cdot P_y \cdot \frac{d(x_s, y_r)^\alpha}{d(y_s, y_r)^\alpha}$$

Therefore, if $\beta \cdot \frac{d(x_s, x_r)^\alpha}{d(y_s, x_r)^\alpha} \geq \frac{1}{\beta} \cdot \frac{d(x_s, y_r)^\alpha}{d(y_s, y_r)^\alpha}$, there is no feasible power assignment for simultaneous scheduling; otherwise, there always exists a feasible power assignment to schedule these two transmissions in parallel.

According to this theorem, we construct the pair-wise link conflict (infeasible) graph as follows: We first take each link as a node in the graph, second we add an edge between any two links which satisfy the inequality given in Theorem 2.6.

PROPOSITION 2.7: Let's suppose there is a link topology whose pair-wise link conflict (infeasible) graph is as shown in Figure 2-3, then any link incremental scheduling algorithms which schedule the links in the order of [1..n] will result in a scheduling length of $\Omega(n)$ (Similar to the worst case behavior of some graph coloring algorithms analyzed in [88]). However, a much fewer or even a constant number of timeslots is possible if we schedule the links in the upper and lower parts of this conflict graph respectively.

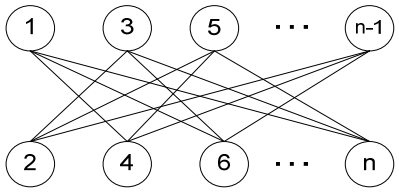


Figure 2-3: A Pair-wise Link Conflict (Infeasible) Graph

香港大學圖書館 HKU Libraries

From this proposition, we have the following three corollaries.

COROLLARY 2.8: The link incremental scheduling algorithms which use the node degree in the pair-wise link conflict graph as the scheduling metric (the criterion for scheduling the next link, i.e., the ordering of links), such as the adjusted GreedyPhysical algorithm (GreedyPhysical algorithm with packet-level power control), has a worst case lower bound of $\Omega(n)$.

COROLLARY 2.9: Since all the links have the same link demands in MFSAT, the link incremental scheduling algorithms which use the link demands as a scheduling metric, such as the Primal Algorithm in [15] and the IDGS algorithm in [36], have a worst case lower bound of $\Omega(n)$.

COROLLARY 2.10: Let's further suppose all the links in this link topology are non-adjacent or have the same number of blocked links, then the link incremental scheduling algorithms which use the number of blocked links as the scheduling metric, such as the JSPCA algorithm in [34] and the LSPC Algorithm in [35], have a worst case lower bound of $\Omega(n)$.

香港大學圖書館 HKU Libraries

Chapter 3 MDCS-Maximum Directed Cut based Scheduling Framework for the MFSAT Problem

In this chapter, we will present a maximum directed cut based scheduling framework (MDCS) for the MFSAT problem. This framework is also a two phase scheduling algorithm, and there is a fundamental difference between MDCS and the heuristic link scheduling algorithms reviewed in Chapter 2. This difference is, in the first phase, we choose a maximum directed cut of the links after finding a maximum matching. In addition, for all the links in each directed cut, we choose to use a link incremental scheduling algorithm instead of a link removal scheduling algorithm. Now before delving into the details of the MDCS framework, we first discuss the insufficiency of using a maximal matching in the first phase of two-phase scheduling algorithms for arbitrary link topologies.

3.1 Insufficiency of Using Maximal Link Matching

We have introduced many state-of-the-art two-phase link scheduling algorithms for arbitrary link topologies in Chapter 2, but most of them either employ a link removal algorithm in the second phase for finding a link independent set or choose to find a maximum (maximal) matching or even a subset of the maximal matching in the first phase. There is only one two-phase link scheduling algorithm which first finds a maximum matching in the first phase and then employs a link incremental algorithm in the second phase (cf. Section 2.3.2). Compared with finding a maximal matching or even a subset of the maximal matching in the first phase, since a maximum

香港大學圖書館 HKU Libraries

matching can offer more potential links to be covered in the same timeslot in the second phase, finding a maximum matching in the first phase (This only takes time $O(n^{1.5})$ since there are only n links [96]) is obviously much better. But is this sufficient for our link incremental scheduling algorithm? Now we give an example to answer this question. The link topology is shown in Figure 3-1.

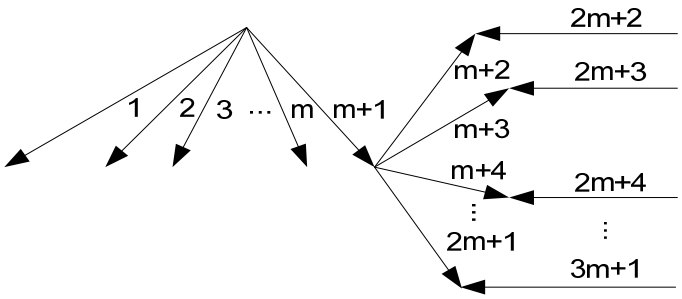


Figure 3-1: An arbitrary link topology with $3m+1$ number of links

Now suppose the first maximum matching of this arbitrary link topology comprises link 1 and links from link $2m+2$ to link $3m+1$. But this maximum matching becomes somewhat inefficient if any link in link 2 to $m+1$ except link 1 can be simultaneously scheduled with any links in $2m+2$ to $3m+1$. The reason is that: if any links in link 2 to $m+1$ can be simultaneously scheduled with the links in the maximum matching, less links will be left in the next phase. Thus the scheduling length could be potentially shortened. In order to solve this problem and to provide more potential links to be scheduled in the same timeslot, we choose to find a maximum directed cut containing this maximum matching. Here by a maximum directed cut, we mean to partition all the nodes into two disjoint node sets so that we can maximize the number of directed edges from one node set to another node set. In this case, we can

香港大學圖書館 HKU Libraries

avoid adding a link such that one link's transmitter (receiver) becomes another link's receiver (transmitter), since this will lead to an infinity value in the link gain matrix H which makes eigenvalue computation impossible and also inefficient for comparisons for scheduling metrics based on H. So now the key here is how to construct this maximum directed cut. Here we note that, a simple maximum directed cut does not work for our scheduling problem, because this maximum directed cut may miss the maximum matching and may comprise too many links that are adjacent to some transmitters or receivers which is very undesirable.

3.2 Maximum Directed Cut with Maximum Link Matching

In order to show the importance of the maximum directed cut construction with maximum matching problem in our scheduling framework, we also take the link topology in Figure 3-1 as an example. Also suppose we first find a maximum matching consisting of link 1 and links from link $2m$+2 to link $3m$+1. Now the first maximum directed cut we find is to add links from link 2 to link m+1, and the second maximum directed cut we find is to add links from link 2 to link m and the links from link m+2 to link $2m$+1. For the first maximum directed cut construction, the total number of timeslots we need to schedule all the links is at least $2m$+1 since the maximum directed cut found in the next phase comprises the links from link m+2 to link $2m$+1 which needs at least m timeslots to schedule, but for the second maximum directed cut construction, a total number timeslots m+1 to schedule all the links is possible. So now we give our algorithm to find this better maximum directed cut.

香港大學圖書館 HKU Libraries

Before we elucidate the algorithm, we need to clarify some notions. First, by an unmatched link, we mean this link is not included in the maximum matching; second, by an unmatched node, we mean this node is not incident to any links in the maximum matching.

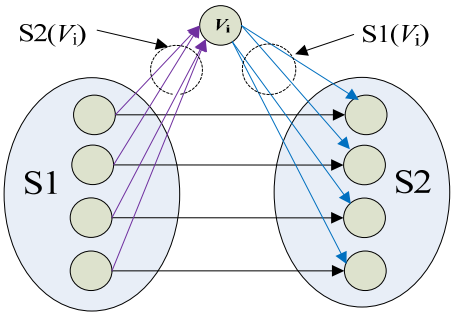


Figure 3-2: An illustrating example for adding an unmatched node in the directed cut

We now give the maximum directed cut construction with maximum matching algorithm in the following. Here the key to this heuristic algorithm is step 4, which is to add an unmatched node into set S1 or S2. We now give an illustrating example of this step in Figure 3-2. Here S1(v_i) denotes the number of directed edges (links) from v_i to the nodes in S2 and S2(v_i) denote the number of directed edges (links) from the nodes in S1 to v_i . (Note that a similar method for maximum cut can be found in [90], but their algorithm can not be applied in our scenario). We now give a theorem to show the worst case performance guarantee of our heurist algorithm for finding this maximum directed cut.

Maximum Directed Cut Construction with Maximum Matching Algorithm:

1: Find a maximum matching;

2: Label the set of transmitters for all the links in the maximum matching as

香港大學圖書館 HKU Libraries

the set S1, and label the set of receivers for all the links in the maximum matching as the set S2; And we called all the links in the maximum matching as a directed cut from S1 to S2.

3: For all the unmatched links which don't have an unmatched node, check whether they can be added to the directed cut; This means that the addition of this new link would not make any node in set S1 (S2) as a receiver (a transmitter);

4: For each unmatched node in the link topology, if $S2(v_i) \geq S1(v_i)$, we put this node in S2, otherwise we put it in S1; Note that in this step, the set S1 and S2 are dynamically updated;

5: Return all the directed links whose transmitters are located in S1 and the corresponding receivers are located in S2.

THEOREM 3.1: For the proposed maximum directed cut with maximum matching problem, the proposed heuristic algorithm can add at least a half of the optimal number of links that can be added to the already existed directed cut (the directed cut derived from step 2 to step 3).

PROOF: Since we first find a maximum matching, we know there are no edges (links) between the unmatched nodes since otherwise it is not a maximum matching. Suppose there are m unmatched nodes $(v_1, v_2, \ldots, v_m)$, then we know that, by using step 4, the number of directed links that can be added to the already existed directed cut is $\sum_{i=1}^{m} \max(S1(v_i), S2(v_i))$. Now we know the optimal number of new links that can be added to the already existed directed cut is equal to or less than $\sum_{i=1}^{m} (S1(v_i) + S2(v_i))$. From this we can prove this theorem.

香港大學圖書館 HKU Libraries

With the help of the maximum directed cut searching algorithm, we now present a maximum directed cut based scheduling (MDCS) framework for arbitrary link topologies.

3.3 Maximum Directed Cut based Scheduling Framework

3.3.1 Pair-wise link conflict graph

In this section, we will briefly review the pair-wise link conflict (infeasible) graph introduced in Section 2.5.3. We call this graph G_{pair}, and this graph will be used by an exact scheduling algorithm in the next Chapter 4. From Theorem 2.6, we can build G_{pair} by just adding an edge between any two infeasible links. This is done in time $O(n^2)$ where n means the number of links. And for each link i, let $N(i)$ denote the number of links which conflict with i, i.e., the number of neighbors of node i in G_{pair}.

3.3.2 The MDCS scheduling framework

MDCS Scheduling Framework:

Input: A set of arbitrarily distributed wireless links $N = \{1,...,n\}$.

Output: The number of used timeslots T to successfully schedule all the links under the SINR model.

1: T=0;

2: While not all links have been scheduled **do**

3: Call the Maximum Directed Cut Construction with Maximum Matching Algorithm in Section 3.2.

4: t=0 and set all the links in the outputted direct cut unchecked and unscheduled;

5: While not all the links in the outputted direct cut have been scheduled **do**

香港大學圖書館 HKU Libraries

6: Construct a pair-wise link conflict graph of all the unscheduled links, and then sort these unscheduled links in a decreasing order of the number of their neighbors in the pair-wise link conflict graph;
7: Schedule the first link and update the remaining links' ordering;
8: Check the next link to see if these links satisfy the SINR constraints;
9: If not satisfied, go back to step 8; otherwise schedule this link and update the ordering of the unchecked links;
10: Repeat step 8 to step 9 until all links have been checked;
11: $t=t+1$ and set all the unscheduled links unchecked;
12: End While;
13: $T=T+t$;
14: End While.

Now we analyze the time complexity of the MDCS scheduling framework. First, we know the time of the maximum directed cut construction with maximum matching algorithm relies on the maximum matching finding time. And we know finding a maximum matching only takes time $O(n^{1.5})$ since there are only n links [96]. Then we analyze the time complexity of scheduling all the links in each outputted directed cut. As we've mentioned in Section 2.4, we know the worst case time complexity of a link incremental scheduling algorithm is $O(n^5)$, then the total time of the MDCS scheduling framework is $O(n^5)$ where n denotes the number of links.

香港大學圖書館 HKU Libraries

3.4 Comparisons of MDCS and other Six Heuristic Link Scheduling Algorithms

3.4.1 Simulation settings

We first give the simulation settings. For any n arbitrarily located nodes on a $2000m \times 2000m$ plane (Here m means meters), we randomly select a link's transmitter and receiver subject to the constraint that they are different nodes on the plane. We then repeat this process until a number of n different links (either with different transmitters or receivers) have been constructed. So in this topology construction, some nodes may not be used (Figure 3-3 gives an example of an arbitrary link topology constructed over 20 arbitrarily located nodes on a plane). In this simulation, since (1) all the arbitrarily constructed link topologies are dense link topologies; and (2) many links share a common node, if we set a very small path loss exponent value or a very high threshold value, no matter what kind of scheduling algorithms we will use, we can only schedule almost one link in each timeslot. The reasons are as follows: (1) If the path loss exponent is very low, say only around 2, all the wireless signals do not rapidly attenuate. Thus all the links generate very large cumulative interferences which could lead to a very small SINR value at each link's receiver. If the SINR threshold is still very high, many links can not be simultaneously scheduled. (2) If many links share a common node, due to the half-duplex constraint, these links can only be scheduled in one by one. From these observations, we set the path loss exponent $\alpha = 5$. But we will test on different SINR threshold values, including $\beta = 1$, $\beta = 2$ and $\beta = 3$.

香港大學圖書館 HKU Libraries

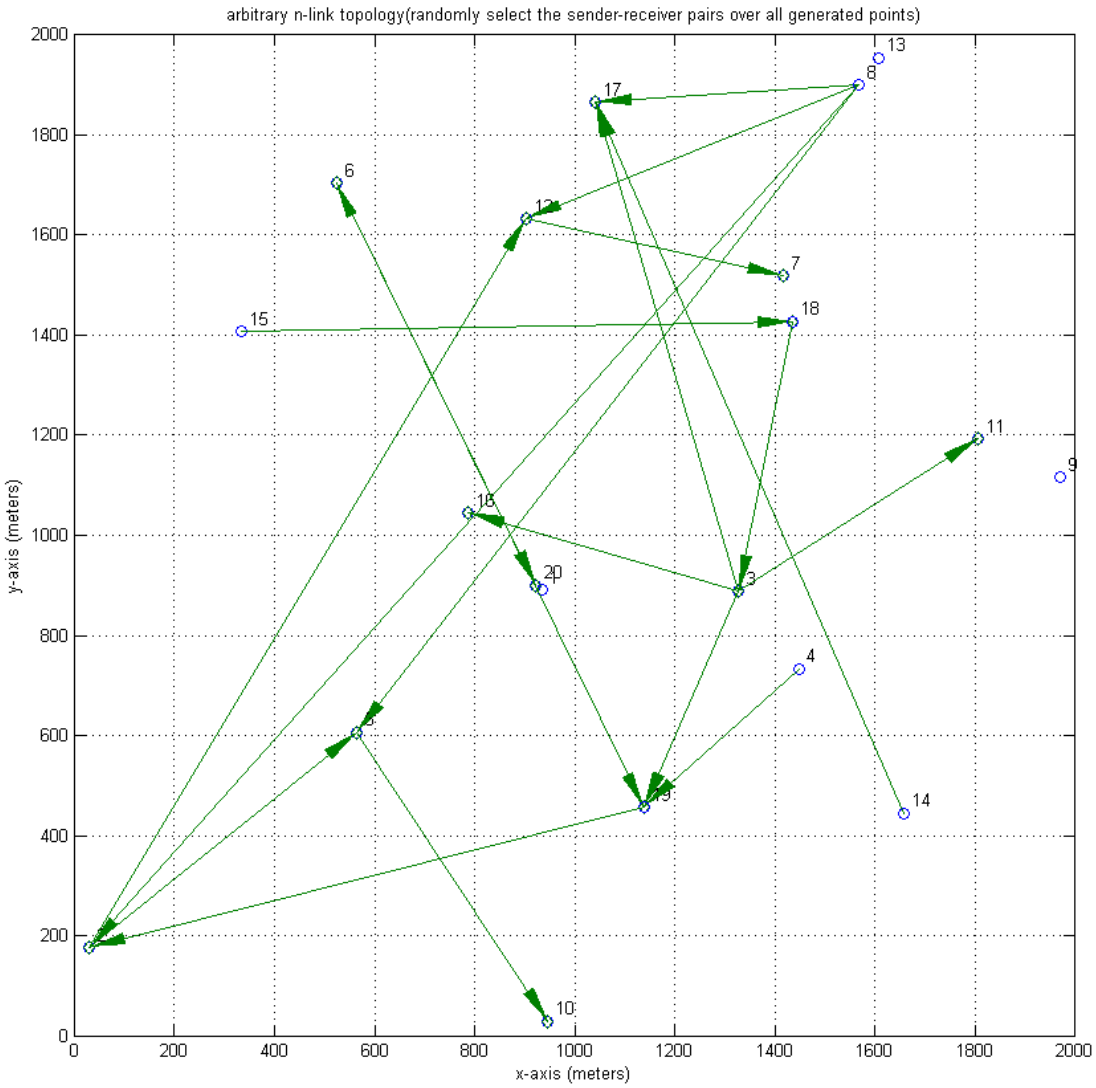


Figure 3-3: An arbitrary link topology constructed over 20 arbitrarily located nodes on a plane

3.4.2 Performance comparisons

We implemented six bottom-up based scheduling algorithms (please refer to Chapter 2 for some of the algorithms descriptions): the proposed MDCS scheduling framework, the bottom up Algorithm B [31,32], the GreedyPhysical algorithm in [17] with packet level power control, the JSPCA algorithm in [34], the LDS algorithm in [55] and the first fit based link increment scheduling algorithm. Here by first fit based link incremental

香港大學圖書館 HKU Libraries

scheduling algorithm, we mean that we just greedily schedule the links in its unsorted order with the bottom up approach. There are two nonlinear power assignment based link incremental scheduling algorithms for arbitrary link topologies, one is NPAN-MOBIHOC06 [59] and the other is LDS [55]. We have tested that the LDS algorithm can generate smaller scheduling lengths than the NPAN-MOBIHOC06 algorithm, so we use LDS as a representative for nonlinear power assignment based link scheduling algorithm. Note that for the LDS algorithm, since its scheduling length relies on the parameter ρ, we have tested different ρ values and find that LDS can achieve the shortest scheduling length when $\rho = 1$, so we set $\rho = 1$ in our simulation. Besides the link incremental based scheduling algorithms, we also implement SORA as a representative for link removal based scheduling algorithms. But since now we are targeted for arbitrary link topologies, we first find a maximum matching in each scheduling phase; then we employ SORA as the link removal algorithm. In addition, for all the scheduling algorithms except LDS, we use the Pareto-optimal power assignment (cf. Proposition 2.4) with no maximum allowable power limitations (In this case we don't care the background noise powers). This assumption, however, can be removed if we set the same maximum allowable power for all the scheduling algorithms. Note that, we have tested these scheduling algorithms over ten sets of link topologies with the number of links ranging from 20 to 110. And for each set of topology, we compute the average scheduling length over 10 different instances.

The final scheduling results can be seen in Figure 3-4, Figure 3-5 and Figure 3-6. According to the three figures, we can draw the first conclusion: the smaller of the SINR threshold value, the fewer timeslots we need to

香港大學圖書館 HKU Libraries

schedule all the links. More conclusions from the scheduling lengths in the three figures are as follows. First, we can sort these eight scheduling algorithms in an increasing order of their scheduling lengths: MDCS, the bottom up Algorithm B, SORA, first fit, JSPCA, the GreedyPhysical with power control and LDS. We have the following observations from this ordering. (1) In matching based link scheduling algorithms, adding more links to the maximum matching in each scheduling phase can significantly reduce the scheduling length. This can be seen from the scheduling lengths of MDCS, the bottom up Algorithm B and the matching based link removal algorithm SORA. (2) Matching based link scheduling algorithms greatly outperform the non-matching based link scheduling algorithms in terms of their scheduling lengths. This can be seen from the scheduling lengths of bottom up Algorithm B and the other three non-matching based scheduling algorithms (first fit, JSPCA and GreedyPhysical). This observation is further strengthened through the result that even the matching based link removal algorithm SORA can generate fewer scheduling lengths than the non-matching based link incremental scheduling algorithms (GreedyPhysical and LDS). (3) Compared with the top down and bottom up based scheduling algorithms, although the SORA algorithm can generate relatively shorter scheduling lengths with small SINR threshold values ($\beta = 1$ and $\beta = 2$) than some link incremental based scheduling algorithms, such as JSPCA and GreedyPhysical, it's obtained by paying more time (cf. Section 2.4). (4) Since our generated arbitrary link topologies bear large ρ-*disturbance* values (Figure 3-3 is an example whose ρ-*disturbance* value could be as large as

香港大學圖書館 HKU Libraries

the number of links when $\rho = 1$), the low disturbance scheduling (LDS) generates the longest scheduling lengths at every topology instance (it almost schedules one link in each timeslot!). In Chapter 7, we will see how LDS performs in the link topologies with much smaller ρ - *disturbance* values.

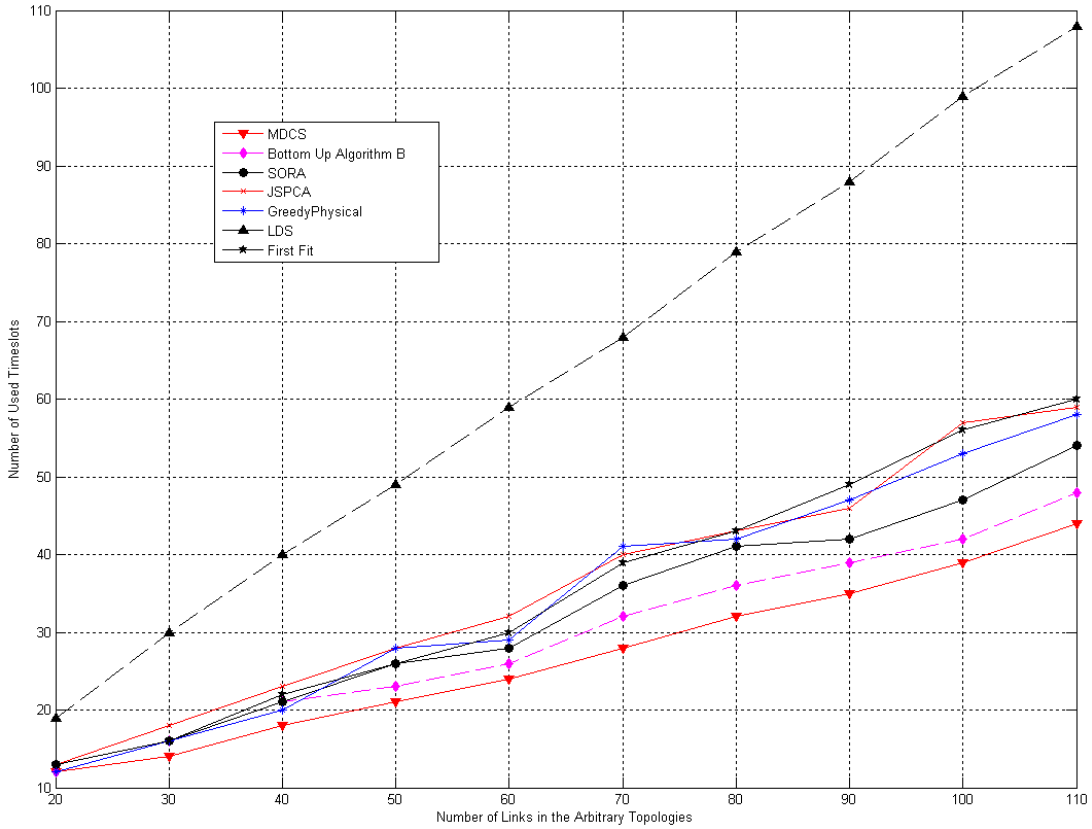


Figure 3-4: Link scheduling results comparisons ($\alpha = 5, \beta = 1$)

香港大學圖書館 HKU Libraries

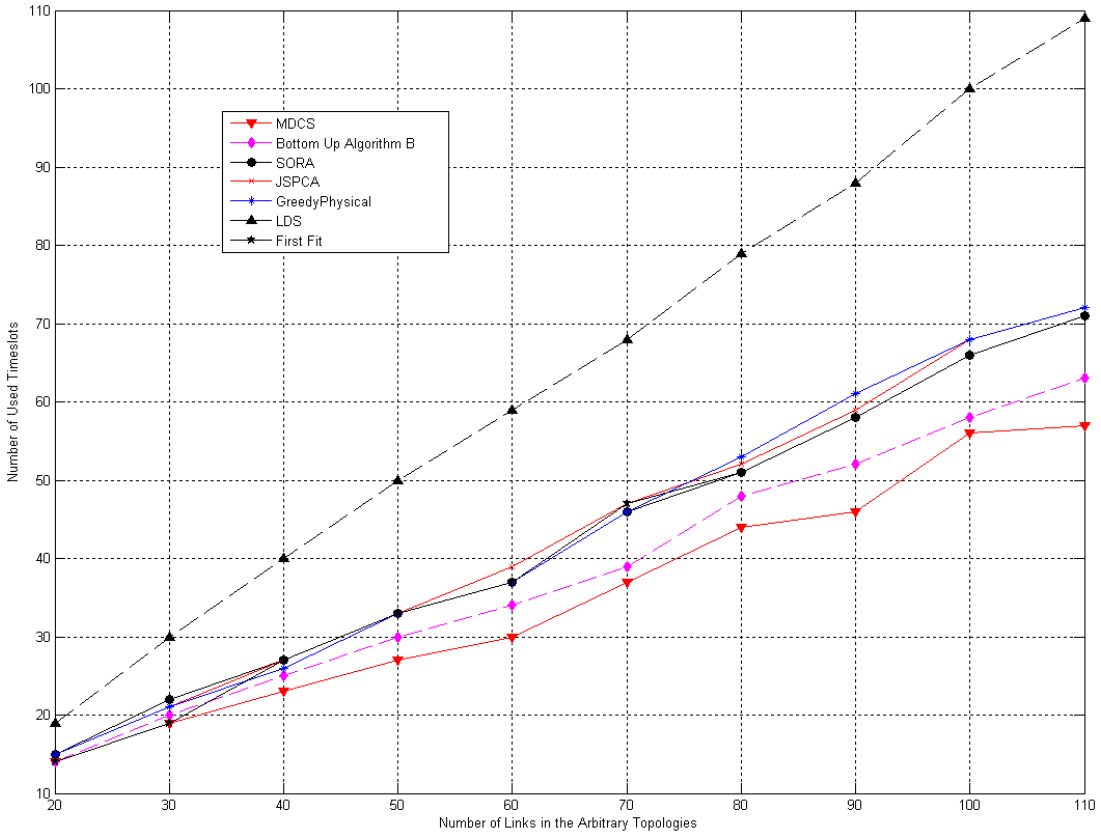


Figure 3-5: Link scheduling results comparisons ($\alpha = 5, \beta = 2$)

香港大學圖書館 HKU Libraries

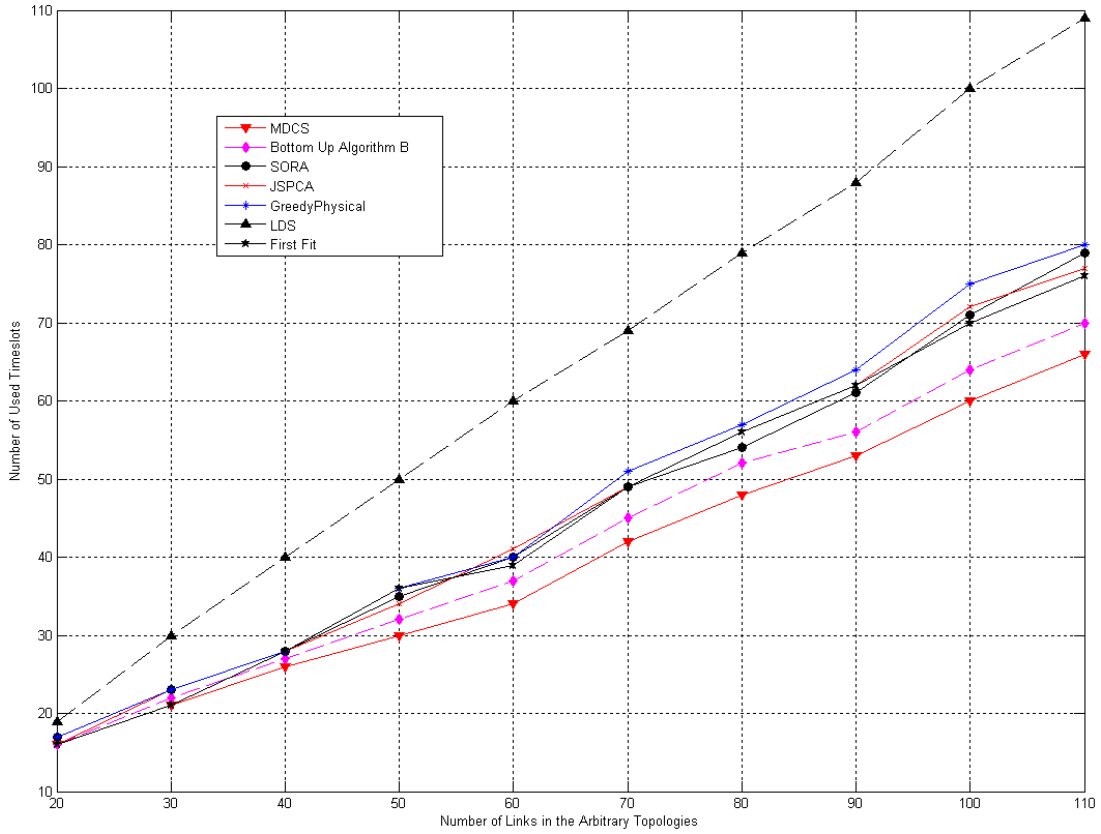


Figure 3-6: Link scheduling results comparisons ($\alpha = 5, \beta = 3$)

香港大學圖書館 HKU Libraries

Chapter 4 Exact and Approximate Link Scheduling Algorithms for the MFSAT Problem

Having discussed the heuristic link scheduling algorithms for the MFSAT problem, we turn to study the exact and approximate link scheduling algorithms in this chapter. Specifically, we propose two classes of exact and approximate link scheduling algorithms, one based on the relatively straightforward set covering, and the other on counting the number of different set covers. Throughout this chapter, we let $p(n)$ denote the time of checking whether the spectral radius of an irreducible non-negative matrix is smaller than 1 or not (Note that according to [95], $p(n) = O(n^3)$, but according to [93,94], faster algorithms maybe possible); then the time complexity for the counting based exact link scheduling algorithm called ESA_MFSAT is $O(3^n \cdot n \cdot \log^2 n \cdot p(n))$ with polynomial space, which represents a substantial improvement over the set covering based exact scheduling which needs time $O(2^{\max_i(r_i)})$. If exponential space is allowed, using either the fast zeta transform [101] or the fast subset convolution [99], the time complexity can be reduced to $O(2^n \cdot n \cdot \log^2 n \cdot p(n))$. Then based on the exact coloring and the maximum link independent set finding algorithms, we present three approximate link scheduling algorithms with approximation ratios $O(n/\log n)$, $O(n/\log^k n)$ and $\lceil (1+\varepsilon) \rceil$, respectively. Here ε is an arbitrary positive value independent of n.The time complexity of the first approximation algorithm is $O(n^2 \, poly \log(n))$ with polynomial space, the time complexity for the second

香港大學圖書館 HKU Libraries

algorithm is $O(n^{1+\log 3\cdot\log^{k-1} n}\, poly\log(n))$ with polynomial space, and the time complexity for the third algorithm is $O(((\binom{n}{n/2}) + 3^{e^{-c}n}\cdot n\cdot\log n)\cdot\log n\cdot p(n))$ with polynomial space.

The remainder of the chapter is organized as follows. We give a new formulation of the MFSAT problem in Section 4.1. In Section 4.2, we present some exact and approximate link scheduling algorithms based on link independent set covering. In Section 4.3, based on the inclusion-exclusion principle, we give the exact coloring algorithm ESA_MFSAT through counting the number of *k*-set coverings. Building upon these results, we present three approximate link scheduling algorithms in Section 4.4. Finally we conclude this chapter with some possible research directions in Section 4.5. Note that we will use the terms scheduling and coloring interchangeably throughout this chapter.

4.1 New Formulation for the MFSAT Problem

In this section, we will give a new formulation of the MFSAT problem, but first some related concepts need to be introduced.

DEFINITION 4.1: A set of non-adjacent links are called a **link independent set** if there exist a positive power vector P^* (cf. Proposition 2.4) satisfying all the SINR constraints; otherwise it is an **infeasible link independent set**.

DEFINITION 4.2: A **maximal link independent set** is a link independent set that is not a proper subset of any other link independent set.

DEFINITION 4.3: The largest maximal link independent set is called a **maximum link independent set**.

香港大學圖書館 HKU Libraries

PROPOSITION 4.4: Any superset of an infeasible link independent set is an infeasible link independent set; each subset of a link independent set is a link independent set.

Based on these definitions, we can rewrite the MFSAT problem as follows:

The MFSAT Problem: Given n arbitrarily distributed single-hop wireless links $N = \{1..n\}$, select a minimum number of link independent sets such that each link has at least one successful transmission under the SINR constraint.

4.2 Set Covering based Exact and Approximate Colorings

From the new formulation of the MFSAT problem we can see that it can be viewed as a kind of set covering problem. So in this section, we give some relatively straightforward exact and approximate link scheduling algorithms based on some traditional techniques such as the generation and test method, the backtracking search and the greedy set covering . Compared to the counting based exact and approximate link scheduling algorithms given in Sections 4.3 and 4.4, we will see that these traditional methods are inferior in terms of either the running time or the approximation ratio.

LEMMA 4.1: The number of maximal link independent sets in arbitrary link topologies is at most $\max_i \binom{n}{i} = \max(\binom{n}{\lfloor \frac{n+1}{2} \rfloor}, \binom{n}{\lceil \frac{n+1}{2} \rceil})$.

PROOF: According to Proposition 4.4, we know that the maximum number of maximal link independent sets equals $\max_i \binom{n}{i}$. Then by observing $\binom{n}{i} / \binom{n}{i-1} \geq 1$, we know that $\max_i \binom{n}{i} = \max(\binom{n}{\lfloor \frac{n+1}{2} \rfloor}, \binom{n}{\lceil \frac{n+1}{2} \rceil})$. This ends the proof.

香港大學圖書館 HKU Libraries

4.2.1 Set covering based exact coloring

Since there are at most $O(2^n)$ link independent sets, a naive brute force optimal covering (such as the generation and test method) takes time $O(2^{2^n})$. An improvement is to consider only the maximal link independent sets, but some post processing is needed since some links may then be scheduled more than needed. All the maximal link independent sets can be found in $O(2^n \cdot p(n))$, and from Lemma 4.1, the optimal set covering takes time $O(2^{\max_i(\binom{n}{i})})$.

4.2.2 Set covering based approximate coloring

This approximation algorithm proceeds as follows: In each timeslot, we find a maximum link independent set among the unscheduled wireless links; then we delete the maximum link independent set and continue until all the links have been scheduled. Actually, this is equivalent to the standard greedy set covering algorithm which is to select a set to maximize the uncovered elements, and the approximation ratio is $O(\log n)$. The decision version of the maximum link independent set finding problem in arbitrary link topologies has been shown to be NP-complete in [41], and an obvious brute force algorithm takes time $O(2^n \cdot p(n))$. For example, we can just enumerate all the k-combinations (k is from n to 1) of the n links, and then check whether they are link independent sets. If yes, we just stop there and output the k links [43]. But according to Proposition 4.4, with the help of binary search, we can give an exact maximum link independent set finding algorithm which takes time $O(\binom{n}{n/2} \cdot \log n \cdot p(n))$. From the Stirling's approximation for large factorials,

香港大學圖書館 HKU Libraries

the above complexity becomes $O(2^n \cdot \log n / \sqrt{n} \cdot p(n))$. This algorithm works as follows: we first check whether there exists a link independent set in all the $n/2$-combinations of the n links; if yes, we check the $3n/4$-combinations, otherwise we check $n/4$ -combinations. This continues until we find the maximum combination. In Section 4.4, we will give another exponential time approximation algorithm with a much better approximation ratio and without increasing the running time.

4.3 Counting based Exact Coloring

4.3.1 The Inclusion-Exclusion Principle

[folklore]: Let B be a finite set with subsets $A_1, A_2, ..., A_n \subseteq B$, and with the convention that $\bigcap_{i \in \varnothing} A_i = B$, then we know the number of elements in B which lie in none of the A_i is

$$|\bigcap_{i=1}^{n} \overline{A_i}| = \sum_{X \subseteq \{1,...,n\}} (-1)^{|X|} \cdot |\bigcap_{i \in X} A_i| . \quad (4.1)$$

Now let's define $S = \{S_1, S_2, ..., S_i, ...\}$ where $i \leq 2^n$ as the set of the link independent sets, B as the set of k-tuples $< S_1, ..., S_k >$, and $A_i \subseteq B$ as the set of those k-tuples whose union does not include link i; then the left hand side of Equation (4.1) can be interpreted as the number of k-tuples in B which cover all the links from $N = \{1, ..., n\}$. On the right hand side of Equation (4.1), for each X, $|\bigcap_{i \in X} A_i|$ means the number of k-tuples whose union does not include all the links in X.

香港大學圖書館 HKU Libraries

4.3.2 Counting the number of k-set-coverings

Here we define a *k*-set-covering as a set covering in which each covering consists of *k* link independent sets. Also we use $c_k(S)$ to denote the number of different *k*-set-coverings. We define $S(X) \subseteq S$ as the set of the link independent sets whose union does not include the links in X, which means $\bigcup_i S_i = N - X$, where $S_i \in S(X)$. And we use $s(X) = |\{S_i \in S : S_i \cap X = \varnothing\}|$ to denote the number of link independent sets in $S(X)$. Then the following lemma holds.

LEMMA 4.2: The number of different *k*-set coverings satisfies

$$c_k(S) = \sum_{X \subseteq N} (-1)^{|X|} \cdot (s(X))^k \tag{4.2}$$

PROOF: With $s(X)$ denoting the number of link independent sets in $S(X)$, $(s(X))^k$ stands for the number of different ways to choose *k* link independent sets from $S(X)$. (Note that the link independent sets in a *k*-set-covering may be non-distinct and non-disjoint.) Now combining the analyses in Section 4.3.1, we have $c_k(S) = |\bigcap_{i=1}^{n} \overline{A_i}|$, which is the left hand side of Equation (4.1), and $(s(X))^k = |\bigcap_{i \in X} A_i|$, which is the right hand side of Equation (4.1). This completes the proof.

THEOREM 4.3: Counting *k*-set coverings can be solved in $O(3^n \cdot p(n))$ time and polynomial space.

PROOF: According to Equation (4.2), we can see that the computational complexity is dominated by computing $s(X)$, i.e., to count the number of link

香港大學圖書館 HKU Libraries

independent sets in $S(X)$. For each X, we can enumerate all the combinations of the links in set $N - X$, which will take time $2^{n-|X|} \cdot p(n-|X|)$, because each testing of the link independent set takes time $p(n)$. Now combining with Equation (4.2) and the binomial theorem, the running time of counting k-set coverings is $T(c_k(S)) = \sum_{m=0}^{n} \binom{n}{m} \cdot 2^{n-m} \cdot p(n-m) = O(3^n \cdot p(n))$. Here $p(n)$ subsumes the time of raising each $s(X)$ to the k-th power. For the space complexity, since we compute $s(X)$ anew for each X, the occupied space is definitely polynomial. This finishes the proof.

THEOREM 4.4: Counting k-set coverings can be solved in $O(2^n \cdot p(n))$ time and $O(2^n \cdot n^2)$ space.

PROOF: We need to introduce the zeta transform of a function f, where f is an indicator function of the link independent set. Specifically, the zeta transform [101] $\hat{f}$ on the subset lattice $(2^N, \subseteq)$ of f is defined by

$$\hat{f}(X) = \sum_{S \subseteq X} f(S) \text{ for } X \subseteq N. \tag{4.3}$$

Now since $s(N-X) = \sum_{S \subseteq X} f(S) = \hat{f}(X)$, we can compute a table containing $s(N-X)$ for all $X \subseteq N$, and using a fast zeta transform introduced in [101], we can compute all $\hat{f}(X)$ with time $O(2^n \cdot p(n) + 2^n n) = O(2^n \cdot p(n))$. So according to Equation (4.2) and by subsuming the time of raising each $s(X)$ to the k-th power into $p(n)$, the time complexity of computing $c_k(S)$ is reduced to $O(2^n \cdot p(n))$. For the space complexity, since we have stored $O(2^n \cdot n)$ number of interim values for calculating $\hat{f}(X)$ (including

香港大學圖書館 HKU Libraries

all $s(X)$), and since the value of $s(X)$ can be up to 2^n, the space complexity is $O(2^n \cdot n^2)$. The proof is heavily relied on the fast zeta transform. For more details of this technique, please refer to lemma 2 in [101]. This ends the proof.

4.3.3 Computing the minimum number of colors

LEMMA 4.5: The **MFSAT** problem can be solved with k colors if and only if $c_k(S) > 0$.

PROOF: On one hand, if all the links can be scheduled with k colors, there must exist a valid k-set covering, which means $c_k(S) > 0$; on the other hand, if $c_k(S) > 0$, there must exist a coloring such that all the links can be scheduled at least once in k timeslots (colors). This finishes the proof.

Now we use $\chi(N)$ to denote the minimum number of colors to schedule all the $N = \{1,...,n\}$ links. Combining with Lemma 4.5, we have the following corollary.

COROLLARY 4.6: $\chi(N) = \min\{k : c_k(S) > 0\}$.

With the help of binary search, the time for computing $\chi(N)$ becomes $\log n \cdot T(c_k(S))$. So according to Theorems 4.3 and 4.4, we have the following corollaries.

COROLLARY 4.7: If we only allow polynomial space, the minimum number of colors $\chi(N)$ can be computed in time $O(3^n \cdot \log n \cdot p(n))$.

COROLLARY 4.8: If exponential space is allowed, the minimum number of colors $\chi(N)$ can be computed in time $O(2^n \cdot \log n \cdot p(n))$.

香港大學圖書館 HKU Libraries

4.3.4 The exact scheduling algorithm: ESA_MFSAT

Although we have computed the minimum number of colors to schedule all the links, we have not constructed a practical schedule yet. In this section, we present an algorithm called **ESA_MFSAT** for scheduling each link at least once while guaranteeing the minimum number of colors. (To demonstrate the use of this algorithm, we will give a detailed illustrating example in Section 4.3.6).

ESA_MFSAT: Exact Scheduling Algorithm for the MFSAT Problem

Input: A set of arbitrarily distributed single-hop wireless links $N = \{1,...,n\}$.

Output: A successful scheduling of all the links under the SINR constraint such that the number of colors is minimized.

1: Construct the pair-wise link conflict graph G_{pair} on N; // (cf. Section 3.3.1)

2: Compute $\chi(N)$, i.e., the minimum number of colors of N.

3: Pick the most constrained link i which has the maximum node degree in the conflict graph, and list all the links in G_{pair} not incident on i. These links form a set $\{j_1, j_2,..., j_m\}$. We construct new pair-wise conflict graphs called $G_{pair}(k)\,(1 \le k \le m)$ on top of G_{pair} by adding the edges between link i and links j_k where $1 \le k \le m$. Let $V(G_{pair})$ and $E(G_{pair})$ denote all the nodes and edges in G_{pair}, then $V(G_{pair}(k)) = V(G_{pair})$ and $E(G_{pair}(k)) = E(G_{pair}) \cup \{ij_1,...,ij_k\}$;

4: Let $S(k), 1 \le k \le m$ denote the set of all the link independent sets in N but excluding any link independent set containing link pairs incident on link i in $G_{pair}(k)$; similar to Corollary 4.6, we have $\chi(G_{pair}(k)) = \min\{k' : c_{k'}(S(k)) > 0\}$; and from Proposition 4.4, we have

$\chi(G_{pair}(k-1)) \le \chi(G_{pair}(k)) \le \chi(G_{pair}(k-1)) + 1$;

5: If $\chi(G_{pair}) = \chi(N) = \chi(G_{pair}(m))$, then we know the color of link i must be different from those of all the other links in some optimal coloring. So we give it a new color number and assign the sender of this link (or the senders of all

香港大學圖書館 HKU Libraries

the actual links if i is a virtual link) based on the Pareto-optimal power vector P^* (cf. Proposition 2.4), then we remove i from N. Otherwise, we can find the smallest k using binary search such that $\chi(G_{pair}(k)) = \chi(G_{pair}) + 1$.

In this case, we can deduce that link i must have the same color with link j_k in some optimal coloring (otherwise $\chi(G_{pair}(k)) = \chi(G_{pair})$). We now replace link i and link j_k with a new virtual link p_{ij_k}, and the neighbors of p_{ij_k} in the conflict graph become $N(p_{ij_k}) = N(i) + N(j_k)$.

6: Repeat step 2 to step 5 until all links have been scheduled (colored).

4.3.5 Correctness and time complexity analysis

We call step 2 to step 5 in the **ESA_MFSAT** algorithm a scheduling round. In each scheduling round, we remove one link, either directly giving it a new color or "contracting" two links (step 5). Since in each scheduling round, our link removals are based on the computed minimum number of colors of all the remaining links, and combining with the analyses in step 4 and step 5, we can guarantee the output is optimal, i.e., the number of colors we actually obtain is minimized.

Also, we need to emphasize the computation of the minimum number of colors $\chi(G_{pair}(k))$. Unlike the computation of $\chi(N)$, which is based on the set of all the link independent sets (e.g., the set S in Sections 4.3.1 and 4.3.2), the computation of $\chi(G_{pair}(k))$ is based on $S(k)$ (the set of the link independent sets in step 4). In addition, we must note that, if there are some virtual links in the conflict graph, due to the aggregate interference effect, all the actual links in these virtual links must be taken into account for checking whether the supersets of these links are link independent sets.

香港大學圖書館 HKU Libraries

We now analyze the time complexity of the ESA_MFSAT algorithm. First let some exponential function $T(\chi(n))$ denote the time of computing the minimum number of colors of scheduling n links. Since each scheduling round causes at most $O(\log n)$ computations of computing the minimum number of colors, and there are n scheduling rounds, the overall optimal scheduling takes time $O(T(\chi(n)) \cdot n \cdot \log n)$. So from Corollaries 4.7 and 4.8, if only polynomial space is allowed, the time complexity becomes $O(3^n \cdot n \cdot \log^2 n \cdot p(n))$, and if exponential space is allowed, the time complexity of the exact scheduling algorithm is $O(2^n \cdot n \cdot \log^2 n \cdot p(n))$.

4.3.6 An illustrating example for ESA_MFSAT

As shown in Figure 4-1, suppose there is a link topology with five links $N = \{1,2,3,4,5\}$, and all the maximal link independent sets have been computed: *{{1,3},{2,4},{3,5},{1,2,5}}*. Recall that X stands for any subset of N, and $S(X)$ represents the set of all the link independent sets in $N - X$ and $s(X)$ means the number of link independent sets in $S(X)$. For clarity of presentation, we use a simpler notation to denote the link independent sets in $S(X) \subseteq S$ (Table 4-1); for example, we use *1* to denote the link independent set *{1}*, and *125* to denote the link independent set *{1,2,5}*.

The 1st step of the **ESA_MFSAT** algorithm is to construct the pair-wise conflict graph G_{pair}, which is shown in Figure 4-2(a). The 2nd step is to compute the minimum number of colors $\chi(N)$. According to Table 4-1, we have *S={1,2,3,4,5,12,13,15,24,25,35,125}*, and we can calculate

香港大學圖書館 HKU Libraries

that $c_1(S) = c_2(S) = 0$, and $c_3(S) = \sum_{X \subseteq N} (-1)^{|X|} \cdot (s(X))^3 = 96 > 0$, and so we know that $\chi(N)$=3. In the 3rd step we pick the most constrained link *4* and add new edges (additional constraints) between link *4* and all the other links which are not incident on it. In this example, only one link $j_1 = 2$ (link *2*) is not incident on link *4*, so we add a new edge between them ($G_{pair}(1)$ as shown in Figure 4-2 (b)). In the 4th step, by removing all the link independent sets containing link pair *4* and *2*, we can achieve $S(1) = \{1,2,3,4,5,12,13,15,25,35,125\}$, and since $c_1(S(1)) = c_2(S(1)) = 0$ and $c_3(S(1)) = 30 > 0$ (based on Table 4-2), we conclude that $\chi(G_{pair}(1)) = 3$. Then we go to the 5th step, since we have known that $\chi(G_{pair}) = \chi(N) = \chi(G_{pair}(1)) = 3$, we can deduce that, in some optimal coloring, link *4* must have a different color with all the other links, and so we give it a new color number and remove it from *N*. Now we have finished the first scheduling round; we then repeat step 2 to step 5 until all links have been colored. We now briefly give the following scheduling rounds below.

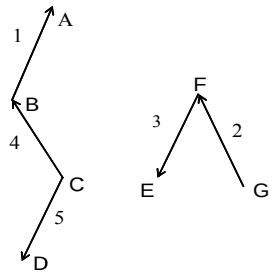


Figure 4-1: A link topology with five links

香港大學圖書館 HKU Libraries

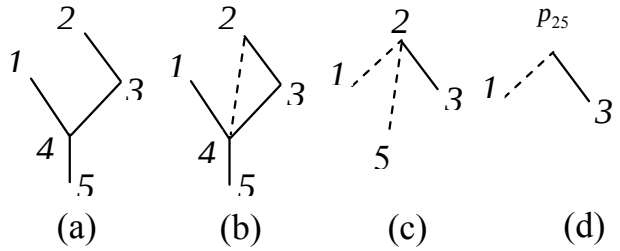


Figure 4-2: a) The original pair-wise conflict graph G_{pair} for the five links $N=\{1,2,3,4,5\}$; b) A new conflict graph $G_{pair}(1)$ constructed on G_{pair}; c) A new conflict graph $G_{pair}(2)$ constructed on the remaining links $N=\{1,2,3,5\}$; d) A new conflict graph $G_{pair}(1)$ constructed on the remaining links $N=\{1,p_{25},3\}$.

Table 4-1: For each subset X of $N=\{1,2,3,4,5\}$, the number of link independent sets $s(X)$ in $S(X)=N-X$

X	$S(X)$	$s(X)$
$\varnothing$	$\{1,2,3,4,5,12,13,15,24,25,35,125\}$	12
$\{1\}$	$\{2,3,4,5,24,25,35\}$	7
$\{2\}$	$\{1,3,4,5,13,15,35\}$	7
$\{3\}$	$\{1,2,4,5,12,15,24,25,125\}$	9
$\{4\}$	$\{1,2,3,5,12,13,15,25,35,125\}$	10
$\{5\}$	$\{1,2,3,4,12,13,24\}$	7
$\{1,2\}$	$\{3,4,5,35\}$	4
$\{1,3\}$	$\{2,4,5,24,25\}$	5
$\{1,4\}$	$\{2,3,5,25,35\}$	5
$\{1,5\}$	$\{2,3,4,24\}$	4
$\{2,3\}$	$\{1,4,5,15\}$	4
$\{2,4\}$	$\{1,3,5,13,15,35\}$	6
$\{2,5\}$	$\{1,3,4,13\}$	4
$\{3,4\}$	$\{1,2,5,12,15,25,125\}$	7
$\{3,5\}$	$\{1,2,4,12,24\}$	5
$\{4,5\}$	$\{1,2,3,12,13\}$	5
$\{1,2,3\}$	$\{4,5\}$	2
$\{1,2,4\}$	$\{3,5,35\}$	3
$\{1,2,5\}$	$\{3,4\}$	2

香港大學圖書館 HKU Libraries

{1,3,4}	*{2,5,25}*	3
{1,3,5}	*{2,4,24}*	3
{1,4,5}	*{2,3}*	2
{2,3,4}	*{1,5,15}*	3
{2,3,5}	*{1,4}*	2
{2,4,5}	*{1,3,13}*	3
{3,4,5}	*{1,2,12}*	3
{1,2,3,4}	*{5}*	1
{1,2,3,5}	*{4}*	1
{1,2,4,5}	*{3}*	1
{1,3,4,5}	*{2}*	1
{2,3,4,5}	*{1}*	1
{1,2,3,4,5}}	$\varnothing$	0

Table 4-2: For each subset X of $G_{pair}(1)$ where $N = \{1,2,3,4,5\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	*{1,2,3,4,5,12,13,15,25,35,125}*	11
{1}	*{2,3,4,5,25,35}*	6
{2}	*{1,3,4,5,13,15,35}*	7
{3}	*{1,2,4,5,12,15,25,125}*	8
{4}	*{1,2,3,5,12,13,15,25,35,125}*	10
{5}	*{1,2,3,4,12,13}*	6
{1,2}	*{3,4,5,35}*	4
{1,3}	*{2,4,5,25}*	4
{1,4}	*{2,3,5,25,35}*	5
{1,5}	*{2,3,4}*	3
{2,3}	*{1,4,5,15}*	4
{2,4}	*{1,3,5,13,15,35}*	6
{2,5}	*{1,3,4,13}*	4
{3,4}	*{1,2,5,12,15,25,125}*	7
{3,5}	*{1,2,4,12}*	4
{4,5}	*{1,2,3,12,13}*	5
{1,2,3}	*{4,5}*	2
{1,2,4}	*{3,5,35}*	3
{1,2,5}	*{3,4}*	2
{1,3,4}	*{2,5,25}*	3

香港大學圖書館 HKU Libraries

{1,3,5}	*{2,4}*	2
{1,4,5}	*{2,3}*	2
{2,3,4}	*{1,5,15}*	3
{2,3,5}	*{1,4}*	2
{2,4,5}	*{1,3,13}*	3
{3,4,5}	*{1,2,12}*	3
{1,2,3,4}	*{5}*	1
{1,2,3,5}	*{4}*	1
{1,2,4,5}	*{3}*	1
{1,3,4,5}	*{2}*	1
{2,3,4,5}	*{1}*	1
{1,2,3,4,5}}	$\varnothing$	0

The 2nd scheduling round: In the 2nd step, similar to Table 4-1, we can construct another Table 4-3 for $N = \{1,2,3,5\}$, and we have $S=\{1,2,3,5,12,13,15,25,35,125\}$. Then we can calculate that $c_1(S) = 0$ and $c_2(S) = 10 > 0$, and so we know that $\chi(N) = \chi(G_{pair}) = 2$. In the 3rd step, we pick link *2* as the most constrained link, and add new edges between link *2* and links $j_1 = 1$ and $j_2 = 5$ ($G_{pair}(2)$ as shown in Figure 4-2(c)). Note that by reducing the edge between links *2* and *5* from $G_{pair}(2)$, we can get $G_{pair}(1)$. In the 4th and 5th steps, we have $S(2) = \{1,2,3,5,13,15,35\}$ and since $c_2(S(2)) = 0$ and $c_3(S(2)) = 36 > 0$ (based on Table 4-4), we know $\chi(G_{pair}(2)) = 3 > \chi(N)$, and then we continue to find that $S(1) = \{1,2,3,5,13,15,25,35\}$ and since $c_2(S(1)) = 2 > 0$ (based on Table 4-5), we get $\chi(G_{pair}(1)) = 2 = \chi(N)$. So in this case, we conclude that $k = 2$ (corresponding to link j_2) is the smallest k to satisfy $\chi(G_{pair}(k)) = \chi(G_{pair}) + 1$. We then deduce that link *2* must have the

香港大學圖書館 HKU Libraries

same color with link 5 in this optimal coloring. So we contract these two links into a new link p_{25}. Then we go to the 3rd scheduling round.

Table 4-3: For each subset X of $N = \{1,2,3,5\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	$\{1,2,3,5,12,13,15,25,35,125\}$	10
$\{1\}$	$\{2,3,5,25,35\}$	5
$\{2\}$	$\{1,3,5,13,15,35\}$	6
$\{3\}$	$\{1,2,5,12,15,25,125\}$	7
$\{5\}$	$\{1,2,3,12,13\}$	5
$\{1,2\}$	$\{3,5,35\}$	3
$\{1,3\}$	$\{2,5,25\}$	3
$\{1,5\}$	$\{2,3\}$	2
$\{2,3\}$	$\{1,5,15\}$	3
$\{2,5\}$	$\{1,3,13\}$	3
$\{3,5\}$	$\{1,2,12\}$	3
$\{1,2,3\}$	$\{5\}$	1
$\{1,2,5\}$	$\{3\}$	1
$\{1,3,5\}$	$\{2\}$	1
$\{2,3,5\}$	$\{1\}$	1
$\{1,2,3,5\}$	$\varnothing$	0

Table 4-4: For each subset X of $G_{pair}(2)$ where $N = \{1,2,3,5\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	$\{1,2,3,5,13,15,35\}$	7
$\{1\}$	$\{2,3,5,35\}$	4
$\{2\}$	$\{1,3,5,13,15,35\}$	6
$\{3\}$	$\{1,2,5,15\}$	4
$\{5\}$	$\{1,2,3,13\}$	4
$\{1,2\}$	$\{3,5,35\}$	3
$\{1,3\}$	$\{2,5\}$	2
$\{1,5\}$	$\{2,3\}$	2
$\{2,3\}$	$\{1,5,15\}$	3

香港大學圖書館 HKU Libraries

{2,5}	*{1,3,13}*	3
{3,5}	*{1,2}*	2
{1,2,3}	*{5}*	1
{1,2,5}	*{3}*	1
{1,3,5}	*{2}*	1
{2,3,5}	*{1}*	1
{1,2,3,5}	$\varnothing$	0

Table 4-5: For each subset X of $G_{pair}(1)$ where $N = \{1,2,3,5\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	*{1,2,3,5,13,15,25,35}*	8
{1}	*{2,3,5,25,35}*	5
{2}	*{1,3,5,13,15,35}*	6
{3}	*{1,2,5,15,25}*	5
{5}	*{1,2,3,13}*	4
{1,2}	*{3,5,35}*	3
{1,3}	*{2,5,25}*	3
{1,5}	*{2,3}*	2
{2,3}	*{1,5,15}*	3
{2,5}	*{1,3,13}*	3
{3,5}	*{1,2}*	2
{1,2,3}	*{5}*	1
{1,2,5}	*{3}*	1
{1,3,5}	*{2}*	1
{2,3,5}	*{1}*	1
{1,2,3,5}	$\varnothing$	0

The 3rd scheduling round: In the 2nd step, also similar to Table 4-1, we can construct another Table 4-6 for $N = \{1, p_{25}, 3\}$, and we have $S = \{1, p_{25}, 3, 1p_{25}, 13\}$, and then we can calculate that $c_1(S) = 0$ and $c_2(S) = 6 > 0$, and so we know that $\chi(N) = \chi(G_{pair}) = 2$. In the 3rd step, we pick link p_{25} as the most constrained link, and add a new edge

香港大學圖書館 HKU Libraries

between link p_{25} and link $j_1 = 1$ ($G_{pair}(1)$ as shown in Figure 4-2(d)). In the 4th and 5th steps, we have $S(1) = \{1, p_{25}, 3, 13\}$ and since $c_2(S(1)) = 2 > 0$ (based on Table 4-7), we obtain $\chi(G_{pair}(1)) = 2 = \chi(N)$, and so we conclude that link p_{25} must have a different color with link *1* and link *3* in this optimal coloring. Then we give it a new color and remove it from *N*. Now we finish the 3rd scheduling round and can proceed to the 4th scheduling round.

The 4th scheduling round: We can easily find that links *1* and *3* must have the same color in this optimal coloring (the interested reader can do the checking). So we give them a new color and we finish the scheduling of all the links. Also the transmission powers of all the links are based on the Pareto-optimal power vector P^* (cf. Proposition 2.4). The final result is we have used three colors for the link independent sets *{4},{2,5}* and *{1,3}*. Of course, this is only one of the optimal colorings. By choosing different coloring strategies or through choosing different orders of the links in step 3 of the **ESA_MFSAT** algorithm, we may obtain different optimal colorings.

Table 4-6: For each subset X of $N = \{1, p_{25}, 3\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	$\{1, p_{25}, 3, 13, 1\, p_{25}\}$	5
{1}	$\{p_{25}, 3\}$	2
$\{p_{25}\}$	*{1,3,13}*	3
{3}	$\{1, p_{25}, 1\, p_{25}\}$	3
{1, p_{25} *}*	*{3}*	1
{1,3}	$\{p_{25}\}$	1
$\{p_{25}, 3\}$	*{1}*	1
{1, p_{25} *,3}*	$\varnothing$	0

香港大學圖書館 HKU Libraries

Table 4-7: For each subset X of $G_{pair}(1)$ where $N = \{1, p_{25}, 3\}$, the number of link independent sets $s(X)$ in $S(X) = N - X$

X	$S(X)$	$s(X)$
$\varnothing$	$\{1, p_{25}, 3, 13\}$	4
$\{1\}$	$\{p_{25}, 3\}$	2
$\{p_{25}\}$	$\{1, 3, 13\}$	3
$\{3\}$	$\{1, p_{25}\}$	2
$\{1, p_{25}\}$	$\{3\}$	1
$\{1, 3\}$	$\{p_{25}\}$	1
$\{p_{25}, 3\}$	$\{1\}$	1
$\{1, p_{25}, 3\}$	$\varnothing$	0

4.4 Counting based Approximate Colorings

4.4.1 Polynomial time approximation

This approximation algorithm is implemented by clustering. We first partition all the n links into $n/\log n$ groups, each group containing $\log n$ links. Then we use the exponential space version of the **ESA_MFSAT** algorithm to compute the minimum number of colors of each group. Let Opt stand for the minimum number of colors, then the actual number of colors we get is at most $n/\log n \cdot Opt$, and so the approximation ratio is $O(n/\log n)$. Since the time complexity of our exact scheduling algorithm is $O(2^n \cdot n \cdot \log^2 n \cdot p(n))$, and the space complexity is $O(2^n \cdot n^2)$, the time complexity of our approximate scheduling algorithms is bounded by $O(n^2 \cdot poly\log(n))$. The space complexity is $O(n^2 \cdot \log n)$.

香港大學圖書館 HKU Libraries

4.4.2 Quasi-polynomial time approximation

Obviously, we can also partition all the n links into $n/\log^k n$ groups, each group containing $\log^k n$ links. Then we use the polynomial space version of the **ESA_MFSAT** algorithm to compute the minimum number of colors of each group. The approximation ratio is $O(n/\log^k n)$. But the time complexity becomes $O(n^{1+\log 3\cdot\log^{k-1} n}\, poly\log(n))$, which is a quasi-polynomial time complexity, i.e., the complexity with the form $O(\exp((\log n)^{O(1)})$. The space complexity is still polynomial.

4.4.3 Exponential time approximation

We have given an exponential time approximate link scheduling algorithm in Section 4.2.2, which is based on repeatedly finding the maximum link independent set on the remaining links. This is equivalent to a standard greedy set covering method with approximation ratio $O(\log n)$. In this section, we will present another exponential time approximation algorithm which is also based on finding the maximum link independent set. But in this algorithm, when the number of the remaining links equals $e^{-\varepsilon} n$, we do not repeat the maximum link independent set finding algorithm. Instead we use our polynomial space version of the exact link scheduling algorithm **ESA_MFSAT** since the number of the remaining links has become small enough.

THEOREM 4.9: The approximation ratio of this polynomial space approximate link scheduling algorithm is $\lceil (1+\varepsilon) \rceil$, and the time complexity of this algorithm is $O((\binom{n}{n/2} + 3^{e^{-\varepsilon} n} \cdot n \cdot \log n) \cdot \log n \cdot p(n))$.

香港大學圖書館 HKU Libraries

PROOF: The proof is adapted from [100,101].

Let Opt be the minimum number of colors for scheduling all the links. And let s be the number of maximum link independent sets we have removed. If we use $\chi(e^{-\varepsilon}n)$ to denote the minimum number of colors we have obtained to schedule the remaining $e^{-\varepsilon}n$ links, then the total number of colors we have used is $\chi(e^{-\varepsilon}n)+s$. Since $\chi(e^{-\varepsilon}n)\le Opt$, we only need to prove that $s\le\lceil\varepsilon\rceil\cdot Opt$.

Since we remove the maximum link independent set in each step, so after at most t steps, the number of remaining links is smaller than or equal to $n\cdot(1-1/Opt)^t$, and due to a standard inequality, we have $n\cdot(1-1/Opt)^t\le n\cdot e^{-t/Opt}$. So if $\varepsilon\cdot Opt\le t\le s\le\lceil\varepsilon\rceil\cdot Opt$, then the number of remaining links is at most $e^{-\varepsilon}n$. By plugging into the time complexity result of the maximum link independent set finding algorithm in Section 4.2.2 and the polynomial space version of the exact scheduling algorithm in Section 4.3.4, we finish the proof.

香港大學圖書館 HKU Libraries

Chapter 5 Exact and Approximate Link Scheduling Algorithms for the MLSAT Problem

In this chapter, we will first transform the MLSAT problem as a set multi-cover problem. Second, we will design a first known exact algorithm for the set multi-cover problem. Third, based on this exact algorithm, we will present a polynomial time polynomial space approximation algorithm for the MLSAT problem. To our knowledge, this is the first known approximation algorithm for the MLSAT problem that is independent of the links' lengths.

5.1 New Formulation for the MLSAT Problem

Similar to Section 4.1, in this section, we will transform the MLSAT problem as a set multi-cover problem. Now based on the same definitions given in Section 4.1, we can rewrite the MLSAT problem as follows:

The MLSAT Problem: Given n arbitrarily distributed single-hop wireless links $N = \{1..n\}$, select a minimum number of link independent sets such that each link has been covered at least a number of times as specified in its coverage requirement, namely, the number of packets each link needs to transmit.

From this new formulation, we can see that the MLSAT problem is actually the same as the set multi-cover problem. Set multi-cover is a generalization of the set cover problem where each element may need to be covered more than once and thus some subset in the given family of subsets may be picked several times for minimizing the number of sets to satisfy the coverage requirement. In this chapter, based on the inclusion-exclusion

香港大學圖書館 HKU Libraries

principle, we will propose a first known exact algorithm for the set multi-cover problem. Specifically, the presented ESMC (Exact Set Multi-Cover) algorithm takes $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space where t is the maximum value in the coverage requirement vector (The $O^*(f(n))$ notation omits a $poly\log(f(n))$ factor).

5.2 Related Work

Recently it has been shown that for some exact algorithms, using the inclusion-exclusion principle can significantly reduce the running time. For example, Björklund et al. have applied the inclusion-exclusion principle to various set cover and set partition problems, obtaining time complexities that are much lower than those of previous algorithms [100,101,104]. This principle has also been used in some early chapters, such as [128] and [131]. By using the Möbius inversion technique which is an algebraic equivalent of the inclusion-exclusion principle, Björklund et al. give a fast algorithm for the subset convolution problem [99] and Nederlof presents a family of fast polynomial space algorithms for the Steiner Tree problem and other related problems [132]. In this chapter, we are interested in designing inclusion-exclusion based exact algorithms for the set multi-cover problem [135,136]. This problem is a generalization of the set cover problem in which each element needs to be covered by a specified integer number of times and each set can be picked multiple times in order to satisfy the coverage requirement. It is a bit surprising that only approximation algorithms have so far been proposed for the set multi-cover problem. In fact, by using the same greedy strategy as for the set cover problem, which is to repeatedly add the

香港大學圖書館 HKU Libraries

set containing the largest number of uncovered elements to the cover, one can achieve the same $O(\log n)$ approximation for the problem [135]. Feige shows that the set cover problem can not be approximated better than $\ln n$ unless $NP \in DTIME(n^{\log\log n})$ [130]. Some parallel approximation algorithms for the set cover problem and its generalizations, such as the set multi-cover problem, the multi-set multi-cover problem and the covering integer programs problem have been presented in [136]. In all these related work on approximation solutions, the set multi-cover problem appears to be no harder than the set cover problem. In this chapter, we will see that finding an exact solution for the set multi-cover problem can take much longer time than that for the fastest exact algorithm for the set cover problem [100,101].

The structure of this chapter is as follows. In Section 5.3, we give a formal definition of the set multi-cover problem. In Section 5.4, based on the inclusion-exclusion principle, we will transform the set multi-cover problem to the problem of counting the number of k-tuples that satisfy the integral coverage requirements. We then give an algorithm for counting these numbers of k-tuples in Section 5.5. In Section 5.6, we give a constructive algorithm for finding the minimum number of sets that meet the coverage requirements. A simple illustrating example for our algorithms is given in Section 5.7. We finally give a polynomial time polynomial space approximate algorithm for the MLSAT problem in Section 5.8.

5.3 The Set Multi-cover Problem

A summary of the various notations used in this chapter and their corresponding definitions is given in Table 5-1. Throughout the chapter, we

香港大學圖書館 HKU Libraries

let the union of a k-tuple $< s_1,...,s_k >$ which is denoted as $C = \bigcup_{i=1}^{k} s_i$ represent a multi-set. This means that we just put all the elements in each s_i into the set C without removing duplicated elements.

Table 5-1: Summary of notations and their definitions

Notation	Definition
N	The universe set, where $N = \{1,...,n\}$ and $\mid N \mid = n$.
F	A family of subsets of N, where $F = \{s_1,...,s_{\mid F \mid}\}$ and $\mid F \mid$ is the total number of subsets in F.
T	The integral coverage requirement vector, where $T = (t_1,...,t_n)$; each $i \in N$ must be covered at least $t_i \geq 1$ times in the picked subsets over F.
t	The maximum integer in the vector T, i.e., $t = \max_{1 \leq i \leq n}(t_i)$.
$c_k(F)$	The number of k-tuples $< s_1,...,s_k >$ over F such that the union of each k-tuple, i.e., $C = \bigcup_{i=1}^{k} s_i$, satisfy the specified coverage requirement T.
$n_k(X)$	The number of k-tuples $< s_1,...,s_k >$ over F such that each $i \in X$ ($X \subseteq N$) appears at most $(t_i - 1)$ number of times in the multi-set C.
$a(X)$	The number of subsets in F that avoid X.
$b(X,Y)$	The number of subsets in F that include Y but avoid $X \setminus Y$.
$p_q^X(n_1,...,n_{\mid X \mid})$ or $p_q^X(n_X)$	The number of q-tuples over F such that each $j \in X$ appears n_j times in the union of each q-tuple. For simplicity, we use n_X to denote $\{n_1,...,n_{\mid X \mid}\}$.

香港大學圖書館 HKU Libraries

The Set Multi-Cover Problem: Let $N = \{1,...,n\}$ be the universe, and F a given family of subsets $\{s_i\}$ over N, and the union of all the subsets in F covers all the elements in N. A legal (k,T) cover is a collection of k subsets over F such that each $i \in N$ must appear at least $t_i \geq 1$ times in the union of the k subsets. Note that the k subsets can be non-distinct which means that some subsets in F can be picked several times. The goal of the set multi-cover problem is to find the minimum k to make a legal (k,T) multi-cover.

Remark 5.1: Since each subset in F can contain each element of N at most once, in order to find a legal (k,T) cover, k must be greater than or equal to t, the maximum integer in the coverage requirement vector T, i.e., $k \geq t$. Also, since the union of F covers all the elements in N, we have $k \leq tn$.

5.4 Counting based Exact Algorithm for the Set Multi-Cover Problem

5.4.1 The Inclusion-Exclusion Principle

This principle has been given in Section 4.3.1. For convenience, we present it again here. Let B be a finite set with subsets $A_1, A_2, ..., A_n \subseteq B$. With the convention that $\bigcap_{i \in \varnothing} A_i = B$, the number of elements in B which lie in none of the A_i is :

$$|\bigcap_{i=1}^{n} \overline{A_i}| = \sum_{X \subseteq N} (-1)^{|X|} \cdot |\bigcap_{i \in X} A_i| \qquad (5.1)$$

香港大學圖書館 HKU Libraries

5.4.2 Counting the number of k-tuples

LEMMA 5.1: Let $n_k(X)$ denote the number of k-tuples $< s_1,...,s_k >$ where for each $j \in X$, the number of j in the set $C = \bigcup_{i=1}^{k} s_k$ is at most $t_j - 1$; then the number of k-tuples that satisfy the coverage requirement T can be computed from the following equation:

$$c_k(F) = \sum_{X \subseteq N} (-1)^{|X|} \cdot n_k(X) \tag{5.2}$$

PROOF: Let B be the set of *k*-tuples $< s_1,...,s_k >$ from *F*, and let A_i be the set of *k*-tuples where element *i* in the multi-set *C* appears at most $(t_i - 1)$ times. The left side of Equation (5.1) is the number of *k*-tuples in which each element *i* in the universe *N* is covered at least t_i times, which is represented by $c_k(F)$, the left side of Equation(5.2). Accordingly, $|\bigcap_{i \in X} A_i|$ is the number of *k*-tuples in which each $j \in X$, which is an element in the set *C*, appears at most $(t_j - 1)$ times; i.e., $n_k(X) = |\bigcap_{i \in X} A_i|$. By the right side of Equation(5.1), we can derive the right side of Equation (5.2).

LEMMA 5.2: We can find a legal (k,T) multi-cover if and only if $c_k(F) > 0$.

PROOF: $c_k(F)$ is the number of *k*-tuples over *F* that satisfy the coverage requirement *T*. The number of legal (k,T) multi-covers is the number of *k* subsets over *F* that satisfy the coverage requirement *T*. Since different orderings of the *k* subsets mean different *k*-tuples while the (k,T) multi-cover concerned remains the same, we know that only when $c_k(F) > 0$ can there be

香港大學圖書館 HKU Libraries

a legal (k,T) multi-cover. Similarly, if there is a legal (k,T) multi-cover, it guarantees that $c_k(F) > 0$. This finishes the proof.

According to Lemma 5.2, we have the following corollary.

COROLLARY 5.3: The minimum k value to make a legal (k,T) multi-cover is equal to the minimum k value that satisfies $c_k(F) > 0$.

Thus we can transform the set multi-cover problem to the problem of computing $c_k(F)$. By using binary search, since $t \le k \le tn$, the time for solving the set multi-cover problem equals the sum of the times for computing the $O(\log(tn))$ numbers of $c_k(F)$. In the next section, we will introduce an algorithm for computing $c_k(F)$.

5.5 An Algorithm for Computing $c_k(F)$

In this section, we show how to compute $c_k(F)$, i.e., to count the number of k-tuples $< s_1,...,s_k >$ over F such that the union of each such k-tuple satisfies the given coverage requirement T.

5.5.1 How to compute $n_k(X)$

According to Equation(5.2), we know that the crux of computing $c_k(F)$ is to obtain $n_k(X)$, i.e., the number of k-tuples over F such that each $i \in X$ appears at most $(t_i - 1)$ times in the union of every k-tuple. Without loss of generality, we assume $X = \{1,2,...,m\}$, and for the simplicity of notation, we let $n_X = \{n_1, n_2,..., n_m\}$. We then denote $p_q^X(n_X) = p_q^X(n_1, n_2,..., n_m)$, the number of q-tuples over F such that for each $j \in X$ the number of the element j in the

香港大學圖書館 HKU Libraries

union of every q-tuple is n_j. Now since the union of each q-tuple can cover each $j \in X$ at most q times, for each $p_q^X(n_1, n_2, ..., n_m)$, we have $n_j \leq q$ for each $j \in X$; otherwise, $p_q^X(n_1, n_2, ..., n_m)$ equals 0. From these definitions, we can easily obtain the following Equation(5.3). This equation means that, in order to obtain $n_k(X)$, we should sum all the $p_k^X(n_X)$ values ($\prod_{i=1}^{m} t_i$ of them), where $p_k^X(n_X)$ is from $p_k^X(0,0,...,0)$ to $p_k^X(t_1 - 1, t_2 - 1, ..., t_m - 1)$. Now our problem becomes how to efficiently compute all the $p_k^X(n_X)$ values.

$$n_k(X) = \sum_{\substack{0 \leq n_j \leq t_j - 1 \\ 1 \leq j \leq m}} p_k^X(n_X) \tag{5.3}$$

Before delving into the details of calculating all these $p_k^X(n_X)$ values, we need to introduce some notations. We use $a(X)$ to denote the number of sets in F that avoid X where $X \subseteq N$, and $b(X,Y)$ to denote the number of sets in F that include Y but avoid $X \setminus Y$, where $Y \subseteq X$. We show next how to get $a(X)$ for all X and $b(X,Y)$ for all X and Y.

5.5.2 How to compute all $a(X)$

There are two ways to compute $a(X)$. The first way is to use the fast zeta transform technique introduced in [101]. By using this technique, all $a(X)$ values can be computed in $O^*(2^n)$ time. And since the technique uses a look-up table to store all the interim values including $a(X)$ for all $X \subseteq N$, it requires $O^*(2^n)$ space. The second way is to compute $a(X)$ directly without storing all the interim values into a look-up table. In order to compute $a(X)$ where $X \subseteq N$, we just need to test every subset $S \subseteq N \setminus X$ to

香港大學圖書館 HKU Libraries

see if S is in F, which takes time $O^*(2^{n-|X|})$ by assuming that the membership test in F can be decided in polynomial time and polynomial space (This is true for our MLSAT problem since checking whether a set of links is a link independent set can be transformed into checking whether the spectral radius of the links' link gain matrix is smaller than 1 or not.). Then for all $X \subseteq N$, the total time for computing $a(X)$ equals $\sum_{X \subseteq N} O^*(2^{n-|X|}) = O^*(\sum_{r=0}^{n} C_n^r 2^{n-r}) = O^*(3^n)$.

5.5.3 How to compute all $b(X,Y)$

Based on the two different ways of computing $a(X)$, we have two corresponding ways to compute all $b(X,Y)$ for all $Y \subseteq X$ and for all $X \subseteq N$.

For arbitrary X and Y, where $Y \subseteq X$, we let $|X| = m$ and $|Y| = r$ and $r \leq m$. Without loss of generality, assume $X = \{1,2,...,m\}$ and $Y = \{1,2,...,r\}$. Then $b(X,Y)$ can be computed via Equation (5.4).

$$b(X,Y) = \sum_{Z \subseteq Y} (-1)^{|Z|} \cdot a(Z \cup (X \setminus Y)) = \sum_{Z \subseteq Y} (-1)^{|Z|} \cdot a(Z \cup \{r+1,...,m\}) \quad (5.4)$$

Equation (5.4) is obtained by applying the inclusion-exclusion principle. Suppose B is a family of subsets of F which avoid $X \setminus Y$, and let $A_i \subseteq B$ ($i \in Y \subseteq X$) be the family of subsets which further avoid element i. Then the left side of Equation (5.1) ($|\bigcap_{i=1}^{|Y|} \overline{A_i}|$) is the number of sets in F that cover Y but avoid $X \setminus Y$ which is the value of $b(X,Y)$. Accordingly, the right side of Equation (5.1) ($|\bigcap_{i \in Z \subseteq Y} A_i|$) is the number of sets in F that avoid $Z \cup (X \setminus Y)$ which is the value of $a(Z \cup (X \setminus Y))$. Thus according to

香港大學圖書館 HKU Libraries

Equation (5.1), we have Equation (5.4). Then we calculate how much time we need to compute all $b(X,Y)$.

First, we do not use a table to store all $a(X)$ values, and the time complexity is given in Lemma 5.4.

Lemma 5.4: For all $Y \subseteq X$ and for all $X \subseteq N$, $b(X,Y)$ can be obtained in $O^*(6^n)$ time and polynomial space.

PROOF: As mentioned earlier, in order to compute $a(X)$ where $X \subseteq N$, we just need to test every subset $S \subseteq N \setminus X$ to see if S is in F, which takes time $O^*(2^{n-|X|})$. For given X and Y, according to Equation (5.4), the time for computing $b(X,Y)$ can be calculated from the formula $\sum_{i=0}^{r} C_r^i \cdot O^*(2^{n-i-m+r})$. By using the Binomial theorem, we have Equation (5.5).

$$\sum_{i=0}^{r} C_r^i \cdot O^*(2^{n-i-m+r}) = O^*(2^{n-m} \cdot 3^r) \tag{5.5}$$

Now for all $Y \subseteq X$, the time for computing $b(X,Y)$ can be calculated through the formula $\sum_{r=0}^{m} C_m^r \cdot O^*(2^{n-m} \cdot 3^r)$. Similarly, by using the Binomial theorem, we have Equation (5.6).

$$\sum_{r=0}^{m} C_m^r \cdot O^*(2^{n-m} \cdot 3^r) = O^*(2^{n+m}) \tag{5.6}$$

Finally, for all $X \subseteq N$, the time for computing $b(X,Y)$ can be calculated through the formula $\sum_{m=0}^{n} C_n^m O^*(2^{n+m})$. Again by the Binomial theorem, we have Equation (5.7).

$$\sum_{m=0}^{n} C_n^m O^*(2^{n+m}) = O^*(6^n) \tag{5.7}$$

According to the computation steps of Equations (5.5), (5.6) and (5.7), since we did not use any look-up table to store the exponential number

香港大學圖書館 HKU Libraries

of $a(X)$ values to speed up the calculation of $b(X,Y)$, the space used is only polynomial. This completes the proof.

Now we give another way to compute all $b(X,Y)$ by using exponential space. Its time and space complexities are given in Lemma 5.5.

Lemma 5.5: For all $Y \subseteq X$ and for all $X \subseteq N$, $b(X,Y)$ can be obtained in $O(4^n)$ time and $O^*(2^n)$ space.

PROOF: As before, by using the fast zeta transform technique introduced in [101], all $a(X)$ values can be computed in $O^*(2^n)$ time and $O^*(2^n)$ space. Then for some given X and Y, according to Equation (5.4), since all $a(X)$ values are known, $b(X,Y)$ can be computed in time 2^r where $r = |Y|$. The time for computing $b(X,Y)$ for all $Y \subseteq X$ equals $\sum_{r=0}^{m} C_m^r \cdot 2^r = 3^m$. Similarly, the time for computing $b(X,Y)$ for all $X \subseteq N$ equals $\sum_{m=0}^{n} C_n^m \cdot 3^m = 4^n$. This finishes the proof.

5.5.4 An Algorithm for computing all $p_k^X(n_X)$

As mentioned in Section 5.4.1, we need to compute $\prod_{i=1}^{m} t_i$ $p_k^X(n_X) = p_k^X(n_1, n_2, ..., n_m)$ values, where $0 \le n_i \le t_i - 1$ and $1 \le i \le m$. Without loss of generality, we assume the positive integers in $\{n_1, n_2, ..., n_m\}$ form a set $n_Y = \{n_1, ..., n_r\}$, where $Y = \{1,2,...,r\}$ and $0 \le r \le m$. Then from the definitions of $a(X)$ and $b(X,Y)$, we have $p_1^X(n_1, n_2, ..., n_m) = b(X, \{1,2,...,r\})$ and $p_1^X(0,0,...,0) = a(X)$. Now for brevity of notation, for any subset $Z = \{r_1, ..., r_i\} \subseteq Y$, we use $(n_X - 1^Z)$ to denote the

香港大學圖書館 HKU Libraries

set $\{n_1,...,n_{r_1}-1,...,n_{r_i}-1,n_{r_{i+1}},...,n_m\}$, i.e., for all $j \in Z$, the corresponding n_j values are decremented by 1, and for all $j \notin Z$, we keep the corresponding n_j values. Then for $2 \le q \le k$, we use the following recursive function to obtain $p_q^X(n_X)$.

$$p_q^X(n_X) = \sum_{Z \subseteq Y} b(X,Z) \cdot p_{q-1}^X(n_X - 1^Z) \tag{5.8}$$

Basically, this equation tells us how to calculate the $p_q^X(n_X)$ value when given $p_{q-1}^X(n_X - 1^Z)$ values for all $Z \subseteq Y$. For example, when $Z = \varnothing$, $b(X,\varnothing) = a(X)$ and $p_{q-1}^X(n_X - 1^Z) = p_{q-1}^X(n_X)$. We already know $a(X)$ means the number of sets in F that avoid X, and $p_{q-1}^X(n_X)$ means the number of (q-1)-tuples from F where for each $j \in X$ the number of the element j in the union of every (q-1)-tuple is n_j ; thus the product of $a(X)$ and $p_{q-1}^X(n_X)$ is the total number of ways to add a set to each of the $p_{q-1}^X(n_X)$ (q-1)-tuples to make it a q-tuple while keeping n_X unchanged. Similarly, for each nonempty $Z \subseteq Y$, we know $b(X,Z)$ means the number of sets in F that cover Z but avoid $X \setminus Z$, where $Z \subseteq Y \subseteq X$, and $p_{q-1}^X(n_X - 1^Z)$ means the number of (q-1)-tuples from F where for each $j \in X$ the number of the element j in the union of every (q-1)-tuple equals the updated n_j value in the set $(n_X - 1^Z)$; thus the product of $b(X,Z)$ and $p_{q-1}^X(n_X - 1^Z)$ is the total number of ways to add a set to each of the $p_{q-1}^X(n_X - 1^Z)$ (q-1)-tuples to make it a q-tuple while satisfying all the n_j values in the set n_X. Finally, the summation of all these products

香港大學圖書館 HKU Libraries

yields the number of q-tuples from F such that for each $j \in X$ the number of the element j in the union of every q-tuple equals n_j, which is $p_q^X(n_x)$.

So according to Equation (5.8), in order to get all $p_k^X(n_x)$, we need to calculate all $p_q^X(n_x)$ where $1 \le q < k$. Now before giving an algorithm for computing all $p_k^X(n_x)$, we need to first analyze the special case where the maximum integer t in the integral coverage requirement vector $T = (t_1,...,t_n)$ equals 1. In this case, set multi-cover becomes the set cover problem. Then as mentioned in Section 5.4.1, we only need to compute $\prod_{i=1}^{m} t_i = 1$ number of $p_k^X(n_x) = p_k^X(\underbrace{0,...,0}_{m})$ values. This means that the number of positive integers in the set $n_x = \{n_1, n_2,..., n_m\}$ is zero, i.e., the set Y in Equation (5.8) is an empty set. Accordingly, Equation (5.8) becomes $p_k^X(0,...,0) = b(X,\varnothing) \cdot p_{k-1}^X(0,...,0) = a(X) \cdot p_{k-1}^X(0,...,0)$. Since $p_1^X(0,...,0) = a(X)$, we can obtain $p_k^X(0,...,0) = (a(X))^k$. Finally from Equations (5.2) and (5.3), we obtain $c_k(F) = \sum_{X \subseteq N} (-1)^{|X|} \cdot (a(X))^k$, which is the same as the formula given in [101] for counting the number of k-tuples that satisfy the set cover requirement. As discussed in [101], based on whether we use exponential space or not (c.f. Section 5.4.2), $c_k(F)$ can be computed in $O^*(2^n)$ time and $O^*(2^n)$ space, or can be computed in $O^*(3^n)$ time and polynomial space.

For the following, we assume that the maximum integer t in the integral coverage requirement vector $T = (t_1,...,t_n)$ is greater than or equal to 2.

Algorithm 5.1 for computing all $p_k^X(n_x)$

Input: The value k where $t \le k \le tn$; the set $X = \{1,2,...,m\}$; the integral

香港大學圖書館 HKU Libraries

coverage requirement sub-vector for X , i.e., $T_x = (t_1, t_2, ..., t_m)$. Here T_x is a sub-vector of T , and we use $\min(T_x)$ and $\max(T_x)$ to denote the minimum and the maximum integers respectively in the sub-vector T_x .

Output: The values for all $p_k^X(n_x)$.

1: For all $X \subseteq N$, by using the fast zeta transform technique given in [101], we compute all $a(X)$ and store them in a look-up table.

2: Based on the first step, for all $Y \subseteq X$ and $X \subseteq N$, we compute all $b(X,Y)$ and store them in another look-up table.

3: For q=2 to k do:

4: By using Equation (5.8), we compute all $p_q^X(n_x)$ from $p_q^X(0,...,0)$ to $p_q^X(\min(q,t_1-1),...,\min(q,t_i-1),...,\min(q,t_m-1))$ (with lexicographic order) and we store all these $p_q^X(n_x)$ values in a look-up table. Here the function $\min(q,t_i-1)$ means choosing the minimum value between q and (t_i-1).

5: End For.

With the above Algorithm 5.1 for computing all $p_k^X(n_x)$, we can calculate $n_k(X)$ and then $c_k(F)$. Then we analyze in the following the time and space complexities for calculating $c_k(F)$.

5.5.5 Time and space complexities for calculating $c_k(F)$

Theorem 5.6: By using Algorithm 5.1 for computing all $p_k^X(n_x)$, $c_k(F)$ can be computed in $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space.

PROOF: The first step of Algorithm 5.1 uses $O^*(2^n)$ time and $O^*(2^n)$ space. For the second step, according to Lemma 5.5, computing all $b(X,Y)$ takes

香港大學圖書館 HKU Libraries

time $O(4^n)$. Obviously there are $\sum_{m=0}^{n} C_n^m 2^m = 3^n$ $b(X,Y)$, so storing all $b(X,Y)$ in a look-up table takes $O^*(3^n)$ space.

In the 'For' loop (step 3 to step 5), we calculate all $p_q^X(n_X)$ from $q=2$ to $q=k$ and store all these $p_q^X(n_X)$ values in a look-up table. So according to Equation (5.8), for each $p_q^X(n_X)$, since all the $b(X,Y)$ and $p_{q-1}^X(n_X)$ values have been stored, the time to compute $p_q^X(n_X)$ is $\sum_{j=0}^{r} C_r^j = 2^r$ where r is the number of positive integers in the set n_X. So in order to calculate the total time for calculating all $p_q^X(n_X)$, we just need to count how many $p_q^X(n_X)$ we need to compute.

Since we know the number of positive integers in the set n_X is r, for each q where $2 \le q \le k$, the number of $p_q^X(n_X)$ we need to compute equals $\prod_{i=1}^{r} \min(q, t_i - 1)$, i.e., those $p_q^X(n_X)$ from $p_q^X(\underbrace{1,...,1}_{r},\underbrace{0,...,0}_{m-r})$ to $p_q^X(\underbrace{\min(q,t_1-1),...,\min(q,t_r-1)}_{r},\underbrace{0,...,0}_{m-r})$.

So if $q \le \min(T_X) - 1 \le t - 1$, the number of $p_q^X(n_X)$ we need to compute is q^r, i.e., all $p_q^X(n_X)$ from $p_q^X(\underbrace{1,...,1}_{r},\underbrace{0,...,0}_{m-r})$ to $p_q^X(\underbrace{q,...,q}_{r},\underbrace{0,...,0}_{m-r})$. Similarly, if $t-1 < q \le k$, the number of $p_q^X(n_X)$ we need to compute equals $\prod_{i=1}^{r}(t_i - 1)$ which is less than $(t-1)^r$, i.e., all $p_q^X(n_X)$ from $p_q^X(\underbrace{1,...,1}_{r},\underbrace{0,...,0}_{m-r})$ to $p_q^X(\underbrace{t_1-1,...,t_r-1}_{r},\underbrace{0,...,0}_{m-r})$. Finally, if

香港大學圖書館 HKU Libraries

$\min(T_x) \le q \le \max(T_x) - 1 \le t - 1$, the number of $p_q^x(n_x)$ we need to compute is at most q^r.

From the above analyses, for a given n_x where the number of positive integers equals r and for all $2 \le q \le k$, the total number of $p_q^x(n_x)$ we have computed is at most:

$$\sum_{q=2}^{t-1} q^r + (k-t+1)\cdot(t-1)^r \tag{5.9}$$

As mentioned earlier in this proof, since the time for computing each $p_q^x(n_x)$ is 2^r, the total time for computing all these $p_q^x(n_x)$ is at most:

$$2^r \cdot (\sum_{q=2}^{t-1} q^r + (k-t+1)\cdot(t-1)^r)$$

Then for all n_x where r, the number of positive integers in each of them, varies from 0 to m, the total time for computing all $p_q^x(n_x)$ is at most:

$$\sum_{r=0}^{m} C_m^r (2^r \cdot (\sum_{q=2}^{t-1} q^r + (k-t+1)\cdot(t-1)^r)) = \sum_{q=2}^{t-1} (2q+1)^m + (k-t+1)\cdot(2t-1)^m$$

Now according to Equation (5.3) which is for computing $n_k(X)$, the total time for computing $n_k(X)$ is less than $\sum_{q=2}^{t-1}(2q+1)^m + (k-t+1)\cdot(2t-1)^m + t^m$, where the last term t^m accounts for the at most t^m number of additions of $p_k^x(n_x)$ to obtain $n_k(X)$.

Finally, according to Equation (5.2) which is for calculating $c_k(F)$, the time for computing $c_k(F)$ is at most:

$$\sum_{m=0}^{n} C_n^m (\sum_{q=2}^{t-1}(2q+1)^m + (k-t+1)\cdot(2t-1)^m + t^m)$$

$$= \sum_{q=2}^{t-2}(2q+2)^n + (k-t+2)\cdot(2t)^n + (t+1)^n$$

Now according to the following helping lemma, Lemma 5.7,

香港大學圖書館 HKU Libraries

$$\sum_{q=2}^{t-2}(2q+2)^n + (k-t+2)\cdot(2t)^n + (t+1)^n$$

$$= O((t-1)\cdot(2t-2)^n) + (k-t+2)\cdot(2t)^n + (t+1)^n = O^*((2t)^n)$$

Lemma 5.7: For any positive integer s, we have

$(s+1)\cdot(s/2)^n \le \sum_{i=1}^{s} i^n \le (s+1)\cdot s^n / 2$.

PROOF: First we define a function $f(x) = x^n + (s-x)^n$, where $0 \le x \le s$. By computing the second derivative of $f(x)$, we know $f(x)$ is a convex function. Thus it achieves the largest value at the boundaries of the x values, which are either $x = 0$ or $x = s$. By computing the first derivate of $f(x)$, we find that it achieves its smallest value at $x = s/2$. So we have $2^{1-n}s^n \le f(x) \le s^n$ for all $0 \le x \le s$. Then by replacing x with all its integer values from 0 to s, and summing these inequalities together, we obtain the result. This finishes the proof.

After proving the time complexity for calculating $c_k(F)$, we now turn to the space complexity. This is equivalent to finding out the total interim values we have stored in the look-up tables. We know already the total spaces for storing all $a(X)$ and $b(X,Y)$ values are $O^*(3^n)$, and now we only need to know the total number of $p_q^X(n_X)$ we have stored in the table. As given in Equation (5.9), for a given n_X and for all $2 \le q \le k$, the total number of $p_q^X(n_X)$ we have computed is at most $\sum_{q=2}^{t-1} q^r + (k-t+1)\cdot(t-1)^r$. Then for all n_X, the total number of $p_q^X(n_X)$ we have stored is at most:

$$\sum_{r=0}^{m} C_m^r \left(\sum_{q=2}^{t-1} q^r + (k-t+1)\cdot(t-1)^r\right) = \sum_{q=2}^{t-2}(q+1)^m + (k-t+2)\cdot t^m$$

香港大學圖書館 HKU Libraries

Finally, for all $X \subseteq N$, the total number of $p_q^x(n_x)$ we have stored is at most:

$$\sum_{m=0}^{n} C_n^m \left(\sum_{q=2}^{t-2}(q+1)^m + (k-t+2)\cdot t^m\right) = \sum_{q=2}^{t-2}(q+2)^n + (k-t+2)\cdot(t+1)^n$$

Again, according to Lemma 5.7, we have:

$$\sum_{q=2}^{t-2}(q+2)^n + (k-t+2)\cdot(t+1)^n = O(t^{n+1} + (k-t+2)\cdot(t+1)^n) = O^*((t+1)^n)$$

Since $t \geq 2$, all the time and spaces consumed in the first and the second step of Algorithm 5.1 can be subsumed in $O^*((2t)^n)$ and $O^*((t+1)^n)$, respectively. This finishes the proof of Theorem 5.6.

5.6 A Constructive Algorithm for the Set Multi-Cover Problem

Although we have computed the minimum number of sets that satisfy the coverage requirement, we have not really constructed these sets. In this section, we present an algorithm called **ESMC** for picking the minimum number of sets such that each element in the universe is covered by at least the required number of times as specified in the integral coverage requirement set. Before giving this constructive algorithm, we need to define two basic elements pair operations.

5.6.1 Two basic elements pair operations

We define two kinds of elements pair operations over a series of sets. One is called elements pair separation, which is to divide a set into two sets such that any pair of elements in the original set will fall into two different sets; the other is called elements pair coalition, which is to merge a pair of

香港大學圖書館 HKU Libraries

elements in the same set into a single element. Their formal definitions are given below.

Elements Pair Separation: For any set $s = \{a, b, x_1, ..., x_m\}$ in F which covers a pair of elements a and b, we replace the set s by separating the two elements into two different sets $s_a = \{a, x_1, ..., x_m\}$ and $s_b = \{b, x_1, ..., x_m\}$.

Elements Pair Coalition: For any set $s = \{a, b, x_1, ..., x_m\}$ in F which covers a pair of elements a and b, we replace the set s with the set $s_{ab} = \{ab, x_1, ..., x_m\}$ where the two elements a and b are merged into a new single element ab.

5.6.2 The constructive algorithm for the set multi-cover problem

We now give a constructive algorithm for finding the minimum number of sets in F that satisfy the integral coverage requirement vector T. This algorithm is based on finding the minimum k value such that the value of $c_k(F)$ is greater than zero.

ESMC: Exact Set Multi-Cover Algorithm

Input: A family F of subsets over the universe N; a coverage requirement vector T.

Output: The minimum number of sets from F to satisfy the requirement T.

1: Set $F_{bak} = F$.

2: Calculate the minimum value of k such that $c_k(F) > 0$.

3: Pick any element a in the universe N.

4: Find all the elements $\{x_1, ..., x_m\}$ in N that appear with a in some set in F.

5: Set $F_0 = F$.

6: **For** i=1 to m **do**:

7: $F = F_0$.

香港大學圖書館 HKU Libraries

8: For the pair of elements (a, x_i), we apply the Elements Pair Separation operation over the set F to generate a new set called F_i.

9: Calculate the value of $c_k(F_i)$.

10: End For

11: If all of the $c_k(F_i)$ values where $0 \le i \le m$ are greater than zero, we can deduce that there exists a set in the optimal cover which only covers the element a since otherwise there must exist some x_i whose separation with the element a can make $c_k(F_i) \le 0$. So we just pick this set in F which covers a and contains the least number of elements. We then decrement the value of k by 1 and update the coverage requirement vector T, i.e., for all elements x_i in the picked set we decrement each of the corresponding t_i values by 1. Also if any $t_i \le 0$ we remove the element i in the universe set N.

12: Else we pick any i such that $c_k(F_i) \le 0$. Then for the pair of elements $\{a, x_i\}$, we apply the Elements Pair Coalition operation over the set F. Note that the element a has become a new single element (ax_i).

13: Repeat step 4 to step 12 until we have picked a set from F.

14: Set $F = F_{bak}$ and we repeat step 3 to step 13 until $k = 0$.

5.6.3 Correctness Analysis

First, according to step 2, we know that the value of k we choose guarantees that we only use the minimum number of sets to satisfy the coverage requirement. Second, according to step 11, we know that, when we pick a set from F in each step, we can guarantee that the picked set must exist in some optimal legal (k, T) multi-cover sets. From this we also know that, when we pick this set, there must exist a legal $(k-1, T')$ multi-cover where T' is the updated coverage requirement vector after picking a subset

香港大學圖書館 HKU Libraries

from F. From the above analysis, we can conclude that we do pick the minimum number of sets from F that satisfies the coverage requirement vector T.

5.6.4 Time and Space Complexities Analysis

The time of the ESMC algorithm can be divided into two parts. The first part is due to step 2, which is to calculate the minimum k value for a legal (k,T) multi-cover. By using binary search, since $t \le k \le tn$, its time corresponds to $O(\log(tn))$ calculations of $c_k(F)$ (c.f. Section 5.3.2). The second part is due to steps 4 to 12 of the algorithm which is to pick a subset from F. We can easily see that it takes $O(n^2)$ calculations of $c_k(F)$. Since we need to pick k subsets, we need $O(kn^2)$ evaluations of $c_k(F)$ in total. So the overall time complexity is dependent on the time complexity for computing $c_k(F)$. Now according to Theorem 5.6, we have the following corollary.

COROLLARY 5.8: By using Algorithm 5.1 for computing all $p_k^x(n_x)$, the ESMC algorithm takes $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space where t is the maximum integer in the coverage requirement vector T.

5.7 An Illustrating Example

In this section, we give a very simple example to show how we calculate the value of $c_k(F)$ and how the ESMC algorithm works for the given example.

Suppose the universe $N = \{1,2,3\}$, the family of subsets over N is $F = \{\{1,2\},\{1,3\},\{2,3\}\}$ and the coverage requirement vector $T = (2,1,1)$. Now we first find the minimum k value to make a legal (k,T) multi-cover. This is

香港大學圖書館 HKU Libraries

equivalent to calculating the minimum k value such that $c_k(F) > 0$. Suppose we first test the case where $k = 2$.

According to Equation (5.2), we have $c_2(F) = \sum_{X \subseteq N} (-1)^{|X|} \cdot n_2(X)$. Now due to Equation (5.3), we have $n_2(X) = \sum_{\substack{0 \le n_j \le t_j - 1 \\ 1 \le j \le |X|}} p_2^X(n_1, ..., n_{|X|})$. Then based on these equations we have Table 5-2 which is to calculate $n_2(X)$ values for all $X \subseteq N$.

Table 5-2: Calculating $n_2(X)$ for all $X \subseteq N$

X	$n_2(X)$
$\varnothing$	$p_2^{\varnothing}(\varnothing)$
{1}	$p_2^{\{1\}}(0) + p_2^{\{1\}}(1)$
{2}	$p_2^{\{2\}}(0)$
{3}	$p_2^{\{3\}}(0)$
{1,2}	$p_2^{\{1,2\}}(0,0) + p_2^{\{1,2\}}(1,0)$
{1,3}	$p_2^{\{1,3\}}(0,0) + p_2^{\{1,3\}}(1,0)$
{2,3}	$p_2^{\{2,3\}}(0,0)$
{1,2,3}	$p_2^{\{1,2,3\}}(0,0,0) + p_2^{\{1,2,3\}}(1,0,0)$

The next step is to compute all the $p_2^X(n_1, ..., n_{|X|})$ values on the right side of Table 5-2. By combining Equation (5.8) which computes $p_q^X(n_1, ..., n_{|X|})$ and Equation (5.4) which computes $b(X, Y)$, we have Table 5-3.

Table 5-3: Calculating $p_2^X(n_1, ..., n_{|X|})$ for all $X \subseteq N$

X	$n_2(X)$
$\varnothing$	$p_2^{\varnothing}(\varnothing) = b(\varnothing, \varnothing) \cdot p_1^{\varnothing}(\varnothing) = a(\varnothing) \cdot a(\varnothing)$ =3*3=9.
{1}	(1): $p_2^{\{1\}}(0) = b(\{1\}, \varnothing) \cdot p_1^{\{1\}}(0)$

香港大學圖書館 HKU Libraries

$= a(\{1\}) \cdot b(\{1\},\varnothing) = a(\{1\}) \cdot a(\{1\}) = 1*1=1;$

(2): $p_2^{\{1\}}(1) = b(\{1\},\varnothing) \cdot p_1^{\{1\}}(1) + b(\{1\},\{1\}) \cdot p_1^{\{1\}}(0)$

$= a(\{1\}) \cdot b(\{1\},\{1\}) + b(\{1\},\{1\}) \cdot b(\{1\},\varnothing)$

$= a(\{1\}) \cdot [a(\varnothing) - a(\{1\})] + [a(\varnothing) - a(\{1\})] \cdot a(\{1\})$

$=1*(3-1)+(3-1)*1=4;$

(3): $p_2^{\{1\}}(0) + p_2^{\{1\}}(1) = 1+4=5.$

{2} $p_2^{\{2\}}(0) = b(\{2\},\varnothing) \cdot p_1^{\{2\}}(0)$

$= a(\{2\}) \cdot b(\{2\},\varnothing) = a(\{2\}) \cdot a(\{2\}) = 1*1=1.$

{3} $p_2^{\{3\}}(0) = b(\{3\},\varnothing) \cdot p_1^{\{3\}}(0)$

$= a(\{3\}) \cdot b(\{3\},\varnothing) = a(\{3\}) \cdot a(\{3\}) = 1*1=1.$

{1,2} (1): $p_2^{\{1,2\}}(0,0) = b(\{1,2\},\varnothing) \cdot p_1^{\{1,2\}}(0,0)$

$= a(\{1,2\}) \cdot b(\{1,2\},\varnothing)$

$= a(\{1,2\}) \cdot a(\{1,2\}) = 0*0=0;$

(2): $p_2^{\{1,2\}}(1,0) = b(\{1,2\},\varnothing) \cdot p_1^{\{1,2\}}(1,0) + b(\{1,2\},\{1\}) \cdot p_1^{\{1,2\}}(0,0)$

$= a(\{1,2\}) \cdot b(\{1,2\},\{1\}) + b(\{1,2\},\{1\}) \cdot b(\{1,2\},\varnothing)$

$= a(\{1,2\}) \cdot [a(\{2\}) - a(\{1\} \cup \{2\})] + [a(\{2\}) - a(\{1\} \cup \{2\})] \cdot a(\{1,2\})$

$=0*(1-0)+(1-0)*0=0;$

(3): $p_2^{\{1,2\}}(0,0) + p_2^{\{1,2\}}(1,0) = 0+0=0.$

{1,3} (1): $p_2^{\{1,3\}}(0,0) = a(\{1,3\}) \cdot a(\{1,3\}) = 0*0=0;$

(2): $p_2^{\{1,3\}}(1,0) = b(\{1,3\},\varnothing) \cdot p_1^{\{1,3\}}(1,0) + b(\{1,3\},\{1\}) \cdot p_1^{\{1,3\}}(0,0)$

$= a(\{1,3\}) \cdot b(\{1,3\},\{1\}) + b(\{1,3\},\{1\}) \cdot b(\{1,3\},\varnothing)$

$=0*1+1*0=0;$

(3): $p_2^{\{1,3\}}(0,0) + p_2^{\{1,3\}}(1,0) = 0+0=0.$

{2,3} $p_2^{\{2,3\}}(0,0) = a(\{2,3\}) \cdot a(\{2,3\}) = 0*0=0.$

{1,2,3} (1): $p_2^{\{1,2,3\}}(0,0,0) = a(\{1,2,3\}) \cdot a(\{1,2,3\}) = 0*0=0;$

(2): $p_2^{\{1,2,3\}}(1,0,0)$

$= b(\{1,2,3\},\varnothing) \cdot p_1^{\{1,2,3\}}(1,0,0) + b(\{1,2,3\},\{1\}) \cdot p_1^{\{1,2,3\}}(0,0,0)$

香港大學圖書館 HKU Libraries

$= a(\{1,2,3\}) \cdot b(\{1,2,3\},\{1\}) + b(\{1,2,3\},\{1\}) \cdot a(\{1,2,3\})$

=0*0+0*0=0;

(3): $p_2^{\{1,2,3\}}(0,0,0) + p_2^{\{1,2,3\}}(1,0,0)$ =0+0=0.

Having calculated all the $n_2(X)$ values which are shown on the right side of Table 5-3, we can obtain $c_2(F) = \sum_{X \subseteq N} (-1)^{|X|} \cdot n_2(X) = 9-5-1-1+0+0+0-0 = 2 > 0$, which means that there are two *2*-tuples that can satisfy the coverage requirement. Since the maximum integer in the coverage requirement vector *T* is 2, we know the minimum *k* value we need to pick is 2. Actually, by calculating the $c_1(F)$ value, which is $c_1(F) = \sum_{X \subseteq N} (-1)^{|X|} \cdot n_1(X) = 3-3-1-1+0+0+0-0 = -2 < 0$, we can also conclude that the minimum *k* value is 2 since picking one set from *F* does not meet the coverage requirement.

Now according to the ESMC algorithm, we briefly show in the following how to pick the two sets that can satisfy the coverage requirement *T*.

First, according to step 3, we pick the element 1 in the universe *N*. Then we can find the elements $\{x_1 = 2, x_2 = 3\}$ that can appear with 1 in some subsets in *F*. Now according to step 6 to step 10, we obtain $F_1 = \{\{1\},\{2\},\{1,3\},\{2,3\}\}$ and $F_2 = \{\{1,2\},\{1\},\{3\},\{2,3\}\}$. From this we can calculate $c_2(F_1) \le 0$ and $c_2(F_2) \le 0$. Then according to step 12, we choose to merge the elements pair $(1,2)$. Now since the new single element (12) does not appear with any other elements in the set *F*, we have $m = 0$. Then since $c_2(F_0) = c_2(F) = 2 > 0$, according to step 11, we just pick the first subset

香港大學圖書館 HKU Libraries

in F which is $\{1,2\}$. Similarly, we can pick the second subset in F which is $\{2,3\}$. This finishes the execution of the ESMC algorithm.

5.8 A Polynomial Time Polynomial Space Approximation Algorithm for the MLSAT Problem

In this section, we will give a polynomial time polynomial space algorithm for the MLSAT problem. First, we know the set multi-cover problem can be exactly solved in $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space where t is the maximum integer in the coverage requirement vector T. Note that since finding and storing all the link independent sets takes $O^*(2^n)$ time and $O^*(2^n)$ space, we know that exactly solving the MLSAT problem also takes $O^*((2t)^n)$ time and $O^*((t+1)^n)$ space. Thus if we partition all the links into $n/\log_{2t} n$ groups where each group contains $\log_{2t} n$ links, we can find the minimum number of timeslots to schedule all the links in each group with polynomial time and polynomial space. Then similar to Section 4.4.1, we know this algorithm can give a polynomial time polynomial space algorithm for the MLSAT problem with approximation ratio $O(n/\log n)$.

Compared with the approximation algorithm given in [137], our approximation ratio is independent of the links' lengths. Note that, the approximation ratio given in [137] could become $O(n)$ in the worst case.

香港大學圖書館 HKU Libraries

Chapter 6 A Nonlinear Power Assignment based Link Scheduling Algorithm for the MFSTT Problem in Wideband Networks

6.1 Ultra-Wideband Networks and Its SINR Model

The MFSTT problem in narrowband networks has been studied in [56,58], but it has not been examined in (ultra)-wideband networks. So in this chapter, we consider the MFSTT problem for ultra-wideband networks (UWB) which are drawing increasing attention in the wireless communications area due to their many promising features [116]. Specifically, since a UWB network is an inherent spread-spectrum network [114], the aggregate interferences caused by other simultaneous transmissions at the intended receiver can be reduced by a processing gain factor, thus making it very competitive in wireless communications (potentially improved throughput capacity while not sacrificing the energy-efficiency and the quality-of-service) [117]. And unlike the narrowband networks, where the interference range is larger than the transmission range, as will be shown later, the interference range of UWB networks around the receiver is much shorter than the transmission range, making more simultaneous transmissions at the receiver possible. For more information about UWB networks, please refer to [116].

Also recently, one of the main findings in UWB network research [115] is that the design of optimal MAC is independent of the choice of routing. Thus the use of ultra-wideband can re-introduce the notion of layer separation between these two layers just like the traditional wire line networks. This will

香港大學圖書館 HKU Libraries

make the resultant network more scalable and it will certainly be a good choice for the generic sensor networks. Furthermore, UWB is multi-path fading resistant, and as the following SINR model shows, it is more flexible in terms of adapting its parameters to meet different requirements (e.g., change in processing gain).

In our analyses, we also adopt the physical signal-to-interference-plus-noise ratio (SINR) model, which means that only when the received power is above the SINR ratio threshold can the message be successfully received. The SINR model in UWB networks was first given in [120], and it is different from the narrowband case in [56,58]. Specifically, the achieved signal-to-interference-plus-noise ratio at the receiver of link i can be represented as:

$$SINR_i = \frac{P_i / d(x_i, x_j)^{\alpha}}{R_i[\eta + T_f \sigma^2 \sum_{k=1, k \neq i} P_k / d(x_k, x_j)^{\alpha}]} \geq \beta$$

where P_i denotes the average transmission power of link i's transmitter x_i; R_i denotes link i's data rate, and $R_i = 1/(N_s N_h T_c)$; N_s denotes the number of pulses per symbol; N_h denotes the number of time slots per Pulse Repetition Interval (*PRI*); T_c denotes the pulse duration; T_f is the PRI, and $T_f = N_h T_c$; σ^2 is a parameter depending on the shape of the monocycle; η is the background noise plus interference from other non-UWB systems; $d(x_i, x_j)$ denotes the Euclidean distance between transmitter x_i and x_j; α is the path loss exponent and β is the SINR threshold.

If we set $N = \eta/(T_f \sigma^2)$ and $m = 1/(R_i T_f \sigma^2)$, the above SINR model can be transformed to a form similar to the spread-spectrum SINR model given in Chapter 1 (cf. Inequality (1.1)):

香港大學圖書館 HKU Libraries

$$SINR = \frac{P_i / d(x_i, x_j)^{\alpha}}{N + \sum_{k=1, k \neq i} P_k / d(x_k, x_j)^{\alpha}} \geq \frac{\beta}{m} \tag{6.1}$$

Here m is the *processing gain* of the UWB network. If m=1, this becomes a traditional narrowband SINR model, as used in [56,58]. The processing gain in (ultra)-wideband networks can be regarded as the signal's ability to combat the aggregate interferences. So in this chapter, we will see how this processing gain can help to reduce the scheduling length of the MFSTT problem. In addition, since all the previously used nonlinear power assignment based scheduling algorithms have not taken care of their total power consumption, we will also pay attention to the energy consumption analysis of the nonlinear power assignment.

The rest of this chapter is as follows. In Section 6.2, for both narrowband and wideband networks, we explore different power assignments and their impacts on pair-wise interference models which play a very important role in the design of wireless protocols and wireless network capacity analyses. In Section 6.3, we continue to compare the narrowband and wideband networks in terms of power limitations in improving the spatial reuse. In Section 6.4, in the context of wideband networks, we will give a nonlinear power assignment based link scheduling algorithm for the MFSTT problem, with the guarantee that all the simultaneous transmissions can be successfully scheduled based on the SINR model. Specifically, our algorithm proves that the scheduling length for the MFSTT problem for wideband networks is $O(\log(n/m) \cdot \log n)$. This result represents an improvement over that for the narrowband networks.

香港大學圖書館 HKU Libraries

In the same section, we also analyze the total power consumption of our nonlinear power assignment based scheduling algorithm. In particular, we show that the poly-logarithmic scheduling length was achieved at the expense of the exponential total power consumption. And in wideband networks, the upper bound of the total power consumption can be reduced by a processing gain factor. Section 6.5 concludes the chapter and discusses some future tasks that could make our algorithm practical.

6.2 Protocol Interference Models in Narrowband and Wideband Networks

In this section, we focus on the impact of the power assignments on the pair-wise interference models, which was often neglected in wireless scheduling algorithm design. Specifically, we will show how the protocol interference models introduced in Chapter 1 for narrowband networks behave in wideband networks. Through this comparison, we will find that there is more room for wideband networks to take advantage of power control to reduce the scheduling length. We first consider narrowband networks.

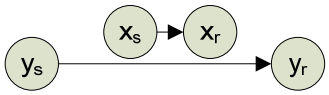
6.2.1 Protocol interference models in narrowband networks

According to inequality (1.4), in order to ensure a successful transmission (x_s, x_r), the protocol interference model with constant power assignment in narrowband networks is:

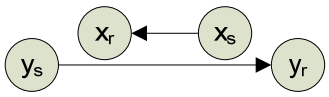
$$d(y_s, x_r) > \beta^{1/\alpha} \cdot d(x_s, x_r)$$

香港大學圖書館 HKU Libraries

Since in narrowband networks, usually the threshold β >1 and consequently the range $\beta^{1/\alpha} \cdot d(x_s, x_r)$ is greater than the sender's transmission range $d(x_s, x_r)$. Thus to ensure a successful transmission, a disc of radius at least $\beta^{1/\alpha} \cdot d(x_s, x_r)$ around each successful receiver x_r must not contain other transmitters. So we denote $\beta^{1/\alpha} \cdot d(x_s, x_r)$ as the interference range (or exclusion region) around each receiver x_r. For example, in Figure 6-1(a), assuming constant power assignment, since $d(x_s, y_r) < d(y_s, y_r)$, transmission (y_s, y_r) is not successful; whereas, since $d(y_s, x_r) > d(x_s, x_r)$, transmission (x_s, x_r) is successful. With this we can distinguish the other graph-based interference models from the protocol interference model which was considered the same in [10]. Notice that the protocol interference model originates from the physical SINR model, and so it can reflect the physical reality including the "capture effect" (cf. Section 1.1.1), while all the other graph-based interference models cannot reflect this reality. For example, since node x_r is in the transmission range of y_s, it suffers from the secondary interference problem, so transmission (x_s, x_r) is not successful.

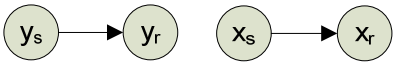


(a) $d(x_s, x_r) = 1, d(y_s, y_r) = 4, d(x_s, y_r) = 2, d(y_s, x_r) = 3$

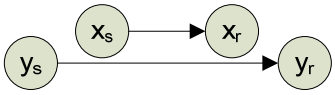


(b) $d(x_s, x_r) = 2, d(y_s, y_r) = 4, d(x_s, y_r) = 1, d(y_s, x_r) = 1$

香港大學圖書館 HKU Libraries



(c) $d(x_s, x_r) = 1, d(y_s, y_r) = 1, d(x_s, y_r) = 1, d(y_s, x_r) = 3$



(d) $d(x_s, x_r) = 2, d(y_s, y_r) = 4, d(x_s, y_r) = 3, d(y_s, x_r) = 3$

Figure 6-1: Pair-wise transmissions examples

Now according to inequality (1.5), in order to ensure a successful transmission (x_s, x_r), the protocol interference model with linear power assignment in narrowband networks is:

$$d(y_s, x_r) > \beta^{1/\alpha} \cdot d(y_s, y_r)$$

This protocol model was used in [64]. But compared with the first protocol interference model, it has attracted much less attention mostly because many capacity analysis papers assume the constant power assignment. Note that here the interference range of receiver x_r has been changed from $\beta^{1/\alpha} \cdot d(x_s, x_r)$ to $\beta^{1/\alpha} \cdot d(y_s, y_r)$. For example, in Figure 6-1(a), assuming linear power assignment, since $d(y_s, x_r) < d(y_s, y_r)$, transmission (x_s, x_r) is not successful. And since $d(x_s, y_r) > d(x_s, x_r)$, transmission (y_s, y_r) is successful.

6.2.2 Protocol interference models in wideband networks

Now we turn to UWB networks. According to inequality (6.1), in order to ensure a successful transmission (x_s, x_r), the following inequality must hold.

香港大學圖書館 HKU Libraries

$$\frac{P_x / d(x_s, x_r)^\alpha}{N + P_y / d(y_s, x_r)^\alpha} \geq \frac{\beta}{m} \quad \Rightarrow \quad \frac{d(y_s, x_r)}{d(x_s, x_r)} > (\frac{\beta}{m})^{\frac{1}{\alpha}} \cdot (\frac{P_y}{P_x})^{\frac{1}{\alpha}} \tag{6.2}$$

We first consider the protocol interference model with constant power assignment in UWB networks. With the constant power assignment, by inequality (6.2), we have

$$d(y_s, x_r) > (\beta / m)^{1/\alpha} \cdot d(x_s, x_r) \tag{6.3}$$

The interference range $\beta^{1/\alpha} \cdot d(x_s, x_r)$ around the receiver x_r is replaced with $(\beta / m)^{1/\alpha} \cdot d(x_s, x_r)$. Hence the interference range becomes smaller than the transmission range.

For example, in Figure 6-1(a), if α =4, β =2, m=100, since

$d(x_s, y_r)$ =2> $(\beta / m)^{1/\alpha} \cdot d(y_s, y_r)$ ≒ 1.5, the previously unsuccessful transmission (y_s, y_r) with constant power assignment in narrowband networks becomes successful in UWB networks. As a result, the two transmissions can be scheduled in parallel.

Second, we take a look at the protocol interference model with linear power assignment in UWB networks. Also by inequality (6.2), we have

$$\frac{d(y_s, x_r)}{d(x_s, x_r)} > (\frac{\beta}{m})^{\frac{1}{\alpha}} \cdot \frac{d(y_s, y_r)}{d(x_s, x_r)} \Rightarrow d(y_s, x_r) > (\tfrac{\beta}{m})^{\frac{1}{\alpha}} \cdot d(y_s, y_r) \tag{6.4}$$

The interference range around receiver x_r is changed from $(\beta / m)^{1/\alpha} \cdot d(x_s, x_r)$ to $(\beta / m)^{1/\alpha} \cdot d(y_s, y_r)$. For example, in Figure 6-1(a), if α =4, β =2, m=100, since $d(y_s, x_r)$ =3> $(\beta / m)^{1/\alpha} \cdot d(y_s, y_r)$ ≒ 1.5, the previously unsuccessful transmission (x_s, x_r) with linear power assignment in

香港大學圖書館 HKU Libraries

narrowband networks becomes successful in UWB networks. So the two transmissions can be simultaneously scheduled.

From the above analyses, on one hand, due to the large processing gain m when using the constant or linear power assignment, many unsuccessful simultaneous transmissions in narrowband networks become successful in UWB networks, thus leading to increased spatial reuse in UWB networks. On the other hand, as the examples in [65] have shown, even in narrowband networks, the unsuccessful simultaneous transmissions can also become successful with a proper arbitrary power assignment. For example, for Figure 6-1(a), if α =4, β =2, N=1, and P_x =80, P_y =3150, the two transmissions can be successfully scheduled in parallel. And for Figure 6-1(c), if α =3, β =4, N=1, and P_x =14, P_y =64, the two transmissions can also take place simultaneously.

6.3 Limitations of Power Control in Narrowband and Wideband Networks

From the last section, it is shown that we can benefit a lot from power control to reduce the scheduling length in wireless networks. In this section, we will show that power control, however, has its limitations in improving the network throughput when some conditions are met. For example, for narrowband networks, according to Theorem 2.6, we know that for any two transmissions (x_s, x_r) and (y_s, y_r), if $d(x_s, y_r) \cdot d(y_s, x_r) \leq \beta^{2/\alpha} \cdot d(x_s, x_r) \cdot d(y_s, y_r)$, then there exists no feasible power assignment for simultaneous transmissions; otherwise, there always exists a feasible power assignment to have a simultaneous schedule. For example, in Figure 6-1(d),

香港大學圖書館 HKU Libraries

if α =4, β =2, and N=1, there will be no feasible power assignment to simultaneously schedule transmission (x_s , x_r) and (y_s , y_r). The same is true of Figure 6-1(b).

We now give another theorem to show that, although there is much more room for UWB networks to reduce the scheduling length through power control, the power control strategy also has its limitations in wideband networks.

THEOREM 6.1. In UWB (or any spread-spectrum) networks, for any two transmissions (x_s , x_r) and (y_s , y_r), if

$d(x_s, y_r) \cdot d(y_s, x_r) > (\beta / m)^{2/\alpha} \cdot d(x_s, x_r) \cdot d(y_s, y_r)$, there always exists a power assignment to schedule these transmissions in parallel; no feasible power assignments for simultaneous schedule, otherwise.

PROOF. Similar to the proof of Theorem 2.6, if the two transmissions can be successfully scheduled, the following two inequalities must follow:

$$\frac{P_x / d(x_s, x_r)^\alpha}{N + P_y / d(y_s, x_r)^\alpha} \geq \frac{\beta}{m} \qquad \frac{P_y / d(y_s, y_r)^\alpha}{N + P_x / d(x_s, y_r)^\alpha} \geq \frac{\beta}{m}$$

From these inequalities, we have

$$\frac{\beta}{m} \cdot P_y \frac{d(x_s, x_r)^\alpha}{d(y_s, x_r)^\alpha} < P_x < \frac{m}{\beta} \cdot P_y \cdot \frac{d(x_s, y_r)^\alpha}{d(y_s, y_r)^\alpha}$$

Therefore, if $\frac{\beta}{m} \cdot \frac{d(x_s, x_r)^\alpha}{d(y_s, x_r)^\alpha} < \frac{m}{\beta} \cdot \frac{d(x_s, y_r)^\alpha}{d(y_s, y_r)^\alpha}$, there always exists a power assignment to simultaneously schedule these two transmissions; otherwise, there is no valid power assignment to give a parallel schedule. This ends the proof.

For example, in Figure 6-1(d), if α =4, β =2, N=1, and m=10, $P_x = P_y$ =1000, the two transmissions can be simultaneously scheduled.

香港大學圖書館 HKU Libraries

Therefore, given any two transmissions in narrowband networks where power control cannot guarantee a simultaneous schedule, they can be scheduled in parallel in UWB networks as long as $d(x_s, y_r) \cdot d(y_s, x_r) > (\beta / m)^{2/\alpha} \cdot d(x_s, x_r) \cdot d(y_s, y_r)$. Given this result, we will discuss how these benefits can help to reduce the scheduling length for the MFSTT problem in the context of (Ultra)-Wideband networks in the next section.

6.4 The NPAW Scheduling Algorithm for the MFSTT Problem in Wideband Networks

We consider an arbitrarily distributed network with nodes $X=\{x_0, x_1, ..., x_{n-1}\}$ in the Euclidean plane, and one of them is a sink node. Here by a sink node, we mean there are no outgoing edges (links) from this node. For any links $f_{ij}=(x_i, x_j)$, $\ell(f_{ij})=d(x_i, x_j)$ denotes the distance between node x_i and node x_j. Now before going into the details of the scheduling algorithm, it is important to distinguish between *link length class* and *link length class set* which are used in our algorithm. A *link length class* is a set of transmission links such that the lengths of these links differ by at most a factor of 2 (line 8 of the main algorithm). A number of link length classes form a *link length class set*. The three kinds of link length class set *L*, *S* and *I* used in our algorithm, and their relationships, are described in Figure 6-2.

香港大學圖書館 HKU Libraries

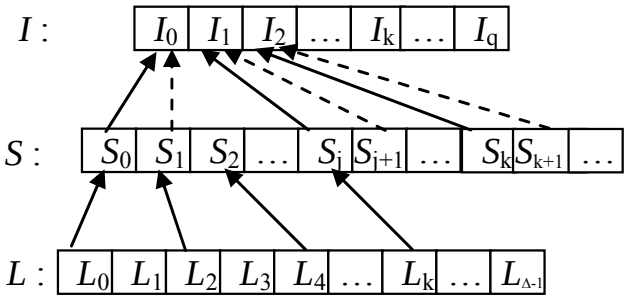


Figure 6-2: Three kinds of link length class set and their relationships

In Figure 6-2, L_k, S_k and I_k denote the respective length classes in each set. L is renamed to S because the empty length classes (containing no transmission links) in L were deleted (line 9 of the main algorithm). For example, the length classes L_1 and L_3 were deleted. S is renamed to I because in each round, the scheduling algorithm only selects the length classes in S with a certain length class separation. The separation value is $\log(4\beta n)$ in [58] but we use $\log(3n\beta/m)$ in our algorithm (line 11 of the main algorithm). The solid arrows from S to I mean we select the length classes $S_0 S_j S_k \ldots$ in the first round, while the dashed arrows mean we select the length classes $S_1 S_{j+1} S_{k+1} \ldots$ in the second round (the details are in Table 6-1 and Table 6-2). Note that only links in L_k have the property $2^k \le \ell(f_{ij}) < 2^{k+1}$, but not those in S_k or I_k (because 2^{k+1} upper bound would not hold for them).

Our scheduling algorithm also uses a nonlinear power assignment. For convenience, we refer to the scheduling algorithm in [55,56,58,59] as "NPAN" (nonlinear power assignment for narrowband networks), and our algorithm "NPAW" (nonlinear power assignment for (ultra)-wideband networks). Note that only the works in [56,58] directly investigate the MFSTT problem. Our main algorithm is different from [56] in the sense that we start the scheduling process after the tree topology has been constructed. Thus, compared with

香港大學圖書館 HKU Libraries

the result in [56], the scheduling length upper bound can be reduced by a $O(\log n)$ factor. In addition, we need to point out that, as shown in step 1 to step 6 in the main algorithm, if the remaining single node in the established tree topology is not the designated sink node, we just need to add an outgoing link from this single node to the pre-determined sink and then to remove the outgoing edge (link) from the sink node.

The challenging part of the algorithm is how to schedule all the links both successfully and efficiently. Just as Figure 6-2 has demonstrated, we first partition all the links into length classes of L which is then renamed to S (lines 8 and 9). Then we use the subroutine Schedule() to schedule the links in length classes $S_{h\cdot\log(3n\beta/m)+k}$ in the k^{th} round (lines 10, 11 and Table 6-2). The trick of this algorithm lies in two aspects: one is the nonlinear power assignment scheme (line 14 of the subroutine). This power assignment uses a power scaling factor τ which depends on the position of the scheduling links in link length class set I (lines 1 and 2 of the subroutine and Figure 6-2). Because short links have a high τ value and long links have a low τ value, this power assignment can increase the power of the short links relative to the long ones so that it makes simultaneous transmissions of very different lengths possible. Furthermore, because this power assignment takes the parameter n (total number of the nodes) into account, it can bound the aggregate interferences through the properly designed protocol interference model (line 10 of the subroutine). But as discussed in Section 6.2, traditional pair-wise protocol interference models cannot guarantee the successful transmission due to the aggregate interference effect.

香港大學圖書館 HKU Libraries

The second part of the trick is the selection of the simultaneous transmitting links in length class set I (Figure 6-2). With the proper length class separation, for each link i, the algorithm can bound the total number of blocking links for this link which is $O(\log n)$ (line 11 of the main algorithm and line 10 of the subroutine), thus guaranteeing that after at most ($O(\log n)$) timeslots, all the links can be successfully scheduled. Therefore the poly-logarithmic scheduling length can be arrived at. Here for the blocking links of link i, we mean the links which can not be simultaneously scheduled with link i.

6.4.1 Correctness analysis

Compared with narrowband networks, there are more links that can be scheduled in each timeslot in wideband networks (link 10 in the Subroutine). In this case, guaranteeing the successful simultaneous transmissions in the same timeslot is of fundamental importance.

Main Algorithm: A Nonlinear Power Assignment based Link Scheduling Algorithm for (Ultra)-Wideband Networks (NPAW)

Input: An arbitrarily distributed set of nodes X

Output: A data gathering tree with the number of timeslots t to schedule all the links in this tree under the SINR model

1: $F = \varnothing$

2: While $|X|>1$ do

3: For each $x_i \in X$ find its closest neighbor x_j such that

$F := F \cup f_{ij}$; {f_{ij} is a directed edge from x_i to x_j.}

4: If F contains bi-directional edges then remove one edge of them; {To make F a directed nearest neighbor forest}

5: Delete all the nodes from node set X except the sink node in each tree of

香港大學圖書館 HKU Libraries

the directed nearest neighbor forest F ;

6: End While {Step 2 to step 6 is to construct a tree topology}

7: Define a constant υ :=4N and a variable μ which is a function of the processing gain m such that μ :=2+ ε +4$\cdot$ $(72)^{\frac{1}{\alpha}} \cdot \sqrt[\alpha]{\frac{\beta}{m}\frac{(\alpha-1)}{(\alpha-2)}}$; α >2; t:=0;

{N is the background noise from the Inequality (6.1) and ε is a small positive parameter.}

8: Partition all the transmission links in F into length class set $L = \{L_0, L_1, ..., L_{\Delta-1}\}$, such that L_k contains all links f_{ij} of length $2^k \le \ell(f_{ij}) < 2^{k+1}$;

{$\Delta = \lceil \log(l_{\max}) \rceil$, and $l_{\max}$ means the maximum link length in F .}

9: Delete all empty length classes L_k in F and rename L to $S = \{S_0, S_1, ..., S_k, ...\}$ such that S_k is the k^{th} smallest non-empty length-class in S ;

10: For k=0 to $\log(3n\beta / m) - 1$ do

11: Schedule all the links

$$f_{ij} \in \bigcup_{h=0}^{n/\log(3n\beta/m)-1} S_{h\cdot\log(3n\beta/m)+k}$$

using subroutine Schedule();

12: End For

13: Return t

Subroutine Schedule():

1: Let F_r be the set of links to be scheduled, rename these link length classes in S to $l = \{l_0, l_1, ..., l_q\}$ with at most q+1 length classes where q=$\lceil n / \log(3n\beta / m) - 1 \rceil$. l_k is the k^{th} smallest length-class in l; {line 11 of the main algorithm}

2: for each $f_{uv} \in l_k$ do $\tau(x_u) := q - k + 1$;

{Links within the smallest length class l_0 have the highest τ value $\lceil n / \log(3n\beta / m) \rceil$, and links within the largest length class l_q have the lowest τ value 1.}

3: while $F_r \neq \varnothing$ do

香港大學圖書館 HKU Libraries

4: $E_t := \varnothing$;

5: Consider all $f_{ij} \in F_r$ in an increasing order of their lengths

6: Boolean:=true;

7: If $E_t \neq \varnothing$

8: For each link $f_{kl} \in E_t$

9: $\delta_{ik} := \tau(x_i) - \tau(x_k)$;

10: if δ_{ik}=0 and d(x_i , x_l) $\leq \mu \cdot \ell(f_{ij}^*)$

or if $\delta_{ik} \neq 0$ and d(x_i , x_l) $\leq (3n\beta / m)^{(\delta_{ik}+1)/\alpha} \cdot \ell(f_{ij}^*)$

Boolean:=false;

11: End For

12: End If

13: If Boolean==True then $E_t := E_t \cup \{f_{ij}\}$; $F_r = F_r \setminus \{f_{ij}\}$

14: Schedule all $f_{ij} \in E_t$ in timeslot t with the transmission power

$P(x_i) := \upsilon(3n\beta / m)^{\tau(x_i)} \cdot \ell(f_{ij}^*)^\alpha$

15: t=t+1;

16: End While

LEMMA 6.2: Consider a scheduled link f_x with intended sender x_s and receiver x_r . Let $I_r(y_i)$ be the interference caused at x_r by simultaneously transmitting nodes y_i for which $\tau(y_i) < \tau(x_s)$. It holds that $I_r(y_i) \leq \upsilon(3n\beta / m)^{\tau(x_s)-1}$.

PROOF: In our main algorithm, because every node y_i transmits messages to its nearest neighbor, we have $d(y_i , x_r) \geq \ell(f_y)$. Hence the interference at x_r caused by y_i is at most

$$I_r(y_i) = \frac{P_i}{d(y_i, x_r)^\alpha} \leq \frac{\upsilon(3n\beta / m)^{\tau(y_i)} \cdot \ell(f_y)^\alpha}{\ell(f_y)^\alpha} = \upsilon(\tfrac{3n\beta}{m})^{\tau(y_i)} \leq \upsilon(\tfrac{3n\beta}{m})^{\tau(x_s)-1}.$$

香港大學圖書館 HKU Libraries

LEMMA 6.3: Consider a scheduled link f_x with intended sender x_s and receiver x_r. Let $I_r(y_i)$ be the interference caused at x_r by simultaneously transmitting nodes y_i for which $\tau(y_i) > \tau(x_s)$. It holds that $I_r(y_i) \le \upsilon(3n\beta / m)^{\tau(x_s)-1}$.

PROOF: Assume for contradiction that there exists a node y_i with $\tau(y_i) > \tau(x_s)$ and $I_r(y_i) > \upsilon(3n\beta / m)^{\tau(x_s)-1}$. Then

$$I_r(y_i) = \frac{P_i}{d(y_i, x_r)^\alpha} \le \frac{\upsilon(3n\beta / m)^{\tau(y_i)} \cdot \ell(f_y)^\alpha}{d(y_i, x_r)^\alpha} > \upsilon(\tfrac{3n\beta}{m})^{\tau(x_s)-1}$$

From this, we have $d(y_i, x_r) < (3n\beta / m)^{(\delta_{is}+1)/\alpha} \cdot \ell(f_y)$.

However, this contradicts the definition of our algorithm. In line 10 of the subroutine, if node y_i has been scheduled (because it has short link length, line 5 of the subroutine), from the above inequality, node x_s should not have been scheduled, which establishes the contradiction. Therefore, $I_r(y_i) \le \upsilon(3n\beta / m)^{\tau(x_s)-1}$ holds.

LEMMA 6.4: Consider a scheduled link f_x with intended sender x_s and receiver x_r. Let I_r^0 be the total interferences caused at x_r by simultaneously transmitting nodes y_i for which $\tau(y_i) = \tau(x_s)$. The following holds:

$I_r^0 \le (\upsilon / 3) \cdot (\beta / m)^{\tau(x_s)-1} \cdot (3n)^{\tau(x_s)}$.

PROOF: The proof of this lemma is similar to that of Lemma 4.4 in [58]. The main idea is that because the lengths of the links in the same length class differ by at most a factor of 2, according to a simple geometric area argument, the blocking links must be bounded by a certain number. The difference is that we change the ring width from

香港大學圖書館 HKU Libraries

$\frac{1}{2}(\mu-3)\cdot\ell(f_x)$ to $\frac{1}{2}(\mu-2-\varepsilon)\cdot\ell(f_x)$. And more importantly, the μ value is greatly reduced due to the introduction of the processing gain m in the denominator. Thus the blocking links in the same length class are greatly reduced. Plugging in the value of μ in line 7 of the main algorithm, the results follow.

THEOREM 6.5: For an arbitrary timeslot *t*, all scheduled transmissions E_t in *t* are received successfully by the intended receivers, and thus the computed schedule is correct.

PROOF: Consider a scheduled link f_x with intended sender x_s and receiver x_r. The aggregate interferences at this receiver x_r can be calculated through Lemmas 6.2, 6.3 and 6.4.

By Lemmas 6.2 and 6.3, we know that for all y_i with $\tau(y_i)>\tau(x_s)$ and $\tau(y_i)<\tau(x_s)$, the interference $I_r(y_i)$ is bounded by $\upsilon(3n\beta/m)^{\tau(x_s)-1}$. Hence, because there are at most n nodes in these sets, it holds that

$$\sum_{y_i:\tau(x_s)\neq\tau(y_i)} I_r(y_i) \le n\cdot\upsilon(\tfrac{3n\beta}{m})^{\tau(x_s)-1} = \frac{\upsilon}{3}\cdot(\frac{\beta}{m})^{\tau(x_s)-1}\cdot(3n)^{\tau(x_s)}$$

Therefore the aggregate interference at x_r is

$$I_r = (\upsilon/3)\cdot(\beta/m)^{\tau(x_s)-1}\cdot(3n)^{\tau(x_s)} + (\upsilon/3)\cdot(\beta/m)^{\tau(x_s)-1}\cdot(3n)^{\tau(x_s)}$$

$$= 2\cdot(\upsilon/3)\cdot(\beta/m)^{\tau(x_s)-1}\cdot(3n)^{\tau(x_s)}$$

And *SINR* at x_r is

$$SINR = \frac{\upsilon\cdot(3n\beta/m)^{\tau(x_s)}\cdot\ell(f_x)^{\alpha}/\ell(f_x)^{\alpha}}{N+2\cdot(\upsilon/3)\cdot(\beta/m)^{\tau(x_s)-1}\cdot(3n)^{\tau(x_s)}}$$

Since υ:=4*N* (line 7 of the main algorithm)

香港大學圖書館 HKU Libraries

$$SINR = \frac{\upsilon \cdot (3n\beta / m)^{\tau(x_s)}}{N + 2 \cdot (\upsilon / 3) \cdot (\beta / m)^{\tau(x_s)-1} \cdot (3n)^{\tau(x_s)}} \geq \frac{\beta}{m}$$

From this, we conclude that the computed schedule is correct.

6.4.2 Efficiency analysis

COROLLARY 6.6: In each timeslot, the blocking links in the same length class in the NPAW algorithm are strictly fewer than the deleted links in the NPAN-INFOCOM06 algorithm in [58].

PROOF: This conclusion is from the proof of Lemma 6.4.

LEMMA 6.7: In each timeslot, the blocking links in different length classes in the NPAW algorithm are fewer than or at most equal to the deleted links in the NPAN-INFOCOM06 algorithm in [58].

PROOF: From line 10 of the subroutine, on one hand, if the difference of the power scaling factors between different length classes is the same, because we have introduced the processing gain m as the denominator in the base, the blocking links must be fewer than its counterpart in NPAN-INFOCOM06. On the other hand, since $(3n\beta / m)^{(\delta_{ik}+1)/\alpha} \leq (3n\beta / m)^{(n/\log(3n\beta/m)-1+1)/\alpha} = 2^{n/\alpha}$, and since $(4n\beta)^{(\delta_{ik}+1)/\alpha} \leq (4n\beta)^{(n/\log(4\beta n)-1+1)/\alpha} = 2^{n/\alpha}$, the deleted links must be at most equal to its counterpart in NPAN-INFOCOM06.

THEOREM 6.8: The scheduling length for the MFSTT problem in (Ultra)-Wideband networks is $O(\log(n/m) \cdot \log n)$.

PROOF: First of all, according to Corollary 6.6 and Lemma 6.7, for each link in a scheduling round (each kth iteration in the for loop in line 10 of the main algorithm), the total number of blocking links must not

香港大學圖書館 HKU Libraries

exceed $O(\log n)$ which is the result of the NPAN-INFOCOM06 algorithm. Hence, after at most $O(\log n)$ timeslots, all the transmission links that remain to be scheduled in the k^{th} scheduling round can be successfully scheduled. And since there are at most $\log(3n\beta / m)$ scheduling rounds, the total scheduling length of this algorithm is: $O(\log n) \cdot \log(3n\beta / m) \in O(\log(n / m) \cdot \log n)$.

6.4.3 Total power consumption analysis

In this section we will analyze the total power consumption for the NPAN-INFOCOM06 algorithm and our NPAW algorithm. First, we will give the analysis for the NPAN-INFOCOM06 algorithm [58].

THEOREM 6.9: For the strong connectivity scheduling algorithm for narrowband networks, i.e., NPAN-INFOCOM06, the lower bound of the total power consumption is $\Omega(n \cdot 2^n)$; and the upper bound of the total power consumption is $O(n^2 \cdot 2^{n\alpha})$, where n is the number of the nodes.

PROOF: In the NPAN-INFOCOM06 algorithm, only links in link length class $S_{h \cdot \log(4\beta n)+k}$ can be simultaneously scheduled in the k^{th} scheduling round (k is from 0 to $\log(4\beta n) - 1$, represented by the columns of Table 6-1). And h is from 0 to $n / \log(4\beta n) - 1$ (represented by the rows of Table 6-1). In particular, let's consider the link length classes S_k and $S_{n-\log(4\beta n)+k}$, which are the shortest length class and the longest length class in the k^{th} scheduling round, respectively.

香港大學圖書館 HKU Libraries

According to Figure 6-2, suppose the length class S_k is mapped from L_u ,we have $u \geq k$; And suppose the length class $S_{n-\log(4\beta n)+k}$ is mapped from $L_{n-\log(4\beta n)+v}$,we have $v \geq u \geq k$. According to the power scaling factor τ of their algorithm, length class S_k has the highest τ value $n/\log(4\beta n)$; and length class $S_{n-\log(4\beta n)+k}$ has the lowest τ value, of 1. So according to the nonlinear power assignment scheme in the algorithm, the power $P(S_k)$ assigned to the links in S_k has the property

$$\upsilon(4\beta n)^{n/\log(4\beta n)} \cdot 2^{u\alpha} \leq P(S_k) < \upsilon(4\beta n)^{n/\log(4\beta n)} \cdot 2^{(u+1)\alpha} \quad \Rightarrow$$
$$\upsilon \cdot 2^n \cdot 2^{u\alpha} \leq P(S_k) < \upsilon \cdot 2^n \cdot 2^{(u+1)\alpha}$$

The power $P(S_{n-\log(4\beta n)+k})$ assigned to links in $S_{n-\log(4\beta n)+k}$ has the property

$$P(S_{n-\log(4\beta n)+k}) \geq \upsilon \cdot (4\beta n)^1 \cdot (2^{n-\log(4\beta n)+v})^\alpha \quad \text{and}$$
$$P(S_{n-\log(4\beta n)+k}) < \upsilon \cdot (4\beta n)^1 \cdot (2^{n-\log(4\beta n)+v+1})^\alpha \quad \Rightarrow$$
$$\upsilon \cdot 2^{n\alpha} \cdot 2^{v\alpha} / (4\beta n)^{\alpha-1} \leq P(S_{n-\log(4\beta n)+k}) < \upsilon \cdot 2^{n\alpha} \cdot 2^{(v+1)\alpha} / (4\beta n)^{\alpha-1}$$

Because $0 \leq k \leq u \leq v \leq \log(4\beta n) - 1$, we have

$$\upsilon \cdot 2^n \cdot 2^0 \leq \upsilon \cdot 2^n \cdot 2^{u\alpha} \leq P(S_k) < \upsilon \cdot 2^n \cdot 2^{(u+1)\alpha} \leq$$
$$\upsilon \cdot 2^{n\alpha} \cdot 2^{v\alpha} / (4\beta n)^{\alpha-1} \leq P(S_{n-\log(4\beta n)+k}) < \upsilon \cdot 2^{n\alpha} \cdot 2^{(v+1)\alpha} / (4\beta n)^{\alpha-1} \leq \upsilon \cdot (4\beta n) \cdot 2^{n\alpha}$$

From this, and because the sink node of the final directed spanning tree transmits with the power $N \cdot \beta \cdot I_{\max}^{\alpha}$, which could be $N \cdot \beta \cdot 2^{n\alpha}$, we get the lower bound of the total power consumption for the strong connectivity scheduling problem in narrowband networks, which is $\Omega(n \cdot 2^n)$, and the upper bound of the total power consumption, which is $O(n^2 \cdot 2^{n\alpha})$.

THEOREM 6.10: For our NPAW scheduling algorithm in UWB (or any spread-spectrum) networks, the lower bound of the total power consumption is still $\Omega(n \cdot 2^n)$; but the upper bound of the total power consumption is

香港大學圖書館 HKU Libraries

reduced to $O(\frac{1}{m} \cdot n^2 \cdot 2^{n\alpha})$, where n is the number of the nodes and m is the processing gain.

PROOF: With our main algorithm, only links in link length class $S_{h \cdot \log(3n\beta/m)+k}$ can be scheduled simultaneously in the k^{th} scheduling round (k is from 0 to $\log(3n\beta/m)-1$, represented by the columns of Table 6-2); and h is from 0 to $n/\log(3n\beta/m)-1$ (represented by the rows of Table 6-2). In particular, let's consider the link length classes S_k and $S_{n-\log(3n\beta/m)+k}$, which are the shortest length class and the longest length class in the k^{th} scheduling round, respectively.

According to Figure 6-2, suppose the length class S_k is mapped from L_u, we have $u \geq k$; and suppose the length class $S_{n-\log(3n\beta/m)+k}$ is mapped from $L_{n-\log(3n\beta/m)+v}$, we have $v \geq u \geq k$. From line 2 of the subroutine Schedule(), the length class S_k has the highest τ value $\lceil n/\log(3n\beta/m) \rceil$, and the length class $S_{n-\log(3n\beta/m)+k}$ has the lowest τ value, of 1. So according to the nonlinear power assignment scheme in our algorithm, the power $P(S_k)$ assigned to the links in S_k has the property

$P(S_k) \geq \upsilon(3n\beta/m)^{\lceil n/\log(3n\beta/m) \rceil} \cdot 2^{u\alpha}$ and

$P(S_k) < \upsilon(3n\beta/m)^{\lceil n/\log(3n\beta/m) \rceil} \cdot 2^{(u+1)\alpha}$ $\Rightarrow$

$\upsilon \cdot 2^n \cdot 2^{u\alpha} \leq P(S_k) < \upsilon \cdot 2^n \cdot 2^{(u+1)\alpha}$

The power $P(S_{n-\log(3n\beta/m)+k})$ assigned to links in $S_{n-\log(3n\beta/m)+k}$ has the property

$P(S_{n-\log(3n\beta/m)+k}) \geq \upsilon \cdot (3n\beta/m)^1 \cdot (2^{n-\log(3n\beta/m)+v})^\alpha$ and

$P(S_{n-\log(3n\beta/m)+k}) < \upsilon \cdot (3n\beta/m)^1 \cdot (2^{n-\log(3n\beta/m)+v+1})^\alpha$ $\Rightarrow$

$\upsilon \cdot 2^{(n+v)\alpha} / (3n\beta/m)^{\alpha-1} \leq P(S_{n-\log(3n\beta/m)+k}) < \upsilon \cdot 2^{(n+v+1)\alpha} / (3n\beta/m)^{\alpha-1}$

Because $0 \leq k \leq u \leq v \leq \log(3n\beta/m)-1$, we have

香港大學圖書館 HKU Libraries

$$\upsilon \cdot 2^n \cdot 2^0 \le \upsilon \cdot 2^n \cdot 2^{u\alpha} \le P(S_k) < \upsilon \cdot 2^n \cdot 2^{(u+1)\alpha} \le$$
$$\upsilon \cdot 2^{(n+v)\alpha} / (3n\beta / m)^{\alpha-1} \le P(S_{n-\log(3n\beta/m)+k}) < \upsilon \cdot 2^{(n+v+1)\alpha} / (3n\beta / m)^{\alpha-1} \le$$
$$\upsilon \cdot (3n\beta / m) \cdot 2^{n\alpha}$$

From this, we get the lower bound of the total power consumption for the MFSTT problem in (Ultra)-Wideband networks is $\Omega(n \cdot 2^n)$, and the upper bound of the total power consumption is $O(\frac{1}{m} \cdot n^2 \cdot 2^{n\alpha})$.

Table 6-1: Link length classes scheduling (in order) in narrowband networks (from left to right, from top to bottom).

S_0	$S_{\log(4\beta n)}$	$S_{2\log(4\beta n)}$	...	$S_{n-\log(4\beta n)}$
S_1	$S_{\log(4\beta n)+1}$	$S_{2\log(4\beta n)+1}$	...	$S_{n-\log(4\beta n)+1}$
...	...	...	...	...
S_k	$S_{\log(4\beta n)+k}$	$S_{2\log(4\beta n)+k}$	...	$S_{n-\log(4\beta n)+k}$
...	...	...	...	...
$S_{\log(4\beta n)-1}$	$S_{2\log(4\beta n)-1}$	$S_{3\log(4\beta n)-1}$	...	S_{n-1}

Table 6-2: Link length classes scheduling (in order) in wideband networks (from left to right, from top to bottom).

S_0	$S_{\log(3n\beta/m)}$	$S_{2\log(3n\beta/m)}$	...	$S_{n-\log(3n\beta/m)}$
S_1	$S_{\log(3n\beta/m)+1}$	$S_{2\log(3n\beta/m)+1}$	...	$S_{n-\log(3n\beta/m)+1}$
...	...	...	...	...
S_k	$S_{\log(3n\beta/m)+k}$	$S_{2\log(3n\beta/m)+k}$	...	$S_{n-\log(3n\beta/m)+k}$
...	...	...	...	...
$S_{\log(3n\beta/m)-1}$	$S_{2\log(3n\beta/m)-1}$	$S_{3\log(3n\beta/m)-1}$	.....	S_{n-1}

From the above two theorems, we can see that the poly-logarithmic scheduling length comes at the expense of exponential total power

香港大學圖書館 HKU Libraries

consumption. But compared with narrowband networks, by Theorem 6.10, we can see that the upper bound of the total power consumption can be reduced by a processing gain factor in (Ultra)-Wideband networks.

6.5 Concluding Remarks

In this chapter, we show that the scheduling length for the MFSTT problem in the context of (Ultra)-Wideband networks is $O(\log(n/m)\cdot\log n)$. Compared with the currently smallest scheduling length for the MFSTT problem in narrowband networks ,which is $O(\log^2 n)$ in [56], we can see that higher processing gain in wideband networks does help to reduce the scheduling length, especially when $m=\Theta(n)$. In addition, by considering the impact of the arbitrary power assignment on pair-wise transmissions scheduling, we explicitly show that when some node distance function is satisfied, there does not exist any power assignment for simultaneous link scheduling, and thus the scheduling length cannot be further improved via the means of power assignment. Therefore, the scheduling algorithm must take full advantage of the power assignment schemes so that it can simultaneously schedule as many links as possible without violating the physical SINR model. Compared to narrowband networks, we show that there is more room for UWB networks to take full advantage of power control to reduce the scheduling length. More importantly, we explicitly prove that the poly-logarithmic scheduling lengths derived from the nonlinear power assignments are gained at the expense of exponential total power consumption in both narrowband networks and UWB networks.

香港大學圖書館 HKU Libraries

In order to turn our algorithm into a practical network protocol, some problems need to be solved first, including the following.

1) Although in UWB networks, the upper bound of the total power consumption can be reduced by a processing gain factor, the exponential lower bound would not change. Thus reducing the total power consumption without sacrificing the scheduling length is a very interesting and challenging task. To take up this challenge, more refined power assignment strategies, either a new nonlinear power assignment or some completely new power assignment methods may need to be designed.

2) With the nonlinear power assignment, every transmitting node must know its own power scaling factor τ, which is based on some global picture, thus making it difficult to implement the algorithm in a distributed manner. To take up this challenge, implementing some randomized algorithm is a possible method.

3) Our algorithm assumes one channel is used, but actually in MIMO networks (e.g., 802.11n), a node can be equipped with multiple radios and operate on multiple channels. Thus extending our algorithm to multi-radio multi-channel scenarios is a natural idea.

香港大學圖書館 HKU Libraries

Chapter 7 MST_MDCS: A New Algorithm for the MFSTT Problem

We have given a nonlinear power assignment based algorithm for the MFSTT problem in (Ultra)-Wideband networks in the last Chapter. In this chapter, we will give another heuristic algorithm for the MFSTT problem in the context of narrowband networks. As described in the NPAW algorithm, for the topology construction part, we iteratively connect all the nodes on the plane by using a nearest neighbor forest algorithm. This tree topology construction algorithm has also been used in the NPAN-INFOCOM06 algorithm. As for the latest joint link scheduling and topology control algorithm NPAN-IPSN07, it iteratively constructs the tree topology through the nearest component connector (NCC) algorithm [60,61]. This algorithm, however, is almost the same as the nearest neighbor forest algorithm. In addition, by using the graph-based interference model called in-interference degree which is to characterize a node's interference by counting the number of transmitters whose transmission range covers this node, Fussen et al. show that, compared with the NCC algorithm, the minimum spanning tree (MST) algorithm would cause a destructive $O(n)$ in-interference [60,61]. However, the NCC algorithm can only lead to a constant in-interference degree. Thus they prefer to the NCC algorithm from the graph-based interference model's point of view. In this chapter, from the SINR model's point of view, we can greatly lower the scheduling length by using the MST topology.

香港大學圖書館 HKU Libraries

7.1 The MST_MDCS Algorithm for MFSTT

We now give the MST_MDCS algorithm for the MFSTT problem. As the algorithm's name shows, the proposed algorithm is based on the minimum spanning tree algorithm. This means that, for the topology construction part in the MFSTT problem, we choose to first connect all the nodes by using a minimum spanning tree algorithm. After the tree topology has been established, we seek to use the maximum directed cut based scheduling framework MDCS to schedule all the links in the tree.

MST_MDCS: Joint Link Scheduling and Topology Construction for MFSTT

Input: A set of arbitrarily distributed nodes on a plane.
Output: A data gathering tree with the number of timeslots T to schedule all the links in this tree under the SINR model.
1: Construct a directed minimum spanning tree over all the nodes;
2: Schedule all the links in this tree using the MDCS scheduling framework;
3: Return the number of used timeslots T.

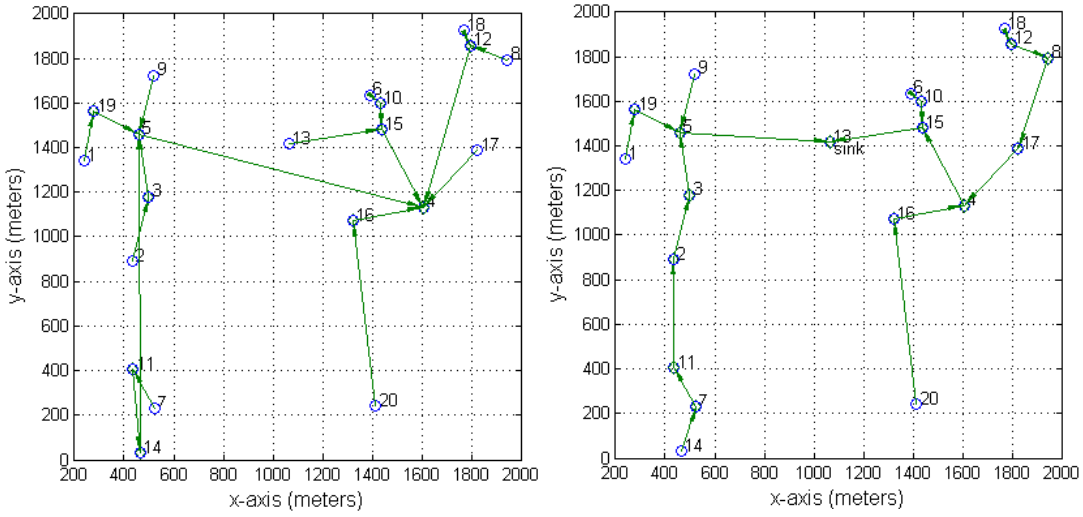
Since the MDCS framework finds a maximum directed cut which also contains a maximum matching in each scheduling phase, we have the following theorem for the number of scheduling phases used in our joint topology construction and scheduling algorithm.

THEOREM 7.1: The number of scheduling phases for our joint topology construction and scheduling algorithm is $O(\log n)$.

PROOF: By using the following two results: (1) For a graph with n edges and a degree k, the number of edges in a maximum matching is lower bounded by $4n/(5k+3)$ [98]; (2) The maximum degree of a minimum spanning tree is bounded by 6, we can easily end the proof.

香港大學圖書館 HKU Libraries

Note that, as shown in [84], we can use a local algorithm to construct the MST. And the degree of this local-MST is also bounded by 6 [84]. In addition, even if we use maximal matching instead of maximum matching in the MDCS scheduling framework, Theorem 7.1 still hold because the number of edges in a maximal matching is lower bounded by $n/(2k-1)$ [97].



(a) (b)

Figure 7-1: a) A tree link topology constructed via a nearest component connector algorithm; b) A tree link topology constructed via a minimum spanning tree algorithm.

7.2 Comparisons with Other Algorithms

First of all, all the nodes are arbitrarily located on a $2000m \times 2000m$ plane and we set the path loss exponent $\alpha = 4$ and the threshold $\beta = 20$. Compared with the simulation setting in Section 3.4, the reason why we set a much higher threshold value here is that the constructed tree topologies are very sparse link topologies. In this case, if we set either a very high path loss exponent or a very low SINR threshold, all the scheduling algorithms could generate very short scheduling lengths which are almost the same as the

香港大學圖書館 HKU Libraries

maximum degree in the tree topology. Two different tree topologies have been shown in Figure 7-1. Specifically, Figure 7-1 (a) gives a tree topology iteratively constructed by the nearest component connector (NCC) algorithm and Figure 7-1(b) shows a tree topology constructed by using a minimum spanning tree algorithm over the same node set. Besides the MDCS scheduling framework and the LDS algorithm, we also implement the NPAN-IPSN07 algorithm which is currently the fastest (in terms of the scheduling length) nonlinear power assignment based link scheduling algorithm that can schedule the NCC-tree (tree constructed with NCC algorithm) in time $O(\log^2 n)$ [56]. Now since the in-interference degree (cf. Section 2.3.1) of a MST topology can be $O(n)$, we can not use the NPAN-IPSN07 algorithm to schedule the links in the MST topology since the SINR constraints may not be satisfied [56]. So for the MST topology, we apply the MDCS and the LDS scheduling algorithms, and for the NCC tree, we can also apply the NPAN-IPSN07 algorithm. But for the NPAN-IPSN07 algorithm, we must pay attention to the background noise value n_i since the scheduling length is also dependent on this parameter. Note that, in this algorithm, when the background noise $n_i < (\alpha - 2)/(2\beta \cdot (\alpha - 1))$, the SNR constraints can not be guaranteed by the proposed nonlinear power assignment (cf. Inequality 2.1). So in this simulation, since we have tested that a much larger n_i value can greatly increase the scheduling length, we set all the n_i as the same value which is a little bit larger than $(\alpha - 2)/(2\beta \cdot (\alpha - 1))$.

The scheduling results can be seen from Figure 7-2. From this figure we have the following observations: (1) the MST topology always yields much

香港大學圖書館 HKU Libraries

shorter scheduling lengths no matter which scheduling algorithm is used; (2) compared with Figure 3-4, Figure 3-5 and Figure 3-6, since the MST and NCC tree topologies have much lower ρ-*disturbance* values, LDS generates shorter scheduling lengths; meanwhile, although the scheduling lengths reductions for the LDS algorithm are not that significant, the scheduling lengths reductions of the MDCS algorithm are quite large; (3) for both MST and NCC tree topologies, the MDCS algorithm always achieves the shortest scheduling lengths; (4) for NCC tree, compared with the NPAN-IPSN07 algorithm, MDCS achieves a much shorter scheduling length.

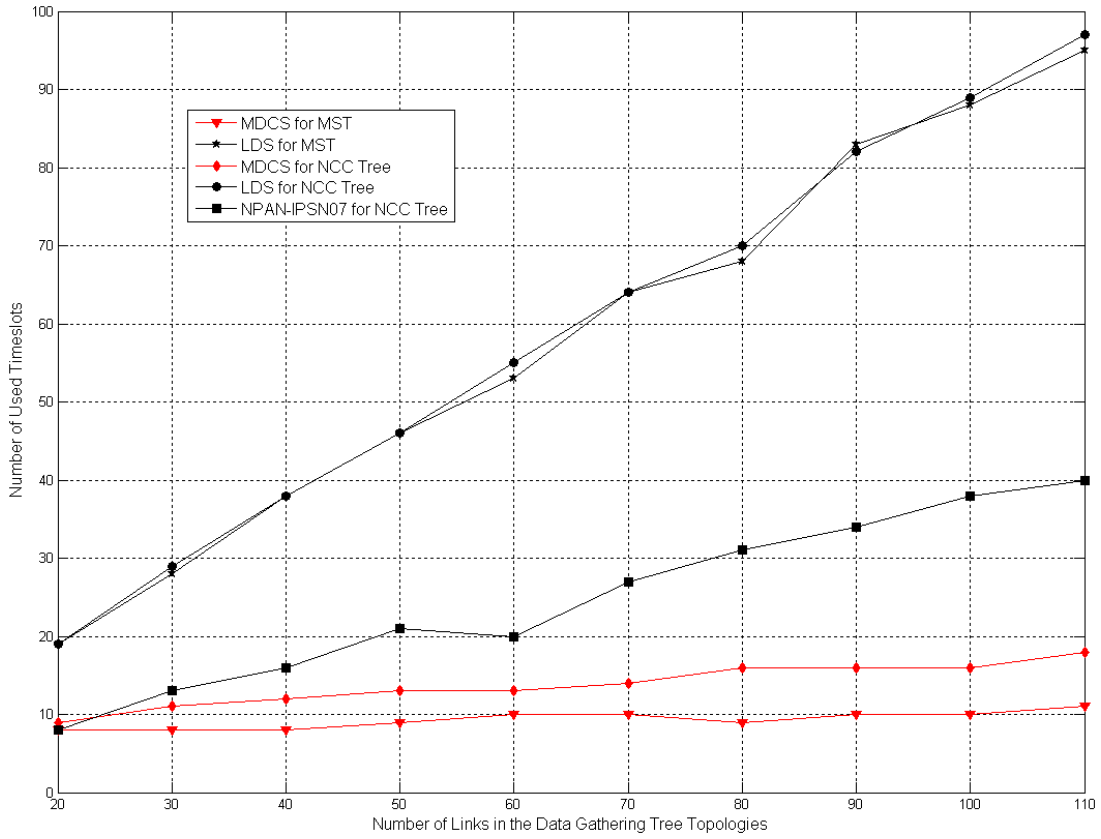


Figure 7-2: Comparisons of scheduling lengths over different tree topologies

7.3 Concluding Remarks

In this chapter, we show that connecting wireless devices with a minimum spanning tree algorithm can significantly lower the scheduling length compared with an iteratively nearest component connector algorithm. This is due to the fact that MST generates shorter links, and shorter links obviously generate much less interferences to other links thus making more links scheduled in the same timeslot. One challenging task for future work is to design local scheduling algorithms that can schedule the links in the tree topology under SINR model both correctly and efficiently.

香港大學圖書館 HKU Libraries

Chapter 8 Conclusions and Future Work

8.1 Conclusions

The work presented in this thesis can be largely classified into four parts. In the first part, we have reviewed the frequently-used interference models and the minimum (frame) length wireless link scheduling algorithms under the SINR model. The subsequent three parts are devoted for the MFSAT, MLSAT and MFSTT problems. Specifically, the second part covers heuristic, exact and approximate scheduling algorithms for the MFSAT problem; the third part proposes both exact and approximation algorithms for the MLSAT problem; and the fourth part incorporates two joint link scheduling and topology construction algorithms for the MFSTT problem.

For the MFSAT problem, the heuristic maximum directed cut based scheduling framework MDCS differs from all the previous heuristic link scheduling algorithms in two aspects. First, the MDCS framework seeks to find a maximum directed cut of the remaining links after finding a maximum link matching. All the existing heuristic scheduling algorithms, however, either find a maximum (maximal) link matching or a subset of the link matching. A large body of them even tries to directly schedule the links without first finding a link matching. The second difference is that the MDCS framework employs the link incremental scheduling algorithm together with the number of neighbors in the pair-wise link conflict graph as a scheduling metric. Extensive simulation results have shown that the MDCS framework

香港大學圖書館 HKU Libraries

significantly outperforms all the previous heurist scheduling algorithms in terms of the scheduling length.

For the MFSAT problem, by transforming it into a set cover problem, we also give the first exact scheduling algorithm and the first polynomial time approximate algorithm with a non-trivial approximation ratio.

For the MLSAT problem, by transforming it into a set multi-cover problem, we also present both exact and polynomial time polynomial space approximation algorithms. In addition, to our knowledge, the proposed exact algorithm for the set multi-cover and the MLSAT problem are the first known exact algorithm for these two problems. And different from the approximation algorithm given in [137], the approximation ratio of our approximate scheduling algorithm is independent of the links' lengths.

For the MFSTT problem, we first generalize the nonlinear power assignment based algorithm for narrowband networks into (Ultra)-Wideband networks. The presented scheduling algorithm demonstrates that a large processing gain in wideband networks can greatly lower the scheduling length. Furthermore, we also prove that all the nonlinear power assignment based scheduling algorithms achieve their poly-logarithmic scheduling lengths at the expense of the total power consumption which is lower bounded by the exponential function of the number of the nodes or links. We also propose another joint link scheduling and topology construction algorithm for the MFSTT problem. Different from all the previous algorithms, this algorithm first construct the tee topology with a minimum spanning tree algorithm rather than the frequently used nearest neighbor forest or nearest component connector algorithm. The simulation results show that, the

香港大學圖書館 HKU Libraries

proposed MST_MDCS algorithm obtains the smallest scheduling length across all the algorithms for the MFSTT problem. Moreover, this simulation results show that connecting all the nodes with the MST algorithm is superior to the nearest component connector algorithm and the nearest neighbor forest algorithm.

8.2 Future Work

There are many open problems in the wireless scheduling area that warrant further attention and investigation. Here we could only touch upon some of them.

Let's first restrict to the MFSAT, MLSAT and MFSTT problems.

First, until the time we are writing the thesis, the hardness of the MLSAT problem is still open. So a rigorous proof is necessary. Second, although we have proposed some polynomial time polynomial space approximation algorithms for the MFSAT and MLSAT problems, they are centralized algorithms. So a local approximation algorithm where each sensor only has limited knowledge of the whole network is necessary for wireless ad hoc and sensor networks that may experience many changes dynamically. For example, we want a sensor node to decide its transmission power locally while guaranteeing higher throughput capacity and lower power consumption. In addition, all the joint link scheduling and topology construction algorithms for the MFSTT problem are also centralized algorithms, thus it entails distributed algorithms for practical network protocols.

We have imposed several assumptions on the wireless link scheduling problems studied in this thesis. So it will be interesting to investigate some of

香港大學圖書館 HKU Libraries

the versions without some restrictions. For example, we can consider multi-radio multi-channel wireless networks, and we can also consider the wireless link scheduling with precedence constraints problem, i.e., some wireless links can not be scheduled before some other links.

There are also many other challenging problems for wireless scheduling under the SINR model.

First, we can consider the minimum length broadcasting (multicasting) scheduling with SINR constraints problems. In these problems, different from point to point link scheduling problems, we must ensure that all the receivers successfully receive the packets from the corresponding sender. These problems have been studied in some papers [53,110], but more work still need to be done.

Second, for the joint scheduling and topology control problem, we can consider some other frequently used topologies in wireless networks. For example, we can consider the minimum length scheduling problem for the dominating set [87], t-spanner or a k-connectivity topology.

Third, just as the authors did in [17], since it becomes very difficult to design an approximate algorithm for arbitrary link topologies, we can resort to designing distributed approximation algorithms for some specific link topologies. For example, we can take full advantage of useful properties of these link topologies, such as the bounded independence number (the number of pair-wisely non-adjacent nodes in each node's k-hop neighborhood) property in growth-bounded-graph [64] to help our algorithm design.

香港大學圖書館 HKU Libraries

Finally, we can also consider the joint link scheduling, power control and routing problems [19,20].

香港大學圖書館 HKU Libraries

References

[1] A. Ephremides. Energy Concerns in Wireless Networks. *IEEE Wireless Communications*, 9(4):48-59, 2002.

[2] P. Björklund, P. Värbrand and D. Yuan. A column generation method for spatial TDMA scheduling in ad hoc networks. Ad Hoc Networks, 2(4): 405-418, 2004.

[3] S. Kompella, J. E. Wieselthier and A. Ephremides. Multi-hop Routing and Scheduling in Wireless Networks Subject to SINR Constraints. In *Proc. 46th IEEE Conference on Decision and Control (CDC)*, New Orleans, LA, USA, 2007.

[4] S. Kompella, J. E. Wieselthier and A. Ephremides. A Cross-layer Approach to Optimal Wireless Link Scheduling with SINR Constraints, In *Proc. Military Communications Conference (MILCOM)*, 2007.

[5] S. Kompella, J. E. Wieselthier and A. Ephremides. Revisiting the Optimal Scheduling Problem. In *Proc. 42nd Annual Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, USA, 2008.

[6] S. Koskie and Z. Gajic. Signal-to-Interference-based Power Control for Wireless Networks: a Survey, 1992-2005. *Dynamics of Continuous, Discrete and Impulsive Systems B: Applications and Algorithms*, 13(2):187-220, 2006.

[7] A.J. Goldsmith and S.B. Wicker. Design challenges for energy-constrained ad hoc wireless networks. *IEEE Wireless Communications*, 9(4):8-27, 2002.

[8] Q. Zhang and Y.-Q. Zhang. Cross-Layer Design for QoS Support in Multihop Wireless Networks. *Proceedings of the IEEE*, 96(1):64-76, 2008.

[9] F. Baccelli, N. Bambos and C. Chan. Optimal Power, Throughput and Routing for Wireless Link Arrays. In Proc. 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Barcelona, Catalunya, Spain, April 2006.

香港大學圖書館 HKU Libraries

[10] Behzad and I. Rubin. On the Performance of Graph-based Scheduling Algorithms for Packet Radio Networks. In *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, 6:3432-3436, San Francisco, December 2003.

[11] J. Grönkvist and A. Hansson. Comparison between graph-based and interference-based STDMA scheduling. In *Proc. MOBIHOC* 2001: 255-258.

[12] Behzad and I. Rubin. Multiple Access Protocol for Power-Controlled Wireless Access Nets. *IEEE Transactions on Mobile Computing*, 3(4): 307-316, 2004.

[13] Behzad and I. Rubin. High Transmission Power Increases the Capacity of Ad Hoc Wireless Networks. *IEEE Transactions on Wireless Communications*, 5(1): 156-165, 2006.

[14] Behzad and I. Rubin. Optimum Integrated Link Scheduling and Power Control for Multihop Wireless Networks. *IEEE Transactions on Vehicular Technology*, 56(1):194-205, 2007.

[15] S. A. Borbash and A. Ephremides. Wireless Link Scheduling With Power Control and SINR Constraints. *IEEE Transactions on Information Theory*, 52(11): 5106-5111, 2006.

[16] S. A. Borbash and A. Ephremides. The Feasibility of Matchings in a Wireless Network. *IEEE Transactions on Information Theory*, 52(6): 2749-2755, 2006.

[17] G. Brar, D. Blough, and P. Santi. Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks. In *Proc. 12th ACM Annual International Conference on Mobile Computing and Networking (MOBICOM)*, Los Angeles, CA, US, Sept. 2006.

[18] D. Chafekar, D. Levin, V.S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Capacity of Asynchronous Random-Access Scheduling in Wireless Networks. In *Proc. 27th Annual Joint Conference*

香港大學圖書館 HKU Libraries

of the IEEE Computer and Communications Societies (INFOCOM), Phoenix, AZ, US, April 2008.

[19] D. Chafekar, V. S. A. Kumar, M. V. Marathe, S. Parthasarathy and A. Srinivasan. Approximation Algorithms for Computing Capacity of Wireless Networks with SINR Constraints. In *Proc. 27th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Phoenix, AZ, US, April 2008.

[20] D. Chafekar, V. S. A. Kumar, M. V. Marathe, S. Parthasarathy and A. Srinivasan. Cross-Layer Latency Minimization in Wireless Networks with SINR Constraints. In *Proc. 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Montreal, Canada, Sept. 2007.

[21] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy and A. Srinivasan. Algorithmic aspects of capacity in wireless networks. In *Proc. SIGMETRICS* 2005: 133-144.

[22] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy and A. Srinivasan. End-to-end packet-scheduling in wireless ad-hoc networks. In *Proc. SODA* 2004: 1021-1030.

[23] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2):388-404, 2000.

[24] K. Jain, J. Padhye, V. N. Padmanabhan, L. Qiu. Impact of Interference on Multi-hop Wireless Network Performance. *Wireless Networks*, 11(4): 471-487, 2005.

[25] J. Tang, G. Xue, C. Chandler and W. Zhang. Link Scheduling with Power Control for Throughput Enhancement in Multihop Wireless Networks. *IEEE Transactions on Vehicular Technology*, 55(3):733-742, 2006.

[26] J. Tang, G. Xue, and W. Zhang. Cross-Layer Design for End-to-End Throughput and Fairness Enhancement in Multi-Channel Wireless Mesh Networks. *IEEE Transactions on Wireless Communications*, 6(10): 3482-3486, 2007.

香港大學圖書館 HKU Libraries

[27] J. Tang, G. Xue and W. Zhang. Cross-Layer Optimization for End-to-End Rate Allocation in Multi-Radio Wireless Mesh Networks. *ACM/Kluwer Journal of Wireless Networks*. In press.

[28] R. L. Cruz and A. Santhanam. Optimal Routing, Link Scheduling, and Power Control in Multi-hop Wireless Networks. In *Proc. 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, San Francisco, CA, US, Mar. 2003.

[29] T. A. ElBatt and A. Ephremides. Joint Scheduling and Power Control for Wireless Ad-hoc Networks. In *Proc. 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, New York, US, June 2002.

[30] T. A. ElBatt and A. Ephremides. Joint scheduling and power control for wireless ad hoc networks. *IEEE Transactions on Wireless Communications*, 3(1): 74-85, 2004.

[31] U.C. Kozat, I. Koutsopoulos, and L. Tassiulas. A Framework for Cross-layer Design of Energy-efficient Communication with QoS Provisioning in Multi-hop Wireless Networks. In *Proc. INFOCOM* 2004.

[32] U.C. Kozat, I. Koutsopoulos, and L. Tassiulas. Cross-Layer Design for Power Efficiency and QoS Provisioning in Multi-Hop Wireless Networks. *IEEE Transactions on Wireless Communications*,5(11): 3306-3315, 2006.

[33] A.K. Das, R.J. Marks, P. Arabshahi and A. Gray. Power controlled minimum frame length scheduling in TDMA wireless networks with sectored antennas. In *Proc. INFOCOM* 2005.

[34] Y. Li and A.Ephremides. A joint scheduling, power control, and routing algorithm for ad hoc wireless networks. *Ad Hoc Networks*, 5(7): 959-973, 2007.

[35] V. Ramamurthi, A.S. Reaz, S. Dixit and B. Mukherjee. Link Scheduling and Power Control in Wireless Mesh Networks with Directional Antennas. In *Proc. ICC*, Beijing China, May, 2008.

香港大學圖書館 HKU Libraries

[36] L. Fu, S. Liew and J. Huang. Joint Power Control and Link Scheduling in Wireless Networks for Throughput Optimization. In *Proc. ICC*, Beijing, China, May 2008.

[37] Q. Wu. Performance of Optimum Transmitter Power Control in CDMA Cellular Mobile Systems. *IEEE Transactions on Vehicular Technology*, 48(2):571-575, 1999.

[38] Q. Wu, W.-L. Wu and J.-P. Zhou. Centralized Power Control in CDMA Cellular Mobile Systems. *Electronics Letters*, 33(2):115-116, 1997.

[39] K. Wang, C.-F. Chiasserini, R.R. Rao and J.G. Proakis. A Joint Solution to Scheduling and Power Control for Multicasting in Wireless Ad Hoc Networks, *EURASIP Journal on Applied Signal Processing*, 2005(1):144-152, 2005.

[40] T.H. Lee, J.C. Lin and Y.T. Su. Downlink Power Control Algorithms for Cellular Radio Systems. *IEEE Transactions on Vehicular Technology*, 44(1):89-94, 1995.

[41] M. Andersin, Z. Rosberg and J. Zander. Gradual Removals in Cellular PCS with Constrained Power Control and Noise. *Wireless Networks*, 2(1): 27-43, 1996.

[42] S.A. Grandhi, R. Vijayan, D.J. Goodman and J. Zander. Centralized Power Control in Cellular Radio Systems. *IEEE Transactions on Vehicular Technology*, 42(4):466-468, 1993.

[43] J. Zander. Performance of Optimum Transmitter Power Control in Cellular Radio Systems. *IEEE Transactions on Vehicular Technology*, 41(1):57-62, 1992.

[44] J. Zander and M. Frodigh. Comment on "Performance of Optimum Transmitter Power Control in Cellular Radio Systems". *IEEE Transactions on Vehicular Technology*, 43(3):636, 1994.

[45] J. Zander. Distributed Cochannel Interference Control in Cellular Radio Systems. *IEEE Transactions on Vehicular Technology*, 41(3):305-311, 1992.

香港大學圖書館 HKU Libraries

[46] Roy D. Yates. A Framework for Uplink Power Control in Cellular Radio Systems. *IEEE Journal on Selected Areas in Communications*, 13(7): 1341-1347, 1995.

[47] G.J. Foschini and Z. Miljanic. A Simple Distributed Autonomous Power Control Algorithm and its Convergence. *IEEE Transactions on Vehicular Technology*, 42(4):641-646, 1993.

[48] R. Jäntti and S.-L. Kim. Power control with partially known link gain matrix. *IEEE Trans. Vehic. Tech.*,52(5):1288-1296, Sept. 2003.

[49] Q.-S. Hua and F. C. M. Lau. The Scheduling and Energy Complexity of Strong Connectivity in Ultra-Wideband Networks. In *Proc. 9th International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, Torremolinos, Malaga, Spain, Oct. 2006.

[50] Q.-S. Hua and F.C.M. Lau. Exact and Approximate Link Scheduling Algorithms under the Physical Interference Model. In *Proc. 5th SIGACT-SIGOPS International Workshop on Foundation of Mobile computing* (DIALM-POMC), Toronto, Canada, Aug. 2008.

[51] Q.-S. Hua, D. Yu, F.C.M. Lau and Y. Wang. Faster Exact Algorithms for Set Multicover and Multiset Multicover Problems. *Submitted to ISAAC 2009*.

[52] T. Locher, P. von Rickenbach, and R. Wattenhofer. Sensor Networks Continue to Puzzle: Selected Open Problems (Invited chapter). In *Proc. 9th International Conference on Distributed Computing and Networking (ICDCN)*, Kolkata, India, Jan. 2008.

[53] Goussevskaia, T. Moscibroda and R. Wattenhofer. Local broadcasting in the physical interference model. In *Proc. 5th SIGACT-SIGOPS International Workshop on Foundation of Mobile computing* (DIALM-POMC), Toronto, Canada, Aug. 2008.

[54] Goussevskaia, Y. A. Oswald, and R.Wattenhofer. Complexity in Geometric SINR. In *Proc. 8th ACM International Symposium on Mobile Ad*

香港大學圖書館 HKU Libraries

Hoc Networking and Computing (MOBIHOC), Montreal, Canada, Sept. 2007.

[55] T. Moscibroda, Y. A. Oswald, and R. Wattenhofer. How Optimal are Wireless Scheduling Protocols? In *Proc. 26th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Anchorage, Alaska, US, May 2007.

[56] T. Moscibroda. The Worst-Case Capacity of Wireless Sensor Networks. In *Proc. 6th International Conference on Information Processing in Sensor Networks (IPSN)*, Cambridge, Massachusetts, US, April 2007.

[57] T. Moscibroda, R. Wattenhofer, and Y. Weber. Protocol Design Beyond Graph-Based Models. In *Proc. 5th Workshop on Hot Topics in Networks (HotNets)*, Irvine, California, US, Nov. 2006.

[58] T. Moscibroda and R. Wattenhofer. The Complexity of Connectivity in Wireless Networks. In *Proc. 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Barcelona, Spain, April 2006.

[59] T. Moscibroda, R. Wattenhofer, and A. Zollinger. Topology Control Meets SINR: The Scheduling Complexity of Arbitrary Topologies. In *Proc. 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Florence, Italy, May 2006.

[60] M. Fussen, R. Wattenhofer and A. Zollinger. Interference Arises at the Receiver. In Proc. of the International Conference on Wireless Networks, Communications, and Mobile Computing (WirelessCom), June 2005.

[61] Martin Fussen. Sensor Networks: Interference Reduction and Possible Applications. Diploma Thesis. Distributed Computing Group, ETH Zurich, Switzerland, 2004.

[62] F. Meyer auf der Heide, C. Schindelhauer, K. Volbert, and M. Gruenewald. Energy, congestion and dilation in radio networks. In *Proc. of the 14th Annual ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, Winnipeg, Canada, August 2002.

香港大學圖書館 HKU Libraries

[63] P. von Rickenbach, S. Schmid, R. Wattenhofer, and A. Zollinger. A Robust Interference Model for Wireless Ad-Hoc Networks. In *Proc. IPDPS* 2005.

[64] S. Schmid and R. Wattenhofer. Algorithmic models for sensor networks. In *Proc. 14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, Island of Rhodes, Greece, April 2006.

[65] R. Wattenhofer. MACbeth: The three witches of media access theory . In *1st IEEE International Workshop on Foundation and Algorithms for Wireless Networking (FAWN)*, Pisa, Italy, March 2006.

[66] M. Burkhart, P. von Rickenbach, R. Wattenhofer and A. Zollinger. Does topology control reduce interference? In *Proc. MobiHoc* 2004: 9-19.

[67] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936-1949, 1992.

[68] F.P. Kelly, A. Maulloo and D. Tan. Rate Control In Communication Networks: Shadow Prices, Proportional Fairness and Stability. *Journal of the Operational Research Society*, V49:237-252, 1998.

[69] X. Lin, Ness B. Shroff and R. Srikant. A Tutorial on Cross-Layer Optimization in Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 24(8): 1452-1463, 2006.

[70] X. Lin and N. B. Shroff. Joint Rate Control and Scheduling in Multihop Wireless Networks. In *Proc. 43rd IEEE Conference on Decision and Control (CDC)*, Paradise Island, Bahamas, December 2004.

[71] X. Lin, Ness B. Shroff. The Impact of Imperfect Scheduling on Cross-Layer Congestion Control in Wireless Networks. *IEEE/ACM Transactions on Networking*, 14(2): 302-315, 2006.

[72] J. Kim, X. Lin and N.B. Shroff. Locally-Optimized Scheduling and Power Control Algorithms for Multi-hop Networks under SINR Interference Models. In Proc. *WiOpt* 2007.

香港大學圖書館 HKU Libraries

[73] G. Sharma, C. Joo and N.B. Shroff. Distributed Scheduling Schemes for Throughput Guarantees in Wireless Networks. In *Proc. Allerton* 2006.

[74] G. Sharma, R.R. Mazumdar and N.B. Shroff. On the Complexity of Scheduling in Wireless Networks. In *Proc. 12th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, Los Angeles, CA, USA, Sept. 2006.

[75] G. Sharma, R.R. Mazumdar, and N.B. Shroff. Maximum Weighted Matching with Interference Constraints. In *Proc. FAWN* 2006.

[76] Joo. A Local Greedy Scheduling Scheme with Provable Performance Guarantee. In *Proc. 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Hong Kong, China, May 2008.

[77] Joo, X. Lin and N.B. Shroff. Understanding the Capacity Region of the Greedy Maximal Scheduling Algorithm in Multi-hop Wireless Networks. In *Proc. 27th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Phoenix, AZ, US, April 2008.

[78] P. Chaporkar, K. Kar and S. Sarkar. Throughput Guarantees Through Maximal Scheduling in Wireless Networks. In Proc. *Allerton* 2005.

[79] X. Lin and S. Rasool. Constant-Time Distributed Scheduling Policies for Ad Hoc Wireless Networks. In *Proc. IEEE Conference on Decision and Control (CDC)*, San Diego, December 2006.

[80] Modiano, D. Shah and G. Zussman. Maximizing Throughput in Wireless Networks via Gossiping. In *Proc. SIGMETRICS/Performance* 2006: 27-38.

[81] P.Santi. Topology Control in Wireless Ad Hoc and Sensor Networks. John Wiley and Sons, Chichester, UK, July 2005.

[82] Y. Wang and X.-Y. Li. Localized Construction of Bounded Degree Planar Spanner for Wireless Ad Hoc Networks. In *Proc. DIALM-POMC* 2003.

[83] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. In *Proc. INFOCOM* 2001.

香港大學圖書館 HKU Libraries

[84] N. Li, Jennifer C. Hou, Lui Sha. Design and Analysis of an MST-based Topology Control Algorithm. IEEE Transactions on Wireless Communications 4(3): 1195-1206 (2005).

[85] K. Moaveni-Nejad and X.-Y. Li. Low-Interference Topology Control for Wireless Ad Hoc Networks. In *Proc. IEEE SECON*, 2005.

[86] Y. Gao, J.C. Hou and H. Nguyen. Topology Control for Maintaining Network Connectivity and Maximizing Network Capacity under the Physical Model. In *Proc. INFOCOM*, Phoenix, AZ, USA, April 2008.

[87] Richa, C.Scheideler, P.Santi. An $O(\log n)$ Dominating Set Protocol for Wireless Ad Hoc Networks under the Physical Interference Model. In *Proc. 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Hong Kong, China, May 2008.

[88] D.S. Johnson. Worst Case Behavior of Graph Coloring Algorithms. In Proc. 5th Southeastern Conf. on Comb., Graph Theory and Computing, 1974, 513-527.

[89] L. Trevisan. Inapproximability of Combinatorial Optimization Problems. Electronic Colloquium on Computational Complexity (ECCC)(065): (2004).

[90] S. Sahni and T. Gonzalez. P-Complete Approximation Problems. *J. ACM*. 23(3): 555-565, 1976.

[91] L. Alfandari, V. Th. Paschos. Master-Slave Strategy and Polynomial Approximation. *Computational Optimization and Applications*, 16, 231-245, 2000.

[92] S. U. Pillai, T. Suel, and S. Cha. The Perron-Frobenius Theorem and Some of its Applications. *IEEE Signal Processing Magazine*, 22(2):62-75, 2005.

[93] R. J. Wood and M. J. O'Neill. An Always Convergent Method for Finding the Spectral Radius of an Irreducible Non-Negative Matrix. *The ANZIAM Journal*,V45:C474-C485,2004.

香港大學圖書館 HKU Libraries

[94] P. Chanchana. An Algorithm for Computing the Perron Root of a Nonnegative Irreducible Matrix. PhD Thesis, North Carolina State University, North Carolina, USA, 2007.

[95] Victor Y. Pan, Zhao Q. Chen. The Complexity of the Matrix Eigenproblem. In *Proc. STOC* 1999: 507-516.

[96] S. Micali and V. V. Vazirani. An *O*(sqrt(|v|) |E|) Algorithm for Finding Maximum Matching in General Graphs. In *Proc. FOCS*, 1980: 17-27.

[97] T. Biedl, E. D. Demaine, C. A. Duncan, R. Fleischer and S. G. Kobourov. Tight Bounds on Maximal and Maximum Matchings. *Discrete Mathematics*, 285(1-3):7-15, 2004.

[98] Yijie Han: Matching for Graphs of Bounded Degree. FAW 2008: 171-173.

[99] Björklund, T. Husfeldt, P. Kaski and Mikko Koivisto. Fourier Meets Möbius: Fast Subset Convolution. In *Proc. 39th Annual ACM Symposium on Theory of Computing (STOC)*, San Diego, California, US, June 2007.

[100] Björklund and T. Husfeldt. Inclusion--Exclusion Algorithms for Counting Set Partitions. In *Proc. 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Berkeley, California, US, Oct. 2006.

[101] Björklund, T. Husfeldt, and M. Koivisto. Set Partitioning via Inclusion--Exclusion. *SIAM Journal on Computing*, to appear.

[102] V. Fomin, F. Grandoni and D. Kratsch. Measure and Conquer: a Simple O(2^0.288n) Independent Set Algorithm. In *Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Miami, Florida, US, Jan. 2006.

[103] M. Fürer and S. P. Kasiviswanathan. Algorithms for Counting 2-Sat Solutions and Colorings with Applications. In *Proc. 3rd International Conference on Algorithmic Aspects in Information and Management (AAIM)*, Portland, OR, US, June 2007.

[104] M. Koivisto. An O*(2^n) Algorithm for Graph Coloring and Other Partitioning Problems via Inclusion--Exclusion. In *Proc. 47th Annual IEEE*

香港大學圖書館 HKU Libraries

Symposium on Foundations of Computer Science (FOCS), Berkeley, California, US, Oct. 2006.

[105] J.W. Moon and L. Moser. On Cliques in Graphs. Israel Journal of Mathematics, Vol. 3, pp. 23-28, 1965.

[106] Takeaki Uno. A Fast Algorithm for Enumeration of Maximal Matchings in General Graphs. *Journal of National Institute of Informatics*, Vol.3, pp. 89-97, 2001.

[107] S. Tsukiyama, M. Ide, H. Aviyoshi and I. Shirakawa. A New Algorithm for Generating all the Maximum Independent Sets. *SIAM Journal on Computing*, V6:505-517, 1977.

[108] E. Tomita, A. Tanaka and H. Takahashi. The Worst-Case Time Complexity for Generating all Maximal Cliques and Computational Experiments. *Theoretical Computer Science*, V363:28-42, 2006.

[109] M. Bomze, M. Budinich, P. M. Pardalos and M. Pelillo. The Maximum Clique Problem. In "Handbook of Combinatorial Optimization" Supplement Vol. A (Eds: DingZhu Du and Panos M. Pardalos), Kluwer Academic Publishers (1999), pp. 1-74.

[110] S. C.-H. Huang, P.-J. Wan, X. Jia, H. Du, and W. Shang. Minimum-Latency Broadcast Scheduling Schemes in Ad Hoc Networks. In *Proc. INFOCOM* 2007.

[111] S. Ramanathan. A Unified Framework and Algorithm for Channel Assignment in Wireless Networks. *Wireless Networks*,5:81-94, 1999.

[112] Balakrishnan, C. Barrett, V. S. Anil Kumar, M. Marathe, and S. Thite. The distance 2-matching problem and its relationship to the MAC layer capacity of ad-hoc wireless networks. *IEEE J. Selected Areas in Communications*, 22(6):1069-1079, August 2004.

[113] W. Wang, X.Y. Li, O. Frieder, Y. Wang and W.Z. Song. Efficient Interference-Aware TDMA Link Scheduling for Static Wireless Networks. In *Proc. 12th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, Los Angeles, CA, US, Sept. 2006.

香港大學圖書館 HKU Libraries

[114]R.C. Qiu, H. Liu, and X. Shen. Ultra-wideband for multiple-access communications. *IEEE Communications Magazine*, 43(2):80-87, Feb. 2005.

[115]Radunovic and J.-Y. Le Boudec. Optimal power control, scheduling and routing in UWB Networks. *IEEE Journal on Selected Areas in Communications*, 22(7):1252-1270, Sept. 2004.

[116]M. Z. Win and R. A. Scholtz. Ultra-wide bandwidth time-hopping spread-spectrum impulse radio for wireless multiple-access communications. *IEEE Transactions on Communications*, 48(4):679-691,2000.

[117]X. Yang and G. de Veciana. Inducing spatial clustering in MAC contention for spread spectrum ad hoc networks. In *Proc. 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Urbana-Champaign, IL, USA May 2005.

[118]Dousse, F. Baccelli, and Patrick Thiran. Impact of interferences on connectivity in ad hoc networks. *IEEE/ACM Trans. Netw.* 13(2): 425-436, 2005.

[119]B.E. Hajek and G.H. Sasaki. Link Scheduling in Polynomial Time. *IEEE Transactions on Information Theory*, 34(5): 910-917, 1988.

[120]F. Cuomo, C. Martello, A. Baiocchi and F. Capriotti. Radio resource sharing for ad-hoc networking with UWB. *IEEE Journal on Selected Areas in Communications*, 20(9):1722-1732, December 2002.

[121]M. Grossglauser and D. Tse. Mobility increases the capacity of ad-hoc wireless networks. *IEEE/ACM Transactions on Networking*, 10(4):477-486, August, 2002.

[122]T. Erlebach, K. Jansen and E. Seidel. Polynomial-Time Approximation Schemes for Geometric Intersection Graphs. *SIAM Journal on Computing*: 34(6): 1302--1323, 2005.

[123]M. Caramia and P. Dell'olmo. Constraint Propagation in Graph Coloring. *Journal of Heuristics*, 8(1):83--107, 2002.

香港大學圖書館 HKU Libraries

[124]M. S. Kodialam and T. Nandagopal. Characterizing the capacity region in multi-radio multi-channel wireless mesh networks. In *Proc. MOBICOM* 2005.

[125]R. Bhatia and M. S. Kodialam. On Power Efficient Communication over Multi-hop Wireless Networks: Joint Routing, Scheduling and Power Control. In *Proc. INFOCOM* 2004.

[126]M. S. Kodialam and T. Nandagopal. Characterizing Achievable Rates in Multi-hop Wireless Networks: the Joint Routing and Scheduling Problem. In *Proc. MOBICOM* 2003.

[127]M. Alicherry, R. Bhatia, and E. L. Li. Joint Channel Assignment and Routing for Throughput Optimization in Multiradio Wireless Mesh Networks. *IEEE Journal on Selected Areas in Communications*,24(11): 1960-1971, 2006.

[128]E.T. Bax. Inclusion and exclusion algorithms for the Hamiltonian path problem. *Information Processing Letters*, 47(4):203-207,1993.

[129]A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. The travelling salesman problem in bounded degree graphs. In *Proc. 35th ICALP* , Iceland, 2008.

[130]U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634-652, 1998.

[131]R.M. Karp. Dynamic programming meets the principle of inclusion-exclusion. *Operations Research Letters*, 1(2):49-51,1982.

[132]J. Nederlof. Fast polynomial-space algorithms using Möbius inversion: Improving on Steiner Tree and related problems, to appear in Proc. ICALP 2009.

[133]Nederlof, A.J.: Inclusion-Exclusion for hard problems. Master Thesis. Utrecht University, The Netherlands (2008)

[134]J. Radhakrishnan. Entropy and counting. In: *Mishra, J.C. (ed.) IIT Kharagpur Golden Jubilee Volume on Computational Mathematics, Modelling and Algorithms*, Narosa Publishers,New Delhi, 2001.

香港大學圖書館 HKU Libraries

[135]V.V. Vazirani. Approximation Algorithms. Berlin: Springer, 2003.

[136]S. Rajagopalan and V.V. Vazirani. Primal-dual RNC approximation algorithms for set cover and covering integer programs. *SIAM Journal on Computing*, 28(2):525-540, 1998.

[137]L. Fu, S. Liew and J. Huang. Power controlled scheduling with consecutive transmission constraints: complexity analysis and algorithm design. To appear in IEEE INFOCOM, 2009.

[138]Q.-S. Hua and F.C.M. Lau. Joint Link Scheduling and Topology Control for Wireless Sensor Networks with SINR Constraints. In *Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice*, IGI Global, to appear.

[139]Q.-S. Hua, Y. Wang, D. Yu and F.C.M. Lau. Set multi-covering via inclusion-exclusion. *Theoretical Computer Science*, doi:10.1016/j.tcs.2009.05.020 (2009)

[140]Q.-S. Hua, D. Yu and F.C.M. Lau. Exact Algorithms for (Multi)set Multicover and #*k*-Multiset Multicover Problems. *Submitted to SODA 2010*.

[141]Richard S. Varga. Matrix iterative analysis. Prentice-Hall, Englewood Cliffs, N.J., 1962.

香港大學圖書館 HKU Libraries