THROUGHPUT OPTIMIZATION OF URBAN WIRELESS MESH NETWORKS

by

Peng Wang

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering

Spring 2009

THROUGHPUT OPTIMIZATION OF URBAN WIRELESS MESH NETWORKS

by

Peng Wang

Approved: _____

Gonzalo R. Arce, Ph.D.
Chair of the Department of Electrical and Computer Engineering

Approved: _____

Michael J. Chajes, Ph.D.
Dean of the College of Engineering

Approved: _____

Debra Hess Norris, M.S.
Vice Provost for Graduate and Professional Education

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Stephan K. Bohacek, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Charles Boncelet, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Daniel Weile, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Errol L. Lloyd, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Chien-Chung Shen, Ph.D.
Member of dissertation committee

To

My Parents: Wancheng Wang and Yu Wang

and

My Wife: Rui Yu

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to my Advisor Prof. Stephan Bohacek. Without his guidance and encouragement, the completion of this thesis would never have happened. During the course of my research, he was always able to make time for me no matter how busy he was. He never ceases to surprise me with his approach and attitude towards a problem. It is hard to imagine having done my thesis without his help. I feel lucky to have an understanding and supporting advisor like him.

I thank my dissertation committee members: Prof. Charles Boncelet, Prof. Daniel Weile, Prof. Errol Lloyd and Prof. Chien-Chung Shen. I would like to thank them for providing valuable advise and investing support in me.

Special thanks to Prof. David Mills for advising my research in network congestion control. His great vision is always a source of inspiration of my research. Thanks to Prof. Gonzalo Arce for his support.

I extend my gratitude to my colleagues in our research group. Jonghyun, Vinay, Hweechul, Andreas and Carlos have all made it a very enjoyable time working in our research lab. I have benefited a lot from them.

I would also like to thank my parents and my wife for their unconditional support and love. This dissertation is dedicated to them.

My doctoral work was funded by the U.S. Army Research Laboratory under the the Collaborative Technology Alliance Program, and the University of Delaware. My thanks to all for their generous funding that made this research possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Interference and collisions greatly limit the throughput of mesh networks that used contention-based MAC protocols such as 802.11. It is widely believed that significantly higher throughput is achievable if transmissions are scheduled. However, since the typical approach to this throughput optimization requires optimizing over a space that is exponential in the number of links, the optimal throughput has appeared to be computationally intractable for all but small networks. This research presents techniques that are typically able to efficiently compute optimal schedules as well as optimal routing.

The technique consists of three layers of optimization. The inner-most optimization computes an estimate of the throughput. This optimization is a standard linear or nonlinear constrained optimization, depending on the objective function. The middle iteration uses the Lagrange multipliers from the inner-most optimization to modify the space over which inner-most optimization is performed. This is a graph theoretic optimization known as the maximum weighted independent set (MWIS) problem. The solvability of this problem is extensively studied, and the empirical evidence shows that usually MWIS arising from wireless mesh network can be solved quickly. The outer-most optimization uses the Lagrange multipliers from the inner-most optimization to find optimal routes. This optimization solves several least cost paths problems and several maximum weighted independent set problems.

Together, these techniques allow optimal schedules to be computed for networks with hundreds and even a few thousand links, and allow optimal routes to be

computed for networks with a few hundred links. Thus, the approach scales to the size of current and planned urban mesh networks. The techniques have been verified on networks generated with a realistic urban propagation simulator. In this setting, it is shown that optimal scheduling increases the throughput by a factor between three and ten over 802.11 CSMA/CA, depending on the density of the mesh routers and gateways.

This thesis also studies the communication models. It is well known that the traditional protocol models have the drawback that they do not accurately model interference. Therefore, the actual throughput provided by these traditional protocol models is poor no matter how good the theoretical throughput offered. A general SINR protocol model is proposed to more accurately represent the interference. Furthermore, techniques were developed to overcome the interference from multiple sources.

The ability to accommodate links that support multiple bit-rates and multiple transmit powers (e.g., 802.11) has been largely neglected by previous efforts on throughput maximization. Accommodating all possible bit-rates and transmission powers into the schemes developed in this thesis greatly increases the computational complexity. Thus, several heuristics are developed and examined.

Chapter 1

INTRODUCTION

By providing connectivity to mobile users, mesh networks are poised to become a major extension to the Internet. More than 400 cities and towns have plans to deploy mesh networks, and several dozen cities have already deployed mesh networks [1]. While some deployments have been in smaller cities, such as Mountain View, CA, and Milpitas, CA, some deployments have been in larger cities such as Corpus Christi's 147 sq. mile deployment. Furthermore, large cities such as Philadelphia and San Francisco are in the final planning stages of city-wide, dense deployments. These mesh networks are meant to enhance city and emergency services communication as well as to provide city-wide, low-cost, ubiquitous Internet access for residents and visitors. Such networks promise to bring dramatic changes to data accessibility and hence have a major impact on society.

1.1 The Mesh Network Scenario

In the research literature, two types of mesh networks are discussed. One is referred to as an enterprise mesh network. Such networks cover a relatively small area, e.g., a college or a business campus, or may even be confined to a single building such as a hospital or airport terminal. The second type of mesh network are large-scale urban mesh networks (LUMNets or simply UMNets). However, as the deployment of these networks continues to expand, they are not always in urban areas. Thus, a better term is large-scale mesh networks (LMNets). Today, while

many of the current deployments include only outdoor nodes, several include indoor nodes as well (e.g., Google's network in Mountain View, CA).

A distinguishing feature of urban mesh networks is the presence of a large number of infrastructure nodes, each with relatively small coverage (i.e., pico or femto cells). This contrasts cellular networks that have far fewer infrastructure nodes, each with large regions of coverage. Urban mesh networks also differ from traditional mobile ad hoc networks in that they have a fixed infrastructure that includes some wired base stations. On the other hand, mesh networks are similar to ad hoc networks in that paths may cross multiple wireless links. However, a path in a mesh network typically includes multiple infrastructure to infrastructure hops and a single infrastructure to mobile hop. These mesh networks, as opposed to cellular networks, wireless LANS, MANETs, and sensor networks, offer distinct features that allow throughput to be realistically maximized.

- The infrastructure nodes (INs) in a mesh network are powered, and hence the energy is not a critical concern.

- INs can have substantial computational power. This power can be in the form of general microprocessors or specialized computational engines such as DSPs and FGPAs. Thus, in terms of computation, INs may be significantly different from nodes in MANETs and sensor networks. It is important to note that while INs can include significant computational power, including computational engines in INs must be justified with significant performance improvements.

- In addition to computational power, INs may also have specialized hardware to support time synchronization. This could include GPS, chip-based atomic clocks, or CDMA-based clocks.

- Unlike nodes in MANETs, INs are stationary. This allows the INs to make intense measurements of the environment. The channels between INs are not static. However, there exists time-varying channels. Since the INs are stationary and not energy constrained, it is possible to make detailed channel measurements and estimate the channel model parameters. Since this estimation can proceed continuously, detailed channel models are feasible.

- While sensor and MANETs are purely self-organizing, in terms of administration, mesh networks resemble wired networks. For example, with the help of distributed monitors, network administrators will measure and "tune" the network. Furthermore, since mesh service providers must meet service agreements, the capability to perform manual traffic engineering is desirable.

- While client nodes will follow the standard versions of 802.11, in today's deployments, the INs within a mesh network are typically made by a single vendor. Hence, specialized protocols can be used for IN to IN communication, while still using standard 802.11 for communication between clients and INs. For example, in today's deployments, vendors use proprietary routing protocols (e.g., [2]). Thus, small modifications to 802.11 for IN to IN communication is realistic. The development of entirely new MAC protocols is also possible, but, of course, this must be justified with substantial improvements in performance.

1.2 Motivation

Based on realistic simulations of mesh networks [3], it is clear that coverage and throughput are critical problems facing mesh networks. While throughput has been a long standing problem in wireless networking, the problem has now become urgent. Specifically, with the current data rates provided by 802.11, the mesh networks that are currently being deployed will likely not meet the data rate needs of

the users. For this reason, within the industry that is currently deploying mesh networks, throughput has been identified as the chief concern facing future deployments [4]. On the other hand, maximizing throughput has been an open problem for at least twenty years. However, this thesis presents significant advances in throughput maximization. The techniques presented here allow throughput to be maximized for networks with several hundreds or even thousands of links. Hence, these techniques are suitable for all current and planned mesh networks. Moreover, based on computations from realistic urban mesh networks, it is concluded that improvements might exceed a factor of 10.

1.3 Problem Statement

The main factor that limits throughput in wireless mesh networks is interference which is the consequence of sharing a communication medium. The concurrent transmissions of multiple links cause interference at each link, and some links may reduce the transmission rate to overcome the interference. Moreover, some links may not be able to transmit if the interference is too large. Therefore, it is known that the transmissions must be scheduled in order to achieve high throughput.

The basic challenge facing throughput maximization is that the space of solutions grows exponentially with the number of links. For example, if power control is not used, then one must maximize over a polytope with 2^L extreme points, where L is the number of links. More specifically, we define an *assignment* to be a specification of which links are transmitting and which links are not transmitting. A *schedule* is the convex sum of assignments, which specifies the fraction of time that each assignment is used. The main challenge to the throughput maximization problem has been that the space of all assignments is too large. For example, when $L = 88$, the space of all assignments is 2^{88}, which is larger than the number of nanoseconds in the age of the universe.

In order to solve the throughput maximization problem, the space of assignments must be reduced. One approach, followed in [5], is to arbitrarily restrict the focus to a subset of all assignments. However, this approach reduces the resulting throughput. For example, in [5], it was found that increasing the size of the solution space improves the throughput. Another approach is to develop a heuristic to determine the subset of considered assignments [6], [7]. However, the performance of these methods typically decreases as the number of links increases [6], [7]. Finally, one can take a brute force approach, and consider the entire set of assignments. This approach is taken in [8], where, due to computational difficulties, the largest network that could be solved has only 15 links.

Like several other approaches to throughput maximization, the approach presented here also restricts the solution space. However, we employ an iterative approach that uses the Lagrange multipliers from the previous iteration to help select the subset of assignments for the next iteration. The result is that after relatively few iterations, the set of considered assignments is relatively small but is such that the optimal solution restricted to the subset provides the same performance as does the optimal solution when all assignments are considered. Also, it was shown in Chapter 3 that the algorithm of optimal scheduling converges geometrically or algebraically in terms of different objective functions.

Besides solving the basic throughput maximization problem, this thesis also considers a wide range of so called "communication models". The literature today contains many different communication models, which can be classified into two types, named as physical communication model and protocol communication model. It is well known that protocol communication models are unrealistic because of the inaccurate modeling of the interference and the neglect of the aggregate interference. We propose a general SINR protocol communication model, and the drawbacks of the protocol communication models can easily be alleviated.

An important aspect of our solution to the throughput optimization problem is reducing the original optimization problem to a graph theoretic problem known as the maximum weighted independent set (MWIS) problem. While it is known that in the worst case the MWIS problem is NP-hard, there has been a huge amount of effort in solving and understanding the MWIS problem. This thesis leverages this work to produce several ways to attack the MWIS problem that arises when optimizing throughput. Two basic approaches are investigated, namely, exactly solving the MWIS problem, and approximately solving the MWIS problem. In the first approach there are several options including specialized computational techniques for finding MWIS, specialized techniques for finding the largest weighted clique, and general computational techniques for solving integer programming problems. Similarly, there are many approximation techniques. Along with approximate methods, we compute the Lovasz number in order to determine whether the approximation is indeed optimal.

The complexity of the set of MWIS problems that arise in wireless schedule is unknown. Without theoretic analysis of the complexity of MWIS, a large number of computational experiments with more than 10,000 topologies were examined to explore the complexity of MWIS. The empirical evidence shows that MWIS from the wireless scheduling can be quickly solved, for example it takes around 1 sec to find a MWIS for a wireless mesh network with 2048 nodes.

While the techniques described above can quickly find the optimal schedule for large networks, the problem becomes computationally complex if links use a wide range of bit-rate and transmission powers. In this case, each physical link is represented as many logical links, one logical link for each combination of transmission power and bit-rate. Techniques to reduce the number of bit-rates and transmission powers considered greatly improve the computational complexity, but may also reduce the resulting throughput. This thesis examines these issues in detail.

Once optimal schedules can be efficiently computed, it becomes possible to compute optimal routes. In general, the computational complexity of routing is exponential in the number of nodes in the network. However, using a technique that is similar to the technique used to develop optimal schedules; the routing problem is decomposed into an iterative problem where Lagrange multipliers from the schedule optimization are used to adjust the routing problem.

1.4 Key Contributions

This dissertation provides an approach to maximize the throughput of an urban mesh network. The following are the key contributions of this dissertation.

- An optimization-based approach to computing the optimal throughput of urban mesh network is provided. While previous efforts have been unable to compute optimal schedules for networks with more than a few tens of links, the methods presented here can be applied to networks with hundreds or even thousands of links. The algorithms have been tested on realistic mesh networks generated by the UDelModels, a publicly available tool for computing urban propagation.

- A detailed analysis of Protocol Communication Models and Physical Communication Models is provided. Moreover, a general SINR protocol communication model is proposed to accurately represent the pairwise interference, while the techniques are also developed to accommodate the interference from multi-sources.

- A detailed analysis of methods to compute new assignments via solving a maximum weighted independent set (MWIS) problems, where the graph arises from an urban mesh network. One of the main obstacles of prior work on throughput optimization is the inability to solve related graph theoretic optimization

problems such as finding MWIS or maximum matchings. This thesis shows that a range of techniques can be used to compute or estimate the MWIS. Furthermore, a large number of simulations show that MWIS in the wireless mesh network can be solved very quickly.

- The technique of finding optimal routing is similar to that of optimal scheduling. While most previous works are computationally intractable and/or neglect co-channel interference, this thesis provides a solution which is practically feasible due to the quick solvability of optimal scheduling, and is able to account for the co-channel interference if the general SINR protocol communication model is used.

- While most prior works consider a link with a single bit-rate and a single transmit power, this thesis presents techniques to compute throughput when links support different bit-rates and transmit powers.

1.5 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 briefly introduces some related work. Chapter 3 presents the optimization-based approach to finding the optimal throughput of urban mesh network. The approach focuses on reducing the size of the space of assignments and finding a new assignment that passes the linear test (3.9). Chapter 4 presents the communication models that can be divided into Protocol Communication Models and Physical Communication Models. The throughput of a network depends on the capabilities of the MAC and physical layer. Several examples of each type of model are discussed. Chapter 5 presents the methods to construct the set of considered assignments. Brief descriptions of the construction of initial set of assignments and the redundancy removal are provided. Various algorithms to find the new assignment are the keys of this chapter.

Then, chapter 6 presents the methods to generate different kind of network topologies. Several numerical experiments used to validate the above results are provided. Chapter 7 presents the solvability of MWIS in the wireless mesh network through a large set of simulations. It shows that the MWIS can be solved quickly although the worst case complexity of MWIS is NP-hard. Chapter 8 provides the joint optimal routing and optimal scheduling scheme, and Chapter 9 presents the optimal scheduling when multi-bit rates and power control are deployed. Finally, Chapter 10 provides concluding remarks and the directions for future work.

Chapter 2

RELATED WORK

The optimal throughput of multi-hop wireless network has been an active area of research in the recent years. A large number of papers have been published regarding the methods to find the optimal or approximate optimal throughput. We focus on the optimization-based approach.

Utility optimization of wired networks was pioneered by the seminal works of Kelly [9], Low [10]. The same framework has been extended to ad hoc networks in [11, 12]. MAC scheduling, power control, and routing are closely related to the network throughput and will be reviewed respectively.

2.1 Scheduling

2.1.1 Interference Model

The main factor that limits throughput in wireless mesh networks is interference which is the consequence of sharing a communication medium. Hence, an accurate modeling of interference is fundamental in order to derive theoretical and/or simulation-based results. In the literature, two main interference models are the protocol model and the physical model.

Much of the prior work utilizes the simplistic protocol communication models such as the primary (i.e., one-hop) and secondary (i.e., two-hop) conflict models which are oblivious to the co-channel interference. The primary interference model, also called the *node exclusive* model, merely assumes that a node cannot simultaneously transmit and receive on the same channel. In other words, two links can

transmit simultaneously on the same channel no matter how close they are, as long as they do not share the same node. As will be shown in detail in Chapter 6, the co-channel interference is fully neglected and the node exclusive model is not realistic.

It is well known that finding a new assignment is equivalent to finding a Maximum Weighted Independent Set (MWIS) in the weighted conflict graph. Unfortunately, in the worst-case, the MWIS problem is NP-hard [13]. It is important to note that under the node exclusive communication model (i.e., when co-channel interference does not arise), the MWIS problem reduces to a maximum matching problem, which is known to be solvable in polynomial time [14, 15] and in some cases can be solved with distributed message passing [16]. It is also known that a maximal matching is a 1/2 approximation of a maximum matching [17]. Also, there exists a distributed approximate algorithm [18] which achieves 2/3 approximation of a maximum matching.

Therefore, neglecting co-channel interference guarantees that the throughput maximization problem has polynomial complexity. Due to this tractability, the node exclusive model has received considerable attention. Consequently, tremendous progress has been made and the general approaches to throughput maximization have been well developed. Many of these approaches are exploited by our prior effort. Works that rely on the node exclusive model include [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31].

The 2-hop or k-hop communication model is an extension of the node exclusive model. However, it still lacks a realistic representation of co-channel interference. In fact, it has the important drawback that it does not consider the accumulation of interference, that is, while communication across link A might be possible while node B *or* node C are transmitting, it might not be possible to communicate across link A if *both* B and C are transmitting. Furthermore,

this model also results in the throughput maximization problem being NP-hard (in the worst case) [32]. Efforts that utilize the 2-hop or k-hop model include [33, 34, 32, 35, 36, 37, 38, 39, 29, 40, 26, 41]. Due to the fact that it can be used to mimic the behavior of CSMA/CA networks such as IEEE 802.11, the 2-hop interference model has been most widely used in the literature.

In the physical interference model, a transmission from node u and v is successful if the SINR (signal to interference and noise ratio) is above a certain threshold, whose value depends on the desired data rate. The effect of co-channel interference or aggregate co-channel interference can be accurately modeled under this model. There has also been significant efforts to find the throughput of a wireless mesh network under the physical model, but focusing on suboptimal solutions. Among the notable works in this direction are [7] and [40] in that they find a bound on how far the solutions are from optimal. However, these bounds are only valid under free-space propagation. Other works that focus on suboptimal throughput and do consider co-channel interference include [43] and [5]. To the best of our knowledge, we are the first to present a viable algorithm to achieve maximum throughput under co-channel interference.

2.1.2 Optimization Space

The main obstacle facing throughput maximization is that the size over which the optimization is performed is exponential with the number of links [8, 44, 5, 45]. For example, if power control is not used, then one must maximize over a polytope with 2^L extreme points, where L is the number of links. In order to solve the throughput maximization problem, the space of assignments must be reduced. One approach, followed in [5], is to arbitrarily restrict the focus to a subset of all assignments. However, this approach reduces the resulting throughput. For example, in [5], it was found that increasing the size of the solution space improves the throughput. Another approach is to develop a heuristic to determine the subset

of considered assignments in [6, 7]. However, the performance of these methods typically decreases as the number of links increases. Finally, one can take a brute force approach, and consider the entire set of assignments. This approach is taken in [8], where, due to computational difficulties, the largest network that could be solved has only 15 links.

2.1.3 Global and Local Information

Perhaps one of the most serious drawbacks of our algorithm is that it requires global knowledge. It is quite common for scheduling to require global knowledge (e.g., [19, 20, 46, 7, 36, 40, 47, 43, 5, 8]). On the other hand, a simple fully distributed approach is appealing and has advantages in terms of robustness.

In the literature, several distributed techniques have been suggested. These include the subgradient-based schemes [47, 8, 48, 49, 50, 51, 52, 31, 27], back-pressure schemes [19, 31], and randomized searching techniques [53]. In the case of back-pressure, it should be pointed out that the schemes described in [19, 31] rely on centralized scheduling. However, the routing and congestion control are distributed.

For the case where co-channel interference does not arise, there exists distributed subgradient-based schemes that are approximately optimal. As explained in [54], the subgradient-based scheme suffers from poor performance in large networks. While the technique of using Lagrange multipliers to help find good assignments appears to be new, there are several prior efforts that share some computational techniques with the proposed effort. These efforts include [24, 25, 32, 26, 55, 56, 57].

2.2 Routing

Like the optimal scheduling problem, there has been extensive work on joint optimal scheduling and routing, but the techniques are computationally intractable and/or neglect co-channel interference [58, 59, 31, 8, 5, 28, 40, 60, 41, 20, 19, 61].

One challenge is that, in the worst case, the number of paths between a source-destination pair increases exponentially with the size of the network. Thus, like scheduling, a naive approach is not viable for moderately sized networks.

However, in a way that is similar to our solution to the scheduling problem, we have developed a technique that iteratively adds paths to the set of considered paths until the optimal set of paths is found.

2.3 Power Control

Power control in multi-hop wireless network is an extensively researched topic [8, 47, 62]. One approach is to explore the theoretical aspects of power control for throughput maximization and not directly address the size of the problem. In [8], Cruz shows that power control does not improve the performance with the assumption that SINR at the receiver is low. The senders always transmit with full power. In [47], it is showed that power control can converge to the global optimal of the CDMA network when we assume that receivers have high SINR. However, in the realistic wireless mesh network, it is impossible to determine which links have high SINR and which links have low SINR, because it depends on many of factors such as channel gains, scheduling, and transmission powers of the active links. Therefore, it is difficult to find the actual effect of power control on the optimal throughput in the realistic wireless mesh network.

On the other hand, many heuristic methods are developed to solve the power control problem and obtain good throughput for wireless mesh network. As we mentioned above, we do not know how far the good throughput is from the optimal due to the unknown of the optimal. Thus, a practical computation of optimal scheduling and optimal power control is proposed in our work. Moreover, the scheduling supports multiple data rates for each link which is seldom considered in previous works. The framework described in this work appears to be significantly more general than prior research.

Chapter 3

OPTIMAL SCHEDULING

3.1 Introduction

The throughput of multihop wireless networks has been the subject of intense research ([31, 44, 8, 6, 27]). There is extensive motivation for this research. For example, in the mesh network setting, determining throughput is useful for network planning. Also, routing and throughput are closely related, and hence, the performance of routing protocols can be greatly improved if the throughput that results from a particular routing can be determined. The optimal throughput provides an upper bound on heuristic techniques, and provides a means to judge the quality of the heuristic. Also, there is theoretical interest in throughput; for example, there has been interest in how the throughput varies as a function of the number of nodes [63].

Generally, there have been two approaches to maximize the throughput of a wireless network, namely, a heuristic-based approach and an optimization-based approach. In the optimization-based approach, an objective function is defined and this objective is maximized subject to the constraints imposed by interference. For example, it is common to define an objective function to be a concave increasing function of flow rates; this function is often referred to as a utility function and it can be shown that maximizing such functions results in fairness among flows ([9, 64, 27]). This chapter focuses on this class of problems. Other, non-utility-based approaches include [8], which minimizes a linear function of transmission powers subject to a

link bit-rate constraint and other constraints imposed by interference. A different throughput problem is related to maximizing an objective function when the traffic demands are stochastic [65].

A critical problem with the optimization-based approaches is that the resulting problem is computationally difficult. More specifically, in order to maximize throughput, the optimal allocation of resources must be determined. This allocation is defined by a schedule that dictates when links transmit and at which frequencies and powers they transmit. However, the space of schedules is extremely large. For example, if power control is not used, then one must maximize over a polytope with 2^L extreme points, where L is the number of links. More specifically, we define an *assignment* to be a specification of which links are transmitting and which links are not transmitting. A schedule is the convex sum of assignments; hence the assignments are the extreme points of the space of schedules. If power control is not used, then the space of assignments contains 2^L elements. If power control is used, then an assignment specifies not only which links are transmitting, but also the transmission power. In this case, the space of assignments is $[0,1]^L$. If one attempts to approximate this continuous space with a discrete grid such as $\{0, 0.33, 0.67, 1\}^L$, then the space has 4^L elements. To comprehend the size of 2^L, consider a network with only 88 links, which is far smaller than the mesh networks being deployed throughout the world. However, 2^{88} *nsec* is the approximate age of the universe. Hence, it is intractable to even initialize the problem. Due to the exponential dependence on the number of links, after considerable reduction of the problem, [8] reports being unable to compute the optimal schedule for a network with more than 15 links.

Due to the computational difficulties, there has been extensive research of heuristic-based approaches (e.g., [6, 34, 33]). While such methods are tractable, and may greatly improve the performance, one is unsure as to how much performance could be further improved if optimization was used.

This thesis focuses on the optimization-based approach which focuses on reducing the size of the space of assignments. There are two key theoretical results that underpin this approach.

- While the optimization may be performed over a space with 2^L assignments (if power control is used, the space is $[0,1]^L$), the optimal solution is the combination of no more than L assignments. Therefore, if these L assignments were somehow known in advance, then the optimization could be performed over a space with L assignments and the result would be identical to the one found by optimizing over the space of all assignments.

- Guided by this result, we focus on searching for these special L assignments. From a solution of the optimization problem over an arbitrary set of L elements, either

  - a new set of elements can be found that will improve the solution, which, in turn, leads to a better set of elements, and so on,
  - or, if no set of better elements exists, then the current set of elements is optimal.

Chapter 5 presents several methods to search for assignments that are known as Maximum Weighted Independent Set (MWIS) problem. Chapter 6 examines the performance of these search methods.

The remainder of the chapter proceeds as follows. In the next section, the system model and notation are presented. The algorithm for optimal scheduling and the corresponding convergence property are discussed in Section 3.3.

3.2 System Model and Problem Formulation

A router-to-router flow is denoted by ϕ, with Φ denoting the set of all such flows. To improve presentation, it is assumed that all flows use a single path,

however, the extension to multipath is straightforward. The data rate of flow ϕ is denoted by f_ϕ, and the path followed by flow ϕ is denoted by $P(\phi)$. The set of all considered paths is $\mathcal{P}$. Using this notation, the total data rate sent over link x is $\sum_{\{\phi|x\in P(\phi)\}} f_\phi$, where $\{\phi\,|x \in P(\phi)\}$ is the set of flows that cross link x. All links are directional.

We define an *assignment* to be a vector $\mathbf{v} = \begin{bmatrix} v_1 & \cdots & v_L \end{bmatrix}$, where there are L links in the network and where $v_x \in \{0,1\}$ with $v_x = 1$ implying that link x is active during assignment $\mathbf{v}$. It is possible to extend this approach to accommodate links with multiple bit-rates and multiple transmit powers. The *set of considered assignments* is denoted by $\mathcal{V}$, while the *set of all assignments* is denoted by $\overline{\mathcal{V}}$. In this simple case where $v_x \in \{0,1\}$, $\overline{\mathcal{V}}$ has 2^L assignments. The size of $\overline{\mathcal{V}}$ is the main challenge facing optimal scheduling. Thus, typically, we only consider a subset of all assignments, i.e., $\mathcal{V} \subsetneq \overline{\mathcal{V}}$.

The data rate across link x during assignment $\mathbf{v}$ is denoted by $R(\mathbf{v}, x)$. In general $R(\mathbf{v}, x)$ is a complicated function. However, here a simple binary relationship is used to define $R(\mathbf{v}, x)$. Specifically,

$$R(\mathbf{v}, x) = \begin{cases} R_x \text{ if } v_y = 0 \text{ for all } y \in \chi(x) \\ 0 \text{ otherwise} \end{cases}, \tag{3.1}$$

where $\chi(x)$ is a set of links that conflict with x, i.e., $y \in \chi(x)$ if simultaneous transmissions over x and y are not possible. R_x is the *nominal data rate* over link x. Techniques to select the nominal data rate for a link are detailed in chapter 4. Here, let R_x be the maximum data rate that the SNR across link x can support. Note that this definition of $R(\mathbf{v}, x)$ neglects the possibility of transmission errors due to the aggregate interference from several links not in $\chi(x)$. However, as discussed in Section 5.3, such problems can easily be addressed. All computations in this work use this technique, and hence the computed throughputs account for multiple interferers. This definition of $R(\mathbf{v}, x)$ also neglects the possibility that $R(\mathbf{v}, x)$ can

take intermediate values between R_x and 0. For example, in the face of moderate interference, retransmissions will result in an effective data rate that is below R_x. With a slight modification, this behavior can be supported by employing a multi-valued definition of $R(\mathbf{v}, x)$. Future work will investigate such modifications.

The set of conflicting links, $\chi(x)$, depends on the communication model. Arguably, the $SINR$ protocol communication model is the most relevant and is the model used in this work. Let $SINR(x, y)$ be the SINR at the receiver of link x when link y is also active. That is, $SINR(x, y) := H_{x,x}/(H_{y,x} + \mathcal{N})$ where $H_{x,x}$ is the strength of the signal transmitted from the transmitter of link x to the receiver of link x, $H_{y,x}$ is the strength of the signal transmitted from the transmitter of link y to the receiver of link x, and $\mathcal{N}$ is the strength of the noise. Then, the SINR communication model specifies that $y \in \chi(x)$ if $SINR(x, y) < T(x)$ or $SINR(y, x) < T(y)$, where $T(x)$ and $T(y)$ are thresholds that depend on the modulation schemes.

A schedule is a convex combination of assignments. Specifically, a schedule is a set $\{\alpha_{\mathbf{v}} : \mathbf{v} \in \mathcal{V}\}$ where $\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1$ and $\alpha_{\mathbf{v}} \geq 0$. Thus, $\alpha_{\mathbf{v}}$ is the fraction of time that assignment $\mathbf{v}$ is used. With this notation, the total data rate that the schedule $\boldsymbol{\alpha}$ provides over link x is $\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R_x v_x$. Finally, the throughput optimization problem is

$$\max_{\boldsymbol{\alpha},\mathbf{f}} G(\mathbf{f}) \tag{3.2a}$$

subject to:

$$\sum_{\{\phi | x \in P(\phi)\}} f_\phi \leq \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \text{ for each link } x \tag{3.2b}$$

$$\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1 \tag{3.2c}$$

$$0 \leq \alpha_{\mathbf{v}} \text{ for each } \mathbf{v} \in \mathcal{V}, \tag{3.2d}$$

where $\mathbf{f}$ is the vector of flow rates. The function G is referred to as the *throughput metric*. Several different throughput metrics are possible. In some cases, the throughput metric is the network utility $G(\mathbf{f}) = \sum_{\phi\in\Phi} U_\phi(f_\phi)$, where U_ϕ is the utility function for flow ϕ. Popular utility functions include $U_\phi(f) = w_\phi \log(f)$ [9, 66, 67] and $U_\phi(f) = w_\phi f^{1-\gamma}/(1-\gamma)$ [68], where w_ϕ is the administrative weight. Another widely used throughput metric is $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$ [5]. This work specifically focuses on the cases when $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$ and $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$. In the case that $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$, the objective function is continuously differentiable, concave, and increasing. The solvability of such problems is detailed in [69]. If $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$, then (3.2) can be written as a linear programming problem, which is extensively studied in [70].

In theory, (3.2) is solvable. However, there is a significant computational challenge in that if $\mathcal{V}$ is the set of all assignments, then the vector $\boldsymbol{\alpha}$ has 2^L elements. Thus, the size of the space over which the optimization is performed must be reduced. This idea of considering a reduced space was considered in [5] and [6], however, the space was constructed arbitrarily. In this work the space is constructed so that the throughput found by optimizing over the reduced space is the same throughput found by optimizing over the entire space.

3.3 Optimal Scheduling

3.3.1 Introduction

The objective of this section is to compute optimal schedules by optimizing over a set of considered assignment $\mathcal{V} \subsetneq \overline{\mathcal{V}}$. The key questions are 1). is it possible to reduce the size of $\mathcal{V}$ without impacting the solution, and 2). if so, how can the set of considered assignments be constructed so that the value of (3.2) with the reduced sized $\mathcal{V}$ is the same or near to the value when $\mathcal{V} = \overline{\mathcal{V}}$? The answer to the first question is provided next and the following subsections focus on the second question.

Theorem 1 *There exists $\mathcal{V}$ with L assignments such that the solution to (3.2) is the same as the solution to (3.2) when $\mathcal{V} = \overline{\mathcal{V}}$.*

Proof. The optimal average data rate over each link is a convex sum of the links rates from different assignments, that is, the optimal bit-rate over link x is $\sum_{\mathbf{v}\in\mathcal{V}} \alpha^*_{\mathbf{v}} R(\mathbf{v}, x)$, where α^* defines the optimal schedule. In other words, the set of feasible link bit-rates is a convex set where the extreme points are some of the rows of R. Obviously, the vector of optimal link rates is the vector $\sum_{\mathbf{v}\in\mathcal{V}} \alpha^*_{\mathbf{v}} R(\mathbf{v}, :) \in \mathbb{R}^L$, the space of vectors with L elements. Due to Caratheodory's Theorem (e.g., Theorem B.6 in [69]), a point within a convex hull in $\mathbb{R}^L$ is specified by at most $L+1$ extreme points. That is, there exists a set, $\mathcal{V}'$ with $L+1$ elements such that

$$\sum_{\mathbf{v}\in\mathcal{V}} \alpha^*_{\mathbf{v}} R(\mathbf{v}, :) = \sum_{\mathbf{v}\in\mathcal{V}'} \alpha'_{\mathbf{v}} R(\mathbf{v}, :),$$

where $\boldsymbol{\alpha}'$ might be different set of weights from $\boldsymbol{\alpha}^*$. Hence, the optimal link bit-rates found by optimizing over $\overline{\mathcal{V}}$, the set of all possible assignments, can be achieved by only using the set of assignments $\mathcal{V}'$. Thus, the resulting utility is unchanged when $\mathcal{V}'$ is used as opposed to $\overline{\mathcal{V}}$.

Now it is shown that $\mathcal{V}'$ can be selected so that $\mathcal{V}'$ has less than $L+1$ elements. Suppose otherwise, that is, $\mathcal{V}'$ has exactly $L+1$ elements, and $\mathcal{V}'$ is the smallest set such that the optimal schedule is in $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$, the convex hull of $\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\}$. Since the faces of $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$ are defined by no more than L extreme points, the assumption that the optimal bit-rates cannot be specified by L points implies that the optimal bit-rates must be in the interior of $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$. That is, there is an open set that contains the optimal point and this open set is in the interior of $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$. For example, letting $\mathbf{r}^{**}$ be the vector of optimal bit-rates, the vector $\mathbf{r}^{**} + \varepsilon\mathbf{r}^{**}$ is also in the interior of $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$, where $\varepsilon > 0$ is small enough. Since $\mathbf{r}^{**}$ is the optimal vector of bit-rates over the interior of $Co(\{R(\mathbf{v}, :) : \mathbf{v} \in \mathcal{V}'\})$, the utility of

$\mathbf{r}^{**}$ must be higher than the utility of $\mathbf{r}^{**} + \varepsilon\mathbf{r}^{**}$. However, this is a contradiction since the link bit-rates $\mathbf{r}^{**} + \varepsilon\mathbf{r}^{**}$ result in uniformly large flow rates than $\mathbf{r}^{**}$, which will increase the throughput. Hence, $\mathcal{V}'$ can be selected to have fewer than $L + 1$ elements. ■

The result, which follows from Caratheodory's Theorem (e.g., Theorem B.6 in [69]), implies that the optimal schedule can be found by considering a set, $\mathcal{V}$, that is relatively small.

3.3.2 Basics

It is well known that Lagrange multiplier theory can be applied to (3.2) (e.g., see [69]). Specifically, associated with each link constraint (3.2b) is a Lagrange multiplier denoted by μ_x, with $\boldsymbol{\mu}$ being the vector of such multipliers. Similarly, associated with the constraint (3.2c) is a Lagrange multiplier denoted by λ. Employing the economic interpretation of Lagrange multipliers, μ_x can be interpreted as the price/bit of sending data over links x, or from the network's point of view, μ_x is the revenue that is collected for each bit that crosses link x. Under this interpretation, the revenue generated by assignment $\mathbf{v}$ is

$$\sum_{x=1}^{L} R(\mathbf{v}, x)\, \mu_x.$$

The multiplier λ can be interpreted as the maximum revenue generated by any assignment in $\mathcal{V}$.

Theorem 2 *Let* $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$ *or* $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$ *and let* μ_x *be the Lagrange multiplier associated with constraint (3.2b) and let* λ *be the Lagrange multiplier associated with constraint (3.2c), then*

$$\lambda = \max_{\mathbf{v}\in\mathcal{V}} \sum_{x=1}^{L} R(\mathbf{v}, x)\, \mu_x. \tag{3.3}$$

Proof. The proofs here are based on the throughput metric $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$ and a single path routing. The extension to multipath and other throughput metrics is straightforward.

The relevant Lagrange function is

$$L(\mathbf{f}, \alpha, \boldsymbol{\mu}, \lambda) = -\sum_{\phi\in\Phi} w_\phi \log(f_\phi) + \lambda\left(\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} - 1\right)$$
$$+\sum_{x=1}^{L} \mu_x \left(\sum_{\{\phi: x\in P(\phi)\}} f_\phi - \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R(\mathbf{v}, x)\right). \tag{3.4}$$

After some manipulation, the dual function is found to be

$$q(\boldsymbol{\mu}, \lambda) = \inf_{\mathbf{f},\alpha\geq 0} -\sum_{\phi\in\Phi} w_\phi \log(f_\phi) - \lambda \tag{3.5}$$
$$+\sum_{x=1}^{L} \mu_x \sum_{\{\phi: x\in P(\phi)\}} f_\phi - \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \left(\sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x - \lambda\right).$$

We immediately note that if $\sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x - \lambda > 0$ for some x, then $q(\boldsymbol{\mu}, \lambda) = -\infty$. Hence, we restrict the domain of q, to be such that $\sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x - \lambda \leq 0$. On the other hand, when solving the dual problem, an objective is to maximize q with respect to λ. It is equivalent to minimizing λ over the domain $\sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x - \lambda \leq 0$. Thus,

$$\lambda^* = \max_{\mathbf{v}\in\mathcal{V}} \sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x, \tag{3.6}$$

proving Proposition 2. ■

Typically, there are many assignments in $\mathcal{V}$ that generate revenue λ. The set of such assignments is referred to as the set of active assignments and is denoted $\mathcal{V}^*$, i.e.,

$$\mathcal{V}^*(\boldsymbol{\mu}) := \left\{\mathbf{v} : \sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x = \max_{\mathbf{v}\in\mathcal{V}} \sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x\right\}. \tag{3.7}$$

From Theorem 1, the optimal schedule multiplexes between a set of no more than L assignments. These assignments are contained in the set $\mathcal{V}^*$.

Theorem 3 *If* $\mathbf{v} \notin \mathcal{V}^*$, *then* $\alpha_{\mathbf{v}} = 0$.

Proof. Furthermore, for this λ, the $\sum_{x=1}^{L} R(\mathbf{v}, x)\,\mu_x - \lambda < 0$ for $\mathbf{v} \notin \mathcal{V}^*$, thus, the infimum in (3.5) must have $\alpha_{\mathbf{v}} = 0$ for $\mathbf{v} \notin \mathcal{V}^*$. ■

Since the revenue generated by active assignments is λ, the optimal schedule also generates revenue λ. Specifically, let R_x^* be the optimal data rate across link x, that is

$$R_x^* := \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}}^* R(\mathbf{v}, x), \tag{3.8}$$

where α^* specifies the optimal schedule. Then, it can easily be shown that

$$\lambda = \sum_{x=1}^{L} R_x^* \mu_x.$$

3.3.3 Evaluating Candidate Assignments

A brute force approach to constructing a good set of assignments is to start with an arbitrary set of assignments, $\mathcal{V}$, select an assignment $\mathbf{v}^+ \notin \mathcal{V}$, and evaluate the resulting throughput with the set of assignments $\mathbf{v}^+ \cup \mathcal{V}$. However, this approach is computationally complex in that (3.2) must be repeatedly solved. Furthermore, it is not clear if the utility of $\mathbf{v}^+$ is only apparent when it is added to $\mathcal{V}$ along with a particular set of other assignments. Alternatively, the question of whether an assignment $\mathbf{v}^+ \notin \mathcal{V}$ will increase the throughput when the set of considered assignments is changed from $\mathcal{V}$ to $\mathbf{v}^+ \cup \mathcal{V}$ is answered by the following theorem.

Theorem 4 *For the set of assignments* $\mathcal{V}$, *let* $\boldsymbol{\mu}$ *and* λ *be the Lagrange multipliers associated with constraints (3.2b) and (3.2c) when (3.2) is solved with this* $\mathcal{V}$. *Now consider an assignment* $\mathbf{v}^+ \notin \mathcal{V}$, *the throughput provided by* $\mathbf{v}^+ \cup \mathcal{V}$ *is greater than that provided by* $\mathcal{V}$ *if and only if*

$$\sum_{x=1}^{L} R\left(\mathbf{v}^+, x\right) \mu_x - \lambda > 0. \tag{3.9}$$

Proof. We slightly modify (3.2) to

$$\begin{aligned} &\min G(\mathbf{f}) \\ &\text{subject to:} \sum_{\{\phi: x \in P(\phi)\}} f_\phi - \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \leq \rho_x \\ &\qquad \sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}} - 1 = A, \end{aligned}$$

so (3.2) is the case where $\boldsymbol{\rho} = 0$ and $A = 0$. We will denote the value of the optimal solution of the above problem as $G^*(\boldsymbol{\rho}, A)$. From sensitivity analysis (e.g., [69]), we have

$$\mu_x^* = \frac{\partial G^*(\rho, A)}{\partial \rho_x} \tag{3.10}$$

$$\lambda^* = \frac{\partial G^*(\rho, A)}{\partial A}. \tag{3.11}$$

Equation (3.10), implies that if the amount of bit-rate that is applied to link x is increased by a small amount ε, then the total utility will be increased by $\mu_x^* \varepsilon$. It is critical to note that in this analysis, the bit-rate applied to link x does not come at the expense of bit-rates of other links.

Now consider the multiplier, λ^*. The constraint $\sum_{\mathbf{v} \in \mathcal{V}} \alpha_{\mathbf{v}} = 1 + A$ can be interpreted as allowing the total bandwidth of size $1 + A$ to be shared among all assignments. Thus, if the bandwidth is increased from size 1 to size $1 + \varepsilon$, then the utility will be increased by $\lambda \varepsilon$. Similarly, if the bandwidth is decreased by ε, then the utility will be decreased by $\lambda \varepsilon$.

While the analysis above assumed that the extra bandwidth is allocated to link x without impacting the bit-rate of the other links, we now consider the more relevant problem where this extra assignment comes at the expense of other links. Specifically, if we allocate assignment $\mathbf{v}^+$ with ε of the bandwidth, then the total bandwidth allocated to the other assignments must be decreased by ε. In particular, let $\mathcal{V}' = \{\mathbf{v}_1, ...\mathbf{v}_N\}$ and when optimizing over the set of assignments $\mathcal{V}'$,

let the associated optimal bandwidth allocated to $\mathbf{v}_i$ be of size α_i^*, where, of course, $\sum_{i=1}^{N} \alpha_i^* = 1$. Now in order to allocate bandwidth ε to assignment $\mathbf{v}^+$, we adjust the allocation to $\alpha_i^+ = (1-\varepsilon)\,\alpha_i^*$, and hence the assignments $\{\mathbf{v}^+, \mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N\}$ are allocated bandwidths of width $\{\varepsilon, (1-\varepsilon)\,\alpha_1^*, (1-\varepsilon)\,\alpha_2^*, ..., (1-\varepsilon)\,\alpha_N^*\}$, respectively. Based on the discussion above, the change in utility is

$$\varepsilon \left(\sum_{x=1}^{L} R\left(\mathbf{v}^+, x\right) \mu_x^* - \lambda \right), \tag{3.12}$$

which is positive if (3.9) holds. ■

Corollary 5 *If Lagrange multipliers that result from optimizing over $\mathcal{V}$ are such that there does not exist an assignment that satisfies (3.9), then the schedule found by optimizing over $\mathcal{V}$ is optimal.*

Theorem 4 provides the main tool for constructing a good set of assignments. Invoking an economic interpretation of the Lagrange multipliers, Theorem 4 implies that an assignment $\mathbf{v}^+$ will increase the utility if it generates more revenue per second than any other assignments in the set $\mathcal{V}$.

Figure 3.1 provides a geometric view of Theorem 4. Under a slightly different problem formulation, [5] considered arbitrarily adding assignments to the set of considered assignments. Theorem 4 provides a more sophisticated technique to decide whether assignments should be added.

3.3.4 Algorithm to Maximize the Throughput

Based on Theorem 4, Algorithm 1 can be used to iteratively add assignments to $\mathcal{V}$ such that the added assignment satisfies (3.9). The intuition behind this algorithm is to compute the values of the Lagrange multipliers by solving (3.2) with $\mathcal{V} = \mathcal{V}(n)$, where $\mathcal{V}(n)$ is the set of assignments at the nth iteration. With these multipliers a new assignment is found that satisfies (3.9). If no such assignment exists, then Corollary 5 implies that the schedule is optimal. If an assignment

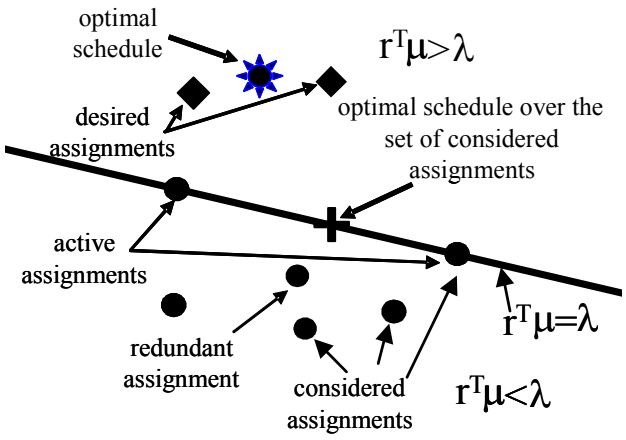


Figure 3.1: A geometric view of the optimal scheduling problem. The above shows the region of bit-rates, where we assume that there are only two links, and hence the space of bit-rates is the plane. The Lagrange multipliers found from optimizing over the set of considered assignments divide the space of bit-rates, $\mathbf{r}$, into two regions, according to whether $\mathbf{r}^T\boldsymbol{\mu} < \lambda$ or $\mathbf{r}^T\boldsymbol{\mu} > \lambda$. The active assignments and the schedule found by optimizing over the considered assignments are on the boundary of this division. An assignment will only improve the performance if $\mathbf{r}^T\boldsymbol{\mu} > \lambda$. The goal is to find the two desired assignments. The optimal schedule is a convex combination of these assignments.

Algorithm 1 Computing an Optimal Schedule

1: Set $n = 0$, then select an initial set of assignments $\mathcal{V}(0)$, a empty assignment $\mathbf{v}(0)$, and a level of accuracy $\rho \geq 0$.
2: **repeat**
3: Set $\mathcal{V}(n+1) = \mathcal{V}(n) \cup \mathbf{v}(n)$, and set $n = n + 1$.
4: Solve (3.2) with $\mathcal{V} = \mathcal{V}(n)$, and, hence, compute the flow rates $\mathbf{f}(n)$ and the Lagrange multiplies $\boldsymbol{\mu}(n)$ and $\lambda(n)$ associated with constraints (3.2b) and (3.2c), respectively.
5: Search for an assignment $\mathbf{v}(n)$ such that

$$\mathbf{v}(n) = \arg\max_{\mathbf{v}\in\overline{\mathcal{V}}} \sum_{x=1}^{L} R(\mathbf{v}, x)\, \mu_x(n) - \lambda(n). \tag{3.13}$$

6: **until**
7: **case** $G(\mathbf{f}) = min_{\phi\in\Phi} f_\phi$:
$\frac{\sum_{x=1}^{L} R(\mathbf{v}(n),x)\mu_x(n) - \lambda(n)}{G(\mathbf{f}(n))} < \rho$
8: **case** $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$:
$\sum_{x=1}^{L} R(\mathbf{v}(n), x)\, \mu_x(n) - \lambda(n) < L\log(1+\rho)$

that satisfies (3.9) is found, then we set to $\mathcal{V}(n+1)$ the union of $\mathcal{V}(n)$ and this newly found assignment. With this new set of assignments, (3.2) is resolved and an improvement of the resulting throughput is guaranteed by Theorem 4. This process is repeated until no new assignments can be found (in which case the optimal schedule has been found) or until the current solution is close enough to optimal. The first step of Algorithm 1 requires an initial set of assignments, $\mathcal{V}(0)$, which is given in Section 5.1. Also, Figure 3.2 shows the flow chart of Algorithm 1.

The following indicates the convergence of Algorithm 1.

Theorem 6 *Let* $\{(\mathbf{f}(n), \boldsymbol{\alpha}(n)) | n = 1, 2, ...\}$ *be the sequence of solutions given by Algorithm 1 with* $\rho = 0$. *Then* $\lim_{n\to\infty} (\mathbf{f}(n), \boldsymbol{\alpha}(n)) = (\mathbf{f}(\infty), \boldsymbol{\alpha}(\infty))$, *the optimal solution of (3.2) when* $\mathcal{V} = \overline{\mathcal{V}}$.

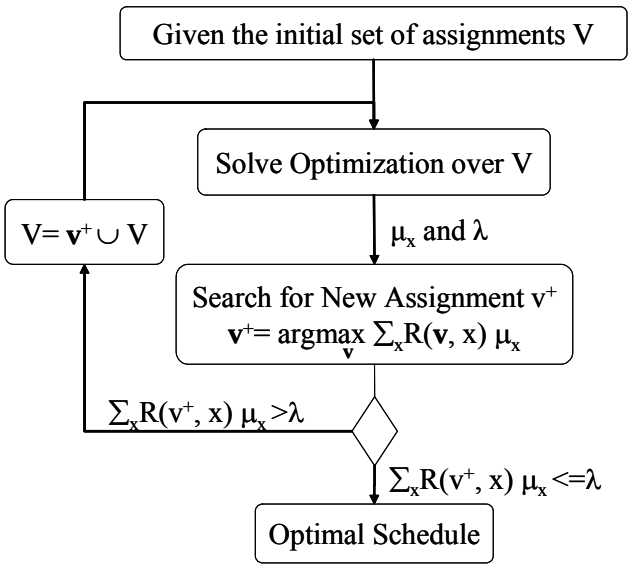


Figure 3.2: Algorithm for optimal scheduling

Theorem 7 *If* $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$, *then* $(G(\mathbf{f}(\infty)) - G(\mathbf{f}(n+1)))/(G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))) < \delta$ *for some constant* $0 < \delta < 1$, *that is,* $G(\mathbf{f}(n))$ *converges to* $G(\mathbf{f}(\infty))$ *geometrically fast.*

Theorem 8 *Let* $A \in \{0,1\}^{|\Phi|\times L}$ *with* $A(\phi, x) = 1$ *if* $x \in P(\phi)$ *and* $A(\phi, x) = 0$ *otherwise. Suppose that the null space of* A *is empty. Then Algorithm 1 with* $\rho = 0$ *converges arithmetically when* $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$, *that is,* $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n)) \leq a/n$ *for some* $a > 0$.

The proofs of these three theorems are in Section 3.5. The condition that A has an empty null space is satisfied if no two links have the exact same set of flows crossing them. In the case that the same set of flows do cross two different links, it is often the case that only one of these links will be a bottleneck and hence μ_x will be zero for the non-bottleneck link. Under this restriction, it is straightforward to show that the conclusion of Theorem 8 holds.

Theorem 9 *Suppose that Algorithm 1 terminates after n^* iterations with $\rho > 0$. If $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$, then $\frac{G(\mathbf{f}(\infty))-G(\mathbf{f}(n^*))}{G(\mathbf{f}(\infty))} < \rho$. Whereas, if $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$, then $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n^*)) = \sum_{\phi\in\Phi} \log(f_\phi(\infty)/f_\phi(n^*)) < L\log(1+\rho)$.*

This theorem is proved in Section 3.5. In our computational experiments we found that Algorithm 1 suffers from numerical issues that reduce the rate of convergence when n and L are large. It appears that solving (3.2) when $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$ is the source of these numerical problems. Thus, when L is large and $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$, we recommend $\rho > 0$. Specifically, when $L \leq 512$, we have found that $\rho = 0.05$ works well, but for $L > 512$, we use $\rho = 0.15$. It is hoped that improvements in solving (3.2) for large L will allow smaller values of ρ.

3.4 Summary

Optimal scheduling can be modeled as an optimization problem and the optimization space is exponential in the number of links which is 2^L where L is the number of links. In fact, it has been proved that the optimal schedule is the linear combination of L assignments. This chapter presents an iterative approach to find these special assignments. The approach is to start with a particular set of assignments, $\mathcal{V}$, select an assignment $\mathbf{v}^+ \notin \mathcal{V}$, and evaluate the resulting utility with the set of assignments $\mathbf{v}^+ \cup \mathcal{V}$. Specifically, at each iteration, a linear test (3.9) is provided to efficiently determine whether an assignment should be added to the set of considered assignments.

Also, this chapter shows the iterative approach has good convergence property. When $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$, the algorithm achieves algebraic convergence. When $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$, the algorithm achieves geometrical convergence.

3.5 Appendix

To simplify the notation, only a single path routing is considered. The extension to multipath and other throughput metrics is straightforward.

Algorithm 2

1: Set $k = 0$, then select an initial set of assignments $\mathcal{V}(0)$ and a empty assignment $\mathbf{v}(0)$.
2: **repeat**
3: Set $\mathcal{V}(k+1) = \mathcal{V}(k) \cup \mathbf{v}(k)$, and set $k = k+1$.
4: Find $\boldsymbol{\mu}(k)$ and $\lambda(k)$, the solutions to (3.14) for $\mathcal{V} = \mathcal{V}(k)$.
5: Find $\mathbf{v}(k) = \arg\max_{\mathbf{v}\in\overline{\mathcal{V}}} \sum_{x=1}^{L} \mu_x(k) R(\mathbf{v}, x) - \lambda(k)$.
6: **until** $\sum_{x=1}^{L} \mu_x(k) R(\mathbf{v}(k)) - \lambda(k) \leq 0$

3.5.1 Proof of Theorem 6

Consider the problem

$$\begin{aligned} &\max g(\boldsymbol{\mu}, \lambda) \\ &\text{subject to } \sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x \leq \lambda \text{ for all } \mathbf{v} \in \mathcal{V} \\ &\qquad h(\boldsymbol{\mu}) = 0 \end{aligned} \tag{3.14}$$

and Algorithm 2 for solving this problem with $\mathcal{V} = \overline{\mathcal{V}}$.

This problem is the dual of (3.2) for either objective function by correctly defining g and h. Specifically, if $g(\boldsymbol{\mu}, \lambda) = \lambda$ and $h(\boldsymbol{\mu}) = \sum_x \mu_x \beta_x - 1$, then (3.14) is the dual of (3.2) when $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$. On the other hand, if $g(\boldsymbol{\mu}, \lambda) = -\sum_{\phi\in\Phi} \log\left(\frac{1}{\sum_{x\in P(\phi)} \mu_x}\right) + \sum_{x=1}^{L} \mu_x \sum_{\{\phi: x\in P(\phi)\}} \frac{1}{\sum_{y\in P(\phi)} \mu_y} - \lambda$ and $h(\boldsymbol{\mu}) \equiv 0$, then (3.14) is the dual of (3.2) with $G(f) = \sum_{\phi\in\Phi} \log(f_\phi)$. Thus the following theorem applies to both cases.

Specifically, cutting plane method is deployed to solve the dual problem and Figure 3.3 shows the idea of Algorithm 2. The ellipse is the domain defined by all assignments $\overline{\mathcal{V}}$, and the triangle is the domain defined by the current set of assignments $\mathcal{V}$. First, find the optimal $(\boldsymbol{\mu}, \lambda)$ by solving the dual problem in the current set $\mathcal{V}$, and project the optimal to the ellipse domain. Therefore, a new assignment is added to the current set of assignments $\mathcal{V}$. Then, find the optimal $(\boldsymbol{\mu}, \lambda)$ in $\mathcal{V}$ again. We keep on doing this until the optimal is found in the ellipse domain, which denotes the optimal of the dual problem for $\overline{\mathcal{V}}$.

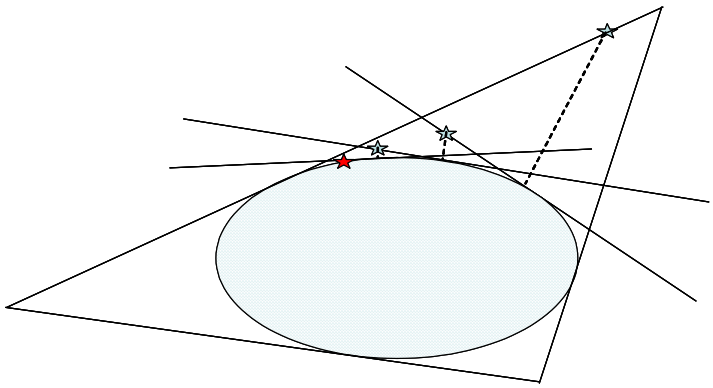


Figure 3.3: Cutting plane method for dual problem

Theorem 10 *The sequence* $\{(\boldsymbol{\mu}(n), \lambda(n)) | n = 0, 1, ...\}$ *given by Algorithm 2 converges to the optimal solution. Thus, Algorithm 1 converges to the optimal solution.*

Lemma 11 *Assume that* $G(\mathbf{f}) = \sum_{\phi \in \Phi} \log(f_\phi)$. *Then the set* $\{(\boldsymbol{\mu}(n), \lambda(n)) | n = 1, 2, ...\}$ *is a bounded set.*

Proof. We first get a lower bound on the optimal value of $G(\mathbf{f})$. Let $\mathbf{v}_x$ be the assignment where link x is active individually. Set $\underline{f} = \min_\phi \min_{x \in P(\phi)} \frac{R(\mathbf{v}_x, x)}{\beta_x}$. Then $\underline{G} = \sum_{\phi \in \Phi} \log\left(\underline{f}\right)$ is a lower bound on $G(\mathbf{f}(\infty))$. Let r^* be the maximum data rate over any link. Then $\overline{G} = \sum_{\phi \in \Phi} \log(r^*)$ is an upper bound on $G(\mathbf{f}(\infty))$. Thus, for a flow θ, we must have that $\underline{G} < \log(f_\theta^*) + \sum_{\phi \in \Phi \setminus \theta} \log(r^*)$. Hence $f_\theta^* \geq \exp\left(\underline{G} - \sum_{\phi \in \Phi \setminus \theta} \log(r^*)\right)$. Since $f_\theta^* = 1/\sum_{x \in P(\theta)} \mu_x^*$ and $\mu_x^* \geq 0$, we have $\mu_x^* < 1/\exp\left(\underline{G} - \sum_{\phi \in \Phi \setminus \theta} \log(r^*)\right)$. Moreover, since $\sum_{x=1}^{L} R(\mathbf{v}, x) \mu_x^* = \lambda^*$, we must have that $\lambda^* \leq L r^* / \exp\left(\underline{G} - \sum_{\phi \in \Phi \setminus \theta} \log(r^*)\right)$. ■

Lemma 12 *Assume that* $G(f) = \min_{\phi \Phi}(f_\phi)$. *Then the set* $\{(\boldsymbol{\mu}(n), \lambda(n)) | n = 1, 2, ...\}$ *is a bounded set.*

Proof. Let $\sum_{x=1}^{L} \mu_x^* \beta_x = F^* \leq r^*$ where r^* is the maximum data rate over any link. Thus, $\mu_x \leq r^*/\beta_x$.Also, $\lambda^* \leq r^*$. ■

Proof. [Proof of Theorem 10] Since $\{(\boldsymbol{\mu}(n), \lambda(n)) : n = 1, 2, ...\}$ is a bounded sequence, there exists a convergent subsequence. Thus, assume that $\{(\boldsymbol{\mu}(n_j), \lambda(n_j)) : j = 1, 2, ...\}$ is such a sequence and $\lim_{j\to\infty}(\boldsymbol{\mu}(n_j), \lambda(n_j)) = (\boldsymbol{\mu}', \lambda')$. Define a sequence of sets of assignments $\mathcal{V}(n_j) := \mathcal{V}(0) \cup \bigcup_{i=1}^{n_j} \mathbf{v}(i)$. We will show that $\sum_{x=1}^{L} R(\mathbf{v}, x)\mu'_x < \lambda'$ for $\mathbf{v} \in \mathcal{V}(n)$. To this end define $u(\boldsymbol{\mu}, \lambda) := \max_{\mathbf{v}\in\overline{\mathcal{V}}} \sum_{x=1}^{L} R(\mathbf{v}, x)\mu_x - \lambda$. It is straightforward to check that u is a continuous function. Also, $u(\boldsymbol{\mu}(n_j), \lambda(n_j)) = \sum_{x=1}^{L} R(\mathbf{v}(n_j), x)\mu_x(n_j) - \lambda(n_j)$, and since $\mathcal{V}(n_j)$ is increasing in j, $\sum_{x=1}^{L} R(\mathbf{v}, x)\mu'_x - \lambda' \leq 0$ for all $\mathbf{v} \in \mathcal{V}(n_j)$ for all j. Therefore, the following string holds

$$
\begin{aligned}
& u((\boldsymbol{\mu}', \lambda')) \\
= \; & u((\boldsymbol{\mu}(n_j), \lambda(n_j))) + u((\boldsymbol{\mu}', \lambda')) \\
& -u((\boldsymbol{\mu}(n_j), \lambda(n_j))) \\
= \; & \sum_{x=1}^{L} R(\mathbf{v}(n_j), x)\mu_x(n_j) - \lambda(n_j) \\
& + (u((\boldsymbol{\mu}', \lambda')) - u((\boldsymbol{\mu}(n_j), \lambda(n_j)))) \\
\leq \; & \left(\sum_{x=1}^{L} R(\mathbf{v}(n_j), x)\mu_x(n_j) - \lambda(n_j)\right) \\
& - \left(\sum_{x=1}^{L} R(\mathbf{v}(n_j), x)\mu'_x - \lambda'\right) \\
& + (u((\boldsymbol{\mu}', \lambda')) - u((\boldsymbol{\mu}(n_j), \lambda(n_j)))) \\
= \; & \left(\sum_{x=1}^{L} R(\mathbf{v}(n_j), x)(\mu_x(n_j) - \mu'_x) - (\lambda(n_j) - \lambda')\right) \\
& + (u((\boldsymbol{\mu}', \lambda')) - u((\boldsymbol{\mu}(n_j), \lambda(n_j)))).
\end{aligned}
$$

Since the entries of R are bounded and since u is continuous, the right-hand side converges to zero as $j \to \infty$. Therefore, $u((\boldsymbol{\mu}', \lambda')) \leq 0$.

Note that $g((\boldsymbol{\mu}(n_j), \lambda(n_j)))$ is a nonincreasing function (since more constraints are added at each iteration) and $g((\boldsymbol{\mu}(n_j), \lambda(n_j))) \geq g((\boldsymbol{\mu}(\infty), \lambda(\infty)))$,

where $(\boldsymbol{\mu}(\infty), \lambda(\infty))$ is the solution to (3.14) for $\mathcal{V} = \overline{\mathcal{V}}$. Since g is continuous, $\lim_{j\to\infty} g((\boldsymbol{\mu}(n_j), \lambda(n_j))) = g((\boldsymbol{\mu}', \lambda')) \leq g((\boldsymbol{\mu}(\infty), \lambda(\infty)))$. Thus $(\boldsymbol{\mu}', \lambda')$ solves (3.14) for $\mathcal{V} = \overline{\mathcal{V}}$. And hence, $(\boldsymbol{\mu}', \lambda')$ solves the dual of (3.14), which has a unique solution, hence $(\boldsymbol{\mu}(\infty), \lambda(\infty)) = (\boldsymbol{\mu}', \lambda')$. Thus, all subsequences of $\{(\boldsymbol{\mu}(n), \lambda(n)) : j = 1, 2, ...\}$ converge to $(\boldsymbol{\mu}', \lambda')$. Hence, $\lim_{n\to\infty}(\boldsymbol{\mu}(n), \lambda(n)) = (\boldsymbol{\mu}', \lambda')$. It is straightforward to show that if $\{(\mathbf{f}(n), \boldsymbol{\alpha}(n)) : n = 1, 2, ...\}$ is the sequence of solutions to the primal problems via Algorithm 1, then $\lim_{n\to\infty}(\mathbf{f}(n), \boldsymbol{\alpha}(n)) = (\mathbf{f}(\infty), \boldsymbol{\alpha}(\infty))$, which is the optimal solution to the primal problem. ■

3.5.2 Proof of Theorems 7 and 9

Define $\Delta(n) = \arg\max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v}, :)\boldsymbol{\mu}(n) - \lambda(n)$. Define $f(n)$ to be the vector of flow rates found during the nth iteration of Algorithm 1 and let $f(\infty) := \lim_{n\to\infty} f(n)$. Thus $G(\mathbf{f}(n))$ to be the optimal value of (3.2) after the nth iteration. And let $G(\mathbf{f}(\infty))$ be the solution to the full problem. Define $\Delta\lambda_n = \lambda(n+1) - \lambda(n)$ and $\boldsymbol{\Delta\mu}(n) = \boldsymbol{\mu}(n+1) - \boldsymbol{\mu}(n)$. Let $\mathbf{v}(n) = \arg\max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v}, :)\boldsymbol{\mu}(n)$. That is, $\mathbf{v}(n)$ is the assignment added at the nth iteration.

Theorem 13 *Let* $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$, *then for* $\mathbf{f}(n)$ *found by Algorithm 1,* $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n)) \leq \Delta(n)$.

Proof. The dual of (3.2) is

$$\max \min_{\mathbf{f}} -\sum_{\phi} \log(f_\phi) + \sum_{x} \mu_x \sum_{\{\phi: x\in P(\phi)\}} f_\phi - \lambda$$
$$\text{subject to } \sum_{x} R(\mathbf{v}, x)\mu_x \leq \lambda \text{ for all } \mathbf{v} \in \bar{\mathcal{V}}.$$

Then as above, $((\boldsymbol{\mu}(n), \max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v}, :)\boldsymbol{\mu}(n)))$ is a feasible (but not optimal) solution to the dual of the full problem. Since there is no gap between the primal and

dual optimal, $\sum_x \mu_x \sum_{\{\phi: x \in P(\phi)\}} f_\phi$ is equal to $\lambda(n)$. After substitute the feasible solution to the cost function, we get

$$-G\left(\mathbf{f}\left(n\right)\right) + \lambda\left(n\right) - \max_{\mathbf{v}\in\bar{\mathcal{V}}} R\left(\mathbf{v},:\right)\boldsymbol{\mu}\left(n\right).$$

Nonetheless, $-G\left(\mathbf{f}\left(\infty\right)\right)$ is the optimal of the primal problem. Therefore,

$$-G\left(\mathbf{f}\left(\infty\right)\right) \geq -G\left(\mathbf{f}\left(n\right)\right) + \lambda\left(n\right) - \max_{\mathbf{v}\in\bar{\mathcal{V}}} R\left(\mathbf{v},:\right)\boldsymbol{\mu}\left(n\right).$$

Then,

$$G\left(\mathbf{f}\left(\infty\right)\right) - G\left(\mathbf{f}\left(n\right)\right) \leq \max_{\mathbf{v}\in\bar{\mathcal{V}}} R\left(\mathbf{v},:\right)\boldsymbol{\mu}\left(n\right) - \lambda\left(n\right).$$

■

To proof the above for the case when $G\left(\mathbf{f}\right) = \min_{\phi\in\Phi} f_\phi$ we rewrite (3.2) to

$$\begin{aligned} & \min -F \\ \text{subject to:} \quad & \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R\left(\mathbf{v},x\right) \geq \beta_x F \\ & \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1 \end{aligned} \tag{3.15}$$

where β_x is the number of flows that pass through link x divided by the bit-rate of link x. Thus, with normalization, $R\left(\mathbf{v},x\right) \in \{0,1\}$.

Theorem 14 *Let* $G\left(\mathbf{f}\right) = \min_{\phi\in\Phi} f_\phi$, *then for* $\mathbf{f}\left(n\right)$ *found by Algorithm 1,* $G\left(\mathbf{f}\left(\infty\right)\right) - G\left(\mathbf{f}\left(n\right)\right) \leq \Delta\left(n\right)$

Proof. The dual of (3.2) with $\mathcal{V} = \bar{\mathcal{V}}$ is

$$\begin{aligned} & \min \lambda \\ -R\left(\mathbf{v},:\right)\boldsymbol{\mu} \quad & \geq \quad -\lambda \text{ for all } \mathbf{v} \in \bar{\mathcal{V}} \\ \sum_x \mu_x \beta_x \quad & \geq \quad 1. \end{aligned}$$

Note that $(\boldsymbol{\mu}(n), \max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v},:)\boldsymbol{\mu}(n))$ is a feasible (but not optimal) solution to the dual of the full problem. Thus, $G(\mathbf{f}(\infty)) \leq \max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v},:)\boldsymbol{\mu}(n)$ and $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n)) \leq \max_{\mathbf{v}\in\bar{\mathcal{V}}} R(\mathbf{v},:)\boldsymbol{\mu}(n) - G(\mathbf{f}(n)) = \Delta(n)$. ■

Theorem 9 follows from Theorems 13 and 14. We now focus on proving Theorem 7. To this end, the following lemmas are proved.

Lemma 15 *$\Delta\lambda(n) \geq R(\mathbf{v},:)\boldsymbol{\Delta\mu}(n)$ for all $\mathbf{v} \neq \mathbf{v}(n)$ and $\mathbf{v} \in \mathcal{V}^*(n+1)$*

Proof. From the identity, $\lambda(n) = \max_{\mathbf{v}\in\mathcal{V}(n)} R(\mathbf{v},:)\boldsymbol{\mu}(n) = R(\mathbf{v},:)\boldsymbol{\mu}(n)$ for $\mathbf{v} \in \mathcal{V}^*(n)$, we have for $\mathbf{v} \in \mathcal{V}^*(n+1)$ and $\mathbf{v} \neq \mathbf{v}(n)$

$$\begin{aligned} & \lambda(n+1) - \lambda(n) \\ = \; & R(\mathbf{v},:)\boldsymbol{\mu}(n+1) - \max_{\mathbf{v}\in\mathcal{V}(n)} R(\mathbf{v},:)\boldsymbol{\mu}(n) \\ \geq \; & R(\mathbf{v},:)\boldsymbol{\mu}(n+1) - R(\mathbf{v},:)\boldsymbol{\mu}(n). \end{aligned}$$

■

Lemma 16 *$\Delta\lambda(n) = R(\mathbf{v}(n),:)\boldsymbol{\Delta\mu}(n) + \Delta(n)$*

Proof. The newly added assignment, $\mathbf{v}(n)$, is always an active assignment in the schedule found in the $(n+1)$th iteration, i.e., $\mathbf{v}(n) \in \mathcal{V}^*(n+1)$. Thus, $\lambda(n+1) = R(\mathbf{v}(n),:)\boldsymbol{\mu}(n+1)$. From the definition of $\Delta(n)$, we have $-\lambda(n) = -R(\mathbf{v}(n),:)\boldsymbol{\mu}(n) + \Delta(n)$. Thus, $\lambda(n+1) - \lambda(n) = R(\mathbf{v}(n),:)\boldsymbol{\mu}(n+1) - R(\mathbf{v}(n),:)\boldsymbol{\mu}(n) + \Delta(n)$. ■

Lemma 17 *$\boldsymbol{\beta}^T\Delta\boldsymbol{\mu}(n) = 0$.*

Proof. The Lagrangian for (3.15) is

$$\begin{aligned} L(\boldsymbol{\mu},\lambda,\boldsymbol{\alpha},F) = \; & -F + \sum_x \mu_x \left(\beta_x F - \sum_{\mathbf{v}} \alpha_{\mathbf{v}} R(\mathbf{v},x)\right) \\ & +\lambda\left(\sum_{\mathbf{v}} \alpha_{\mathbf{v}} - 1\right). \end{aligned}$$

Since F appears linearly, for the optimal value of $\boldsymbol{\mu}$ we must have that $-1 + \sum_x \beta_x \mu_x = 0$. Thus, for all n, $\boldsymbol{\beta}^T \boldsymbol{\mu}(n) = 1$, and hence $\boldsymbol{\beta}^T \Delta\boldsymbol{\mu}(n) = 0$. ■

Note that the set of active assignments $\mathcal{V}^*(n+1)$ and the corresponding matrix of data-rates must be schedulable in the sense that sufficient data must flow on each link. This gives rise to following condition on $\mathcal{V}^*(n+1)$ and R.

There exists a vector $\boldsymbol{\alpha}$ with $\sum_{\mathbf{v}\in\mathcal{V}^*(n+1)} \alpha_{\mathbf{v}} = 1$ and $\alpha_{\mathbf{v}} > 0$ for all $\mathbf{v} \in \mathcal{V}^*(n+1)$ such that there exists a $t > 0$ such that

$$\boldsymbol{\alpha} R(:, x) \geq t\beta_x \text{ for all } x. \tag{3.16}$$

Lemma 18 *There exists a $q > 0$ that is independent of $\mathcal{V}^*(n+1)$ such that $\max_{\mathbf{v}\in\mathcal{V}^*(n+1)\setminus\mathbf{v}(n)} R(\mathbf{v}, :)\Delta\boldsymbol{\mu}(n) \geq -qR(\mathbf{v}(n), :)\Delta\boldsymbol{\mu}(n)$.*

Proof. Multiplying both sides of (3.16) by $\Delta\mu_x(n)$ and summing results in

$$\sum_{x=1}^{L} \Delta\mu_x(n) \sum_{\mathbf{v}\in\mathcal{V}^*(n+1)} \alpha_{\mathbf{v}} R(\mathbf{v}, x) \geq t \sum_{x=1}^{L} \beta_x \Delta\mu_x(n)$$

and from Lemma 17, we have

$$\sum_{\mathbf{v}\in\mathcal{V}^*(n+1)\setminus\mathbf{v}(n)} \alpha_{\mathbf{v}} \sum_x \Delta\mu_x(n) R(\mathbf{v}, x) \quad + \quad \alpha_{\mathbf{v}(n)} \Delta\mu_x(n) R(\mathbf{v}(n), x) \qquad \geq \qquad 0.$$

Thus,

$$\max_{\mathbf{v}\in\mathcal{V}^*(n+1)\setminus\mathbf{v}(n)} R(\mathbf{v}, :)\Delta\boldsymbol{\mu}(n) \qquad \geq \qquad -q(R, \boldsymbol{\beta}) R(\mathbf{v}(n), x)\Delta\mu_x(n)$$

where $q(R, \boldsymbol{\beta}) > 0$ is a constant that depends on the vector $\boldsymbol{\alpha}$ given by Condition 3.5.2, and hence depends on the matrix of active assignments $\mathcal{V}^*(n+1)$ and the vector $\boldsymbol{\beta}$. The set of active assignments is in the set $\mathrm{V}(\boldsymbol{\beta}, L)$ where

$$\mathcal{V}(\boldsymbol{\beta}, L) := \left\{ \{0,1\}^{s\times L} \mid s \leq L, \text{ Condition 3.5.2 holds} \right\},$$

where s is the number of active assignments, and $s \leq L$. Clearly, $\mathrm{V}(\boldsymbol{\beta}, L)$ is a compact set (actually, it is a finite set). Hence, there exists a $q := \min_{R \in \mathcal{V}(\boldsymbol{\beta}, L)} q(R, \boldsymbol{\beta})$ where $q > 0$ and thus $\max_{\mathbf{v} \in \mathcal{V}^*(n+1) \setminus \mathbf{v}(n)} R(\mathbf{v}, :) \boldsymbol{\Delta\mu}(n) \geq -q \Delta\mu_x(n) R(\mathbf{v}(n), x)$ for any $\mathcal{V}^*(n+1) \in \mathrm{V}(\beta, L)$. ■

Lemma 19 *$\Delta\lambda(n) \geq \delta \Delta(n)$ for some $\delta > 0$.*

Proof. From Lemmas 15, 16, and 18

$$\begin{aligned}
\Delta\lambda(n) &\geq \max_{\mathbf{v} \in \mathcal{V}^*(n+1) \setminus \mathbf{v}(n)} R(\mathbf{v}, :) \Delta\boldsymbol{\mu}(n) \\
&\geq -qR(\mathbf{v}(n), :) \Delta\boldsymbol{\mu}(n) \\
&= q(\Delta(n) - \Delta\lambda(n)) \\
\Delta\lambda(n)(1+q) &\geq q\Delta(n) \\
\Delta\lambda(n) &\geq \frac{q}{q+1} \Delta(n).
\end{aligned}$$

■

Proof. [Proof of Theorem 7] Since there is no duality gap, $\lambda(n) = G(\mathbf{f}(n))$. Thus, from Lemma 19

$$\frac{G(\mathbf{f}(n+1)) - G(\mathbf{f}(n))}{\Delta(n)} \geq \delta.$$

On the other hand, by Theorem 14, $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n)) \leq \Delta(n)$. Therefore, we have

$$\begin{aligned}
&\frac{G(\mathbf{f}(n+1)) - G(\mathbf{f}(n))}{G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))} \geq \delta \\
&\frac{(G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))) - (G(\mathbf{f}(\infty)) - G(\mathbf{f}(n+1)))}{G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))} \geq \delta \\
&1 - \frac{(G(\mathbf{f}(\infty)) - G(\mathbf{f}(n+1)))}{G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))} \geq \delta \\
&\frac{(G(\mathbf{f}(\infty)) - G(\mathbf{f}(n+1)))}{G(\mathbf{f}(\infty)) - G(\mathbf{f}(n))} \leq 1 - \delta < 1.
\end{aligned}$$

■

Note that in practice, numerical errors limit the accuracy of the solutions. These errors result in $G(\mathbf{f}(\infty)) - G(\mathbf{f}(n+1))$ approaching zero slowly for large n. Thus, if $\Delta_{Threshold}$ is very small, it may take many iterations before $\Delta(n) < \Delta_{Threshold}$. Thus, $\Delta_{Threshold}$ should not be to small. While further research is required to understand the source and impact of numerical errors, we suspect that errors in channel gain measurements and node synchronization result in more significant reduction in actual throughput than using a large value of $\Delta_{Threshold}$.

3.5.3 Proof of Theorem 8

Lemma 20 *$\|\boldsymbol{\mu}(n+1) - \boldsymbol{\mu}(n)\| \geq \delta |\lambda(n+1) - \lambda(n)|$ for some $\delta > 0$.*

Proof. Recall that $\lambda(n) = \max_{\mathbf{v} \in \mathcal{V}(n)} R(\mathbf{v},:)\boldsymbol{\mu}(n)$ and for $\mathbf{v} \in \mathcal{V}^*(n)$ we have $\lambda(n) = R(\mathbf{v},:)\boldsymbol{\mu}(n)$. Let $\mathbf{v}' \in \mathcal{V}^*(n+1) \cap \mathcal{V}(n)$. Then

$$R(\mathbf{v}',:)\boldsymbol{\mu}(n+1) - R(\mathbf{v}',:)\boldsymbol{\mu}(n) \geq \lambda(n+1) - \lambda(n).$$

Since $R(\mathbf{v}',x) \in \{0,1\}$, there exists an x such that $\mu_x(n+1) - \mu_x(n) \geq \frac{1}{L}(\lambda(n+1) - \lambda(n))$, and $\|\boldsymbol{\mu}(n+1) - \boldsymbol{\mu}(n)\| \geq \frac{1}{L}|(\lambda(n+1) - \lambda(n))|$. ■

Lemma 21 *Let $A \in \{0,1\}^{|\Phi| \times L}$ with $A(\phi,x) = 1$ if $x \in P(\phi)$ and $A(\phi,x) = 0$ otherwise. Suppose that the null space of A is empty. Then $A(\phi,:)\boldsymbol{\mu} = 1/f_\phi$ and there exists a $\delta > 0$ such that $\|\mathbf{f}(n) - \mathbf{f}(n+1)\| \geq \delta \|\boldsymbol{\mu}(n) - \boldsymbol{\mu}(n+1)\|$.*

Proof. Since the null space of A is empty, all the singular values of A are nonzero. Thus, $\left(\sum_{\phi \in \Phi}\left(\frac{1}{f_\phi(k)} - \frac{1}{f_\phi(k+1)}\right)^2\right)^{1/2} \geq \frac{1}{\underline{\sigma}}\|\boldsymbol{\mu}(k) - \boldsymbol{\mu}(k+1)\|$, where $\underline{\sigma}$ is the smallest singular value of A.

Recall that the proof of Lemma 11 showed that there exists a $\underline{f}$ such that $f_\phi(n) \geq \underline{f}$ for all n and ϕ. Direct calculation shows that $|f_\phi(n) - f_\phi(n+1)| \geq \underline{f}^2\left|\frac{1}{f_\phi(n)} - \frac{1}{f_\phi(n+1)}\right|$. Thus, $\|\mathbf{f}(n) - \mathbf{f}(n+1)\| \geq \frac{\underline{f}^2}{\underline{\sigma}}\|\boldsymbol{\mu}(k) - \boldsymbol{\mu}(k+1)\|$. ■

Combining the previous lemmas we get the following.

Lemma 22 *If the assumption of Lemma 21 holds, then*
$\|\mathbf{f}(n)-\mathbf{f}(n+1)\| \geq \delta \|(\boldsymbol{\mu}(n+1),\lambda(n+1))-(\boldsymbol{\mu}(n),\lambda(n))\|$ *for some* $\delta \geq 0$.

Proof. [Proof of Theorem 8] [1]Define $u(\boldsymbol{\mu},\lambda) := \max_{\mathbf{v}\in\overline{\mathcal{V}}} \sum_{x=1}^{L} R(\mathbf{v},x)\mu_x - \lambda$. Let $(\boldsymbol{\mu}(0),\lambda(0))$ be the multipliers that result from solving (3.2) with $\mathcal{V} = \{\mathbf{v} : v_x = 1$ for exact one $x\}$. Let $\lambda_o = \max_{\mathbf{v}\in\overline{\mathcal{V}}} \sum_{x=1}^{L} R(\mathbf{v},x)\mu_x(0)$. Then, $u((\boldsymbol{\mu}(0),2\lambda_o)) = -\lambda_o$. Thus, for each n, there exists a $\gamma(n) \in [0,1]$ such that

$$0 = u(\gamma(n)(\boldsymbol{\mu}(n),\lambda(n)) + (1-\gamma(n))(\boldsymbol{\mu}(0),2\lambda_o)). \tag{3.17}$$

From (3.17),

$$\begin{aligned} 0 &= u(\gamma(n)(\boldsymbol{\mu}(n),\lambda(n)) + (1-\gamma(n))(\boldsymbol{\mu}(0),2\lambda_o)) \\ &\leq \gamma(n)\,u(\boldsymbol{\mu}(n),\lambda(n)) + (1-\gamma(n))\,u(\boldsymbol{\mu}(0),2\lambda_o), \end{aligned}$$

where the inequality is implied by the convexity of u. Therefore,

$$u(\boldsymbol{\mu}(n),\lambda(n)) \geq -\frac{(1-\gamma(n))}{\gamma(n)} u(\boldsymbol{\mu}(0),2\lambda_o).$$

Since $\mathbf{v}(n) \in \mathcal{V}^*(n+1)$, we have $R(\mathbf{v}(n),:)\boldsymbol{\mu}(n+1) - \lambda(n+1) = 0$. Also, $u((\boldsymbol{\mu}(n),\lambda(n))) = R(\mathbf{v}(n),:)\boldsymbol{\mu}(n+1) - \lambda(n+1)$. Therefore,

$$\begin{aligned} & -\frac{(1-\gamma(n))}{\gamma(n)} u(\boldsymbol{\mu}(0),2\lambda_o) \\ \leq\ & u(\boldsymbol{\mu}(n),\lambda(n)) \\ =\ & R(\mathbf{v}(n),:)\boldsymbol{\mu}(n) - \lambda(n) \\ & - (R(\mathbf{v}(n),:)\boldsymbol{\mu}(n+1) - \lambda(n+1)) \\ =\ & R(\mathbf{v}(n),:)(\boldsymbol{\mu}(n) - \boldsymbol{\mu}(n+1)) - (\lambda(n) - \lambda(n+1)) \\ \leq\ & r^* \|(\boldsymbol{\mu}(n+1),\lambda(n+1)) - (\boldsymbol{\mu}(n),\lambda(n))\|, \end{aligned}$$

[1] This proof is based on a proof in [116].

where r^* is the highest data rate across any link, and hence $R(\mathbf{v}, x) \leq r^*$. From the above and Lemma 22 we have

$$\begin{aligned} \|\mathbf{f}(n) - \mathbf{f}(n+1)\| &\geq -\frac{\delta(1-\gamma(n))}{\gamma(n)} u(\boldsymbol{\mu}(0), 2\lambda_o) \\ &\geq -\delta(1-\gamma(n)) u(\boldsymbol{\mu}(0), 2\lambda_o) \end{aligned}$$

for some $\delta > 0$. Or

$$(1-\gamma(n)) \leq \frac{\|\mathbf{f}(n) - \mathbf{f}(n+1)\|}{-\delta u(\boldsymbol{\mu}(0), 2\lambda_o)}. \tag{3.18}$$

Corresponding to the point $(\boldsymbol{\mu}(0), 2\lambda_o)$, define flow rates $\tilde{\mathbf{f}}$ where $\tilde{f}_\phi = \frac{1}{\sum_{x \in P(\phi)} \mu_x(0)}$. Clearly, this set of data rates is suboptimal but feasible. Similarly, $\mathbf{f}(n)$ is suboptimal but feasible. Hence, $\gamma(n)\mathbf{f}(n) + (1-\gamma(n))\mathbf{f}$ is suboptimal but feasible. Therefore,

$$-G(\mathbf{f}(\infty)) \leq -G\left(\gamma(n)\mathbf{f}(n) + (1-\gamma(n))\tilde{\mathbf{f}}\right),$$

where $\mathbf{f}(\infty)$ is the vector of optimal flow rates. Then

$$\begin{aligned} & (-G(\mathbf{f}(\infty))) - (-G(\mathbf{f}(n))) \\ \leq\ & \left(-G\left(\gamma(n)\mathbf{f}(n) + (1-\gamma(n))\tilde{\mathbf{f}}\right)\right) - (-G(\mathbf{f}(n))) \\ \leq\ & K\left\|\gamma(n)\mathbf{f}(n) + (1-\gamma(n))\tilde{\mathbf{f}} - \mathbf{f}(n)\right\| \\ =\ & K(1-\gamma(n))\left\|\mathbf{f}(n) - \tilde{\mathbf{f}}\right\|, \end{aligned} \tag{3.19}$$

where $K = \max_{\mathbf{f} \in \left\{\mathbf{f} | \underline{f} \leq f_\phi \leq r^*\right\}} \|\nabla G(\mathbf{f})\|$ and $\nabla G(\mathbf{f})$ is the gradient of G at $\mathbf{f}$ and $\underline{f}$ is the lower bound on the flow rates given in Lemma 11.

Combining (3.18) and (3.19) yields,

$$\begin{aligned} & (-G(\mathbf{f}(\infty))) - (-G(\mathbf{f}(n))) \\ \leq\ & \frac{K}{-\delta u(\boldsymbol{\mu}(0), 2\lambda_o)} \|\mathbf{f}(n) - \mathbf{f}(n+1)\| \left\|\mathbf{f}(n) - \tilde{\mathbf{f}}\right\| \end{aligned} \tag{3.20}$$

Define $D(n) := (-G(\mathbf{f}(\infty))) - (-G(\mathbf{f}(n)))$. Thus, (3.20) implies

$$D(n) \leq K_1 \|\mathbf{f}(n) - \mathbf{f}(n+1)\|, \tag{3.21}$$

where $K_1 = \frac{K}{-\delta u(\boldsymbol{\mu}(0), 2\lambda_o)} \max_{\left\{\mathbf{f} \left| \underline{f} \leq f_\phi \leq r^* \right.\right\}} \left\| \mathbf{f} - \tilde{\mathbf{f}} \right\|$.

On the other hand, over the domain $\left\{\mathbf{f} \left| \underline{f} \leq f_\phi \leq r^* \right.\right\}$, $\sum_{\phi \in \Phi} \log(f_\phi)$ is a strongly convex function (see Proposition B.5 in [69]). Thus,

$$\begin{aligned} & D(n) - D(n+1) \\ = \; & (-G(\mathbf{f}(n+1))) - (-G(\mathbf{f}(n))) \\ \geq \; & \rho \left\| \mathbf{f}(n) - \mathbf{f}(n+1) \right\|^2 \end{aligned} \tag{3.22}$$

for some $\rho > 0$.

From (3.21) and (3.22),

$$D(n)^2 \leq K_1^2 \left\| \mathbf{f}(n) - \mathbf{f}(n+1) \right\|^2 \leq \frac{K_1^2}{\rho} \left(D(n) - D(n+1) \right),$$

or

$$D(n+1) \leq D(n) - \frac{\rho}{K_1^2} D(n)^2.$$

As shown in [117],

$$\begin{aligned} \frac{1}{D(n+1)} \; & \geq \; \frac{1}{D(n)} \frac{1}{1 - \frac{\rho}{K_1^2} D(n)} \\ & = \; \frac{1}{D(n)} \sum_{i=0}^{\infty} \left(\frac{\rho}{K_1^2} D(n) \right)^i \\ & \geq \; \frac{1}{D(n)} \left(1 + \frac{\rho}{K_1^2} D(n) \right) = \frac{1}{D(n)} + \frac{\rho}{K_1^2}. \end{aligned}$$

Using induction, we have,

$$\frac{1}{D(n)} \geq \frac{1}{D(0)} + n \frac{\rho}{K_1^2},$$

or

$$D(n) \leq \frac{1}{\frac{1}{D(0)} + n \frac{\rho}{K_1^2}} \leq \frac{1}{n \frac{\rho}{K_1^2}}.$$

Thus,

$$G(\mathbf{f}(\infty)) - G(\mathbf{f}(n)) \leq \frac{K_1^2}{\rho} \frac{1}{n}.$$

■

Chapter 4

COMMUNICATION MODELS

The throughput of a network depends on the capabilities of the MAC and physical layer. However, including a detailed model of communication can complicate the development and analysis of computation schemes, for example, the resulting scheme might be computational intractable. Hence, simplifications are often made. The communication models found in the literature can be divided into two classes, namely *protocol communication models* and *physical communication models*. Several examples of each type of model are discussed in this chapter. Specifically, precise definitions of the physical models and protocol models are provided in Section 4.1 and Section 4.2 respectively.

4.1 Physical Communication Models

4.1.1 Shannon Capacity

Shannon's Theorem specifies that the maximum possible data rate across a link is

$$BW \times \log_2 (1 + SINR)$$

where BW is the bandwidth of the channel and the SINR is the ratio of the signal power to the power of the interference and noise[1]. If links a, b, ..., and c are

[1] It is assumed that the noise *and* interference are additive white Gaussian nose (AWGN).

transmitting, then the SINR at the receiver of link x is

$$SINR^r_{a,b,\ldots,c}(x) := \frac{H^{t,r}_{x,x}}{H^{t,r}_{a,x} + H^{t,r}_{b,x} + \cdots + H^{t,r}_{c,x} + \mathcal{N}_0},$$

where $\mathcal{N}_0$ is the noise power and $H^{t,r}_{x,x}$ is the normalized channel gain across link x. That is $H^{t,r}_{x,x}$ is the signal strength at the receiver of link x due to the data transmission by the transmitter of link x. Similarly, $H^{t,r}_{a,x}$ is the signal strength at the receiver of link x due to the data transmission by the transmitter of link a. Note that the transmission power is embedded into $H^{t,r}_{a,x}$. Thus, if the channel gain from the transmitter of link a to the receiver of link x is $h_{a,x}$, and link a transmits with power p_a, then $H^{t,r}_{a,x} = h_{a,x}p_a$. If a link can transmit at different powers, we represent that link with multiple distinct links between the same transmitter and receiver. By convention, $H^{t,r}_{x,y} = \infty$ if x's transmitter is y's receiver. Hence, in this case, if data is transmitted across link x, then it is not possible to receive any data transmitted across link y.

Considering the above, the maximum data rate achievable across link x when links a,b,..., and c are transmitting is given by

$$R_{a,b,\ldots,c}(x) = BW \times \log_2(1 + SINR^r_{a,b,\ldots,c}(x))$$

Of course, here, the Shannon capacity does not account for any fixed overhead (e.g., 801.11's SIFS and preamble).

4.1.2 802.11 Style Model

The Shannon capacity cannot be achieved in practice. Today's physical layers, such as 802.11a support a set of modulation and coding schemes. Each scheme coincides with a particular relationship between SINR and bit-error probability[2]. We assume that M modulation schemes provide bit-rates $BR(1)$, $BR(2), \cdots, BR(M)$.

[2] The impact of delay spread and Doppler spread are ignored.

The probability of successful packet transmission with the mth modulation scheme is denoted by $PSP_Z(m, SINR)$, where the subscript Z denotes the packet size. Hence, if links $a, b, \cdots, c$ are transmitting, then the probability of successful transmission of a packet of length Z across link x with modulation scheme m is denoted by $PSP_Z\left(m, SINR^r_{a,b,\ldots,c}(x)\right)$.

4.1.2.1 Without ACKs

Here it is assumed that ACKs are not used. Rather, the modulation scheme is selected so that losses occur rather infrequently. The resulting data rate when links $a, b, ..,$ and c are also transmitting is denote by $R_{a,b,\ldots,c}(x)$ and is given by

$$R_{a,b,\ldots,c}(x) = \max_m \frac{Z \times 8}{\left(\frac{Z \times 8}{BR(m)} + F_{oh}\right)} \times PSP_Z\left(m, SINR^r_{a,b,\ldots,c}(x)\right)$$

where F_{oh} is the fixed overhead that represents time spacings such as SIFS and radio synchronization.

4.1.2.2 With Unsynchronized ACKs

The average number of transmissions until the data is successfully delivered across the link is $1/PSP$. Thus, the effective data rate is approximately $BR \times PSP$ where it is assumed that exponential back-off is disabled. We consider two ways to ACK packets. In the first case, it is assumed that the ACK is transmitted just after the data packet is transmitted. This scheme is nearly the same as 802.11. One important difference between this case and 802.11 is that carrier sensing, RTS and CTS are not used. In this case, ACKs can be transmitted at any time; hence interference is due to both data and ACK transmissions. Thus, the SINR at the receiver of link x is

$$SINR^{r,UA}_{a,b,\ldots,c}(x) = \frac{H^{t,r}_{x,x}}{\max\left(H^{t,r}_{a,x}, H^{r,r}_{a,x}\right) + \ldots + \max\left(H^{t,r}_{c,x}, H^{r,r}_{c,x}\right) + \mathcal{N}_0}$$

where $H_{a,x}^{r,r}$ is the signal strength at the receiver of link x due to an ACK transmission by the receiver of link a. Since a link cannot simultaneously transmit data and ACKs, $\max\left(H_{a,x}^{t,r}, H_{a,x}^{r,r}\right)$ is the worst case interference due to the data or ACK transmission across link a.

Similarly, the SINR experienced at the transmitter of link x when receiving the ACK packets is

$$SINR_{a,b,...,c}^{t,UA}(x) = \frac{H_{x,x}^{r,t}}{\max\left(H_{a,x}^{t,t}, H_{a,x}^{r,t}\right) + ... + \max\left(H_{c,x}^{t,t}, H_{c,x}^{r,t}\right) + \mathcal{N}_0}$$

where $H_{a,x}^{t,t}$ and $H_{a,x}^{r,t}$ are the signal strengths at the transmitter of link x due to the data transmission and the ACK transmission by link a respectively.

Finally, if unsynchronized ACKs are used, links a, b, ..., and c are transmitting, and the data packets are size Z, then the effective data rate across link x is

$$\begin{aligned} R_{a,b,...,c}(x) = & \max_{m,n} \frac{Z \times 8}{\left(\frac{Z \times 8}{BR(m)} + \frac{14 \times 8}{BR(n)} + 2F_{oh}\right)} \\ & \times PSP_Z\left(m, SINR_{a,b,...,c}^{r,UA}(x)\right) \\ & \times PSP_{14}\left(n, SINR_{a,b,...,c}^{t,UA}(x)\right) \end{aligned}$$

where it is assumed that ACKs have 14 B, as is the case in 802.11. Closely related schemes to select the modulation may fix the ACK modulation scheme to be the slowest bit-rate (i.e., $n = 1$) or to require the data and ACK to use the same bit-rate (i.e., $m = n$).

4.1.2.3 With Synchronized ACKs

In the previous case, ACKs may interfere with data transmissions and vice versa. This interference can be eliminated if the ACK transmissions are synchronized. Specifically, we define

$$SINR_{a,b,...,c}^{r,SA}(x) = \frac{H_{x,x}^{t,r}}{H_{a,x}^{t,r} + H_{b,x}^{t,r} + ... + H_{c,x}^{t,r} + \mathcal{N}_0}$$

and

$$SINR^{t,SA}_{a,b,...,c}(x) = \frac{H^{r,t}_{x,x}}{H^{r,t}_{a,x} + H^{r,t}_{b,x} + ... + H^{r,t}_{c,x} + \mathcal{N}_0}.$$

Then the maximum data rate across link x

$$\begin{aligned} R_{a,b,...,c}(x) = & \max_{m,n} \frac{Z \times 8}{\left(\frac{Z \times 8}{BR(m)} + \frac{14 \times 8}{BR(n)} + 2F_{oh}\right)} \\ & \times PSP_Z\left(m, SINR^{r,SA}_{a,b,...,c}(x)\right) \\ & \times PSP_{14}\left(n, SINR^{t,SA}_{a,b,...,c}(x)\right). \end{aligned}$$

Remark 1 *While 802.11 uses ACKs, it is difficult to synchronize them in practice. For example, if the packets are of different sizes and/or the transmission bit-rates are different, then the ACK packets that follow each data transmission would not be synchronized. Another approach to ACK synchronization would be to use block data transfer and block ACK transmission at the end of the time slot, as is done in 802.11e. One important drawback of this approach is that the link layer will reorder packets; TCP interprets packets that are greatly out of order as an indication of congestion. Thus, we conclude that synchronized ACKs is not particularly realistic. Nonetheless, the model is included in our study in order to understand the impact of interference induced by ACKs.*

4.2 Protocol Communication Models

In the above models, the data rate depends on the SINR, and the interference can be from multiple sources. One drawback of such models is that they do not easily lend themselves to representations as a graph. The protocol models are an alternative class of models that does allow graph-based analysis of communication and throughput algorithms. For this reason, the protocol model is widely used. The drawback of this model is that it does not accurately model interference.

Four types of protocol models are considered. In all cases, it is assumed that transmissions across a link can only occur if no transmission is occurring across

any other links in a specific set of links. These links are referred to as the set of *conflicting neighbors* of the link. It is further assumed that a link can transmit at its full data rate if none of its conflicting neighbors are transmitting, where the full data rate is the achievable data rate if no link in the entire network is transmitting. On the other hand, if any link within its set of conflicting neighbors is transmitting, no transmission is possible. The difference between the various Protocol Communication Models is the set of conflicting neighbors.

4.2.1 Node Exclusive Model

The Node Exclusive Model is the simplest communication model. In this case, a link cannot transmit only if the transmitter or receiver is also involved in a transmission. Since $H_{x,y}^{t,r} = \infty$ implies that the transmitter of link x is the receiver of link y, the set of conflicting neighbors in this case is

$$\chi(x) := \left\{ y \;\middle|\; \begin{array}{l} H_{y,x}^{t,r} = \infty,\ H_{y,x}^{r,t} = \infty, \\ H_{y,x}^{t,t} = \infty,\ H_{y,x}^{r,r} = \infty \end{array} \right\}.$$

It is assumed that if data transmissions are successfully received, then the receiver transmits an ACK, which must be correctly received in order to complete the data delivery. Thus, the (theoretical) effective data rate across link x is

$$\begin{aligned} R_{\varnothing}(x) = \max_{m} & \frac{Z \times 8}{\left(\frac{Z \times 8}{BR(m)} + \frac{14 \times 8}{BR(1)} + 2F_{oh}\right)} \\ & \times PSP_Z\left(m, SNR_{\varnothing}^{r}(x)\right) PSP_{14}\left(1, SNR_{\varnothing}^{t}(x)\right), \end{aligned} \tag{4.1}$$

where $SNR_{\varnothing}^{r}(x) := H_{x,x}^{t,r}/\mathcal{N}_0$ and $SNR_{\varnothing}^{t}(x) := H_{x,x}^{r,t}/\mathcal{N}_0$, and we assume that the ACK is 14 B and is transmitted at the slowest bit-rate. It is important to note that this data rate neglects interference and hence might not be achieved in practice.

Therefore, the data rate of link x is

$$R_{a,b,\ldots,c}(x) = \begin{cases} R_{\varnothing}(x) \ \text{ if } a, b, \ldots, c \notin \chi(x) \\ 0 \text{ otherwise} \end{cases},$$

where $R_{\varnothing}(x)$ is given in (4.1).

4.2.2 Two-Hop Node Exclusive Model

The Node Exclusive Model ignores transmissions by nearby nodes. Consequently, this model greatly overestimates the physical layer's ability to withstand interference. This model can be made less optimistic by considering interference from transmissions that are "two hops" away either the transmitter or receiver. The definition of a *hop* is problematic. Specifically, even if the channel is quite poor and hence, the probability of successful transmission is near to, but greater than zero, if ACKs are used, then eventually a packet will be delivered, establishing a communication link.

One approach is to define that nodes are one hop apart if the route forwards packet directly between the nodes. Thus, define $\mathcal{N}(\nu)$ to be the set of nodes that node v transmits packet to or receives packets from. Define $v_t(x)$ and $v_r(x)$ to be the transmitter and receiver of link x, respectively.

$$\chi(x) := \{y|\ (N(\nu_t(x)) \cup N(\nu_r(x))) \cap (N(\nu_r(y)) \cup N(\nu_r(y))) \neq \varnothing\},$$

and

$$R_{a,b,...,c}(x) = \begin{cases} R_{\varnothing}(x) \text{ if } a,b,...,c \notin \chi(x) \\ 0 \text{ otherwise} \end{cases}.$$

4.2.3 The Sensing Communication Model

The sensing protocol model is perhaps the most widely examined communication model. In this case, a transmission across a link between nodes A and B cannot occur if there is some node C that is also transmitting where the received signal strength of C's transmission at either node A or B is above the *Channel Sensing Threshold*, denoted by γ. This model is motivated by 802.11. In 802.11 a node will only transmit if the channel is idle just before transmission. More specifically, the node will only transmit if the received signal strength of the aggregate of all other node's transmissions is below the *Channel Sensing Threshold*. Furthermore, since

802.11 uses either RTS-CTS-Data-ACK or Data-ACK, transmissions are two-way. Thus, in order for a transmission to occur, both the receiver and the transmitter must find the channel to be idle before transmissions.

The set of conflicting neighbors of link x is denoted $\chi(x)$ and is given by

$$\chi(x) := \left\{ y \left| \begin{array}{c} H_{y,x}^{t,r} > \gamma,\ H_{y,x}^{r,t} > \gamma, \\ H_{y,x}^{t,t} > \gamma,\ H_{y,x}^{r,r} > \gamma \end{array} \right. \right\}.$$

This model can be interpreted in a slightly different way based on the *Interference Range*. In this model, a transmission across a link x will fail if link y is transmitting where $H_{y,x}^{t,r}$ >*Interference Range*. Since transmissions are bidirectional, this *Interference Range* model is the same as the Sensing Communication Model, but the *Channel Sensing Threshold* is replaced with the *Interference Range*.

The data rate of link x is

$$R_{a,b,...,c}(x) = \begin{cases} R_{\varnothing}(x) \text{ if } a, b, ..., c \notin \chi(x) \\ 0 \text{ otherwise} \end{cases}.$$

4.2.4 SINR Protocol Model

The Sensing Protocol Model simplifies 802.11 by assuming that a transmission can not occur from A to B if the received signal strength from any single node's transmission exceeds the *Channel Sensing Threshold* at either node A or B, and transmission will successfully occur at full rate otherwise. Thus, two simplifications are made.

1. The Sensing Protocol Model neglects the aggregate of the interference from multiple nodes transmitting.

2. The Sensing Protocol Model assumes that if the interfering signal strength is below particular *Channel Sensing Threshold*, then transmission at the full rate is possible.

This second simplification results from defining a single *Channel Sensing Threshold* for all links, where better performance would likely be achieved when such a parameter is determined on a per link basis. One alternative is to define that links are not in conflict if $SINR$ at each receiver is above some threshold. SINR protocol model is the protocol version of the physical model. Here, we provide the SINR protocol model of the *Without ACKs*, the *With Unsynchronized ACKs* and the *With Synchronized ACKs*, respectively.

4.2.4.1 Without ACKs

Define the modulation scheme used by link x via

$$\mathcal{M}(x) := \arg\max_{m} \frac{Z \times 8}{\frac{Z \times 8}{BR(m)} + F_{oh}} \times PSP_Z\left(m, SNR^r_{\varnothing}(x)\right)$$
$$\text{s.t.: } \frac{PSP_Z\left(m, SNR^r_{\varnothing}(x)\right) - PSP_Z\left(m, SNR^r_{\varnothing}(x) - G2 - G3\right)}{PSP_Z\left(m, SNR^r_{\varnothing}(x)\right)} \leq G1$$

where $G2$ and $G3$ are used to reduces sensitivity to interference. Specifically, $G2$ is the buffer for binary conflicts and $G3$ is the buffer for multi-conflicts which is presented in Section 5.3. In this study, we used $G1 = 0.01$, $G2 = 2$ dB, and $G3 = 1$ dB.

Two links are in conflict if they cannot both transmit simultaneously and achieve the target transmission probability. Let $\mathcal{T}(x, m)$ be the minimum required $SINR$ to decode the data and achieve the target success probability at the receiver of link x when the modulation scheme m is used. A set of $\mathcal{T}(x, m), m = 1, ..., \mathcal{M}(x)$ can be obtained for modulation schemes from

$$\mathcal{T}(x, m) = \min\left\{\mathcal{T} \left| \frac{PSP_Z\left(m, SNR^r_{\varnothing}(x)\right) - PSP_Z\left(m, \mathcal{T} - G2\right)}{PSP_Z\left(m, SNR^r_{\varnothing}(x)\right)} \leq G1 \right.\right\}.$$

The set of conflicting neighbors is

$$\chi(x) := \left\{ y \left| \begin{array}{c} SINR^r_y(x) < \mathcal{T}(x, \mathcal{M}(x)) \\ \text{or } SINR^r_x(y) < \mathcal{T}(y, \mathcal{M}(y)) \end{array} \right. \right\}.$$

and the (theoretical) effective data rate across link x is

$$R_{\varnothing}(x) = \frac{Z \times 8}{\frac{Z \times 8}{BR(\mathcal{M}(x))} + F_{oh}} \times PSP_Z\left(\mathcal{M}(x), SNR_{\varnothing}^{r}(x)\right).$$

Therefore, the data rate of link x is

$$R_{a,b,...,c}(x) := \begin{cases} R_{\varnothing}(x) \text{ if } a, b, ..., c \notin \chi(x) \\ 0 \text{ otherwise} \end{cases}.$$

4.2.4.2 With Unsynchronized ACKs

Define the modulation schemes for data and ACK used by x via

$$\begin{aligned}(\mathcal{M}(x), \mathcal{N}(x)) := \arg\max_{m,n} & \frac{Z \times 8}{\left(\frac{Z \times 8}{BR(m)} + \frac{14 \times 8}{BR(n)} + 2F_{oh}\right)} \times PSP_Z\left(m, SNR_{\varnothing}^{r,UA}(x)\right) \\ & \times PSP_{14}\left(n, SNR_{\varnothing}^{t,UA}(x)\right)\end{aligned} \tag{4.2}$$

such that:

$$\frac{PSP_Z\left(m, SNR_{\varnothing}^{r,UA}(x)\right) - PSP_Z\left(m, SNR_{\varnothing}^{r,UA}(x) - G2 - G3\right)}{PSP_Z\left(m, SNR_{\varnothing}^{r,UA}(x)\right)} \leq G1$$

$$\frac{PSP_{14}\left(n, SNR_{\varnothing}^{t,UA}(x)\right) - PSP_{14}\left(n, SNR_{\varnothing}^{t,UA}(x) - G2 - G3\right)}{PSP_{14}\left(n, SNR_{\varnothing}^{t,UA}(x)\right)} \leq G1$$

where we assume that data and ACK packets use the same set of $G1, G2, G3$, and it is easy to extend to use different set of Guards. m and n are the modulation schemes that solve (4.2), or the other schemes described just after (4.2) that restrict $m = n$ or $n = 1$. These modulation selection schemes are named as *OptDataAck*, *SameAck* and *MinAck*, respectively.

Two links are in conflict if they cannot both transmit simultaneously and achieve the target data and ACK transmission probability. Let $\mathcal{T}_{data}(x, m)$ and

$\mathcal{T}_{ack}(x,m)$ be the minimum required $SINR$ to decode the data or ACK packet and achieve the target success probability at link x when the modulation scheme m is used. A set of $\mathcal{T}_{data}(x,m)$ and $\mathcal{T}_{ack}(x,n)$, $m=1,...,\mathcal{M}(x)$, $n=1,...,\mathcal{N}(x)$ can be obtained for modulation schemes from

$$\mathcal{T}_{data}(x,m)=\min\left\{\mathcal{T}\left|\frac{PSP_Z\left(m,SNR_{\varnothing}^{r,UA}(x)\right)-PSP_Z\left(m,\mathcal{T}-G2\right)}{PSP_Z\left(m,SNR_{\varnothing}^{r,UA}(x)\right)}\leq G1\right.\right\}$$

and

$$\mathcal{T}_{ack}(x,n)=\min\left\{\mathcal{T}\left|\frac{PSP_{14}\left(n,SNR_{\varnothing}^{t,UA}(x)\right)-PSP_{14}\left(n,\mathcal{T}-G2\right)}{PSP_{14}\left(n,SNR_{\varnothing}^{t,UA}(x)\right)}\leq G1\right.\right\}.$$

The set of conflicting neighbors is

$$\chi(x):=\left\{y\left|\begin{array}{c}SINR_y^{r,UA}(x)<\mathcal{T}_{data}(x,\mathcal{M}(x))\\ SINR_y^{t,UA}(x)<\mathcal{T}_{ack}(x,\mathcal{N}(x))\\ \text{or } SINR_x^{r,UA}(y)<\mathcal{T}_{data}(y,\mathcal{M}(y))\\ SINR_x^{t,UA}(y)<\mathcal{T}_{ack}(y,\mathcal{N}(y))\end{array}\right.\right\}.$$

and the (theoretical) effective data rate across link x is

$$\begin{aligned}R_{\varnothing}^{SINR}(x) &= \frac{Z\times 8}{\left(\frac{Z\times 8}{BR(\mathcal{M}(x))}+\frac{14\times 8}{BR(\mathcal{N}(x))}+2F_{oh}\right)}\\ &\quad\times PSP_Z\left(\mathcal{M}(x),SNR_{\varnothing}^{r,UA}(x)\right)\\ &\quad\times PSP_{14}\left(\mathcal{N}(x),SNR_{\varnothing}^{t,UA}(x)\right).\end{aligned}$$

Therefore, the data rate of link x is

$$R_{a,b,...,c}(x):=\begin{cases}R_{\varnothing}^{SINR}(x) \text{ if } a,b,...,c\notin\chi(x)\\ 0 \text{ otherwise}\end{cases}.$$

Note that there are three ways to select the ACK rate (i.e., setting $n=1$, $n=m$ or optimizing over both m and n). While selecting different ACK rates will not result in a significant change in the effective data rate, it will change the value of $\mathcal{T}_{ack}(x)$, which may greatly impact $\chi(x)$ and hence have a significant impact on spatial multiplexing.

4.2.4.3 With Synchronized ACKs

When the synchronized ACKs are considered, the set of conflicting neighbors are defined as

$$\chi(x) := \left\{ y \left| \begin{array}{c} SINR_y^{r,SA}(x) < \mathcal{T}_{data}(x, \mathcal{M}(x)) \\ SINR_y^{t,SA}(x) < \mathcal{T}_{ack}(x, \mathcal{N}(x)) \\ \text{or } SINR_x^{r,SA}(y) < \mathcal{T}_{data}(y, \mathcal{M}(y)) \\ SINR_x^{t,SA}(y) < \mathcal{T}_{ack}(y, \mathcal{N}(y)) \end{array} \right. \right\}.$$

and the data rate of link x is

$$R_{a,b,...,c}(x) := \begin{cases} R_{\varnothing}^{SINR}(x) \text{ if } a, b, ..., c \notin \chi(x) \\ 0 \text{ otherwise} \end{cases}.$$

4.2.5 Multiple Modulation Schemes and Transmit Powers SINR Protocol Model

Today's transceivers support a number of bit-rates and transmit powers. Here we develop an extension of the SINR Protocol Model that supports multiple bit-rates and transmit powers. We focus on the SINR protocol model and just consider the case *without ACKs*, because it is simple and the other two ACK models can be easily extended.

Suppose that there are M modulation/coding schemes and S transmission powers available for each link, then associated with each physical link x is the set of logical links $x_{m,s}$ with $1 \leq m \leq M$ and $1 \leq s \leq S$. In this case, the assignment specifies which links are transmitting, their bit-rate, and their transmission power. Specifically, an assignment $\mathbf{v} \in \{0,1\}^{L \times M \times S}$, where $v_{x_{m,s}} = 1$ implies that the physical link x is transmitting at bit-rate m and with power $p_{x_{m,s}}$. Note that, $p_{x_{m,s}}$ need not be the same as $p_{y_{m,s}}$ or $p_{x_{n,s}}$, that is, the set of transmission powers depend on the link and the modulation.

Let us define SNR and SINR at the receiver of logical link $x_{m,s}$ as

$$\begin{aligned} SNR^r_{\varnothing}\left(x_{m,s}\right) &= \frac{H_{x,x}p_{x_{m,s}}}{\mathcal{N}_0}, \\ SINR^r_{y_{n,t}}\left(x_{m,s}\right) &= \frac{H_{x,x}p_{x_{m,s}}}{H_{y,x}p_{y_{n,t}}+\mathcal{N}_0}. \end{aligned}$$

Let $\mathcal{T}(x,m,s)$ be the minimum required $SINR$ to decode the data and achieve the target success probability at the receiver of link x when the modulation scheme m and transmission power s are used. Then, $\mathcal{T}(x,m,s)$ can be obtained from

$$\mathcal{T}(x,m,s) = \min\left\{\mathcal{T}\left|\frac{PSP_Z\left(m,SNR^r_{\varnothing}\left(x_{m,s}\right)\right)-PSP_Z\left(m,\mathcal{T}-G2\right)}{PSP_Z\left(m,SNR^r_{\varnothing}\left(x_{m,s}\right)\right)}\leq G1\right.\right\}.$$

The set of conflicting neighbors is

$$\chi\left(x_{m,s}\right) := \left\{ y \left| \begin{array}{l} SINR^r_{y_{n,t}}\left(x_{m,s}\right) < \mathcal{T}\left(x,m,s\right) \\ \text{or } SINR^r_{x_{m,s}}\left(y_{n,t}\right) < \mathcal{T}\left(y,n,t\right) \\ \text{or } x = y \end{array} \right. \right\}.$$

and the (theoretical) effective data rate across link $x_{m,s}$ is

$$R_{\varnothing}\left(x_{m,s}\right) = \frac{Z\times 8}{\frac{Z\times 8}{BR(m)}+F_{oh}} \times PSP_Z\left(m,SNR^r_{\varnothing}\left(x_{m,s}\right)\right).$$

Therefore, the data rate of link $x_{m,s}$ is

$$R_{a,b,...,c}\left(x_{m,s}\right) := \left\{ \begin{array}{l} R_{\varnothing}\left(x_{m,s}\right) \text{ if } a,b,...,c \notin \chi\left(x_{m,s}\right) \\ 0 \text{ otherwise} \end{array} \right. .$$

Note that the size of the conflict graph grows quickly as more modulation schemes and transmit powers are considered.

4.3 Summary

The communication models found in the literature can be divided into two classes, named as physical communication models and protocol communication models. In the case of physical communication models, this chapter presents two different

models. One is Shannon capacity, and the other is 802.11 style model with three different options, which are *Without ACKs*, the *With Unsynchronized ACKs* and the *With Synchronized ACKs*. In these models, the data rate depends on the SINR, and the interference may be from multiple sources. The graph theory can not be applied in these models.

In the case of protocol communication models, this chapter presents node exclusive model, 2-hop node exclusive model, sensing model, and SINR protocol model with three options, which are *Without ACKs*, the *With Unsynchronized ACKs* and the *With Synchronized ACKs*. The protocol model defines the normial data rate and the set of conflict links for each link, and difference between the various protocol models is the set of conflicting links. Here, the conflict links is based on pairwise. Therefore, the graph-based algorithms can be easily applied to the protocol models.

Chapter 5

CONSTRUCTING THE SET OF CONSIDERED ASSIGNMENTS

The algorithms described in Chapter 3 iteratively add assignments to the set of considered assignments. An approach to constructing a good set of assignments is to start with a particular set of assignments, $\mathcal{V}$, select an assignment $\mathbf{v}^+ \notin \mathcal{V}$, and evaluate the resulting utility with the set of assignments $\mathbf{v}^+ \cup \mathcal{V}$. Specifically, at each iteration, a linear test (3.9) is provided to efficiently determine whether an assignment should be added to the set of considered assignments.

In this chapter it will be shown that, in the case of Protocol models, finding an assignment is equivalent to finding the maximum weighted independent set (MWIS). Solving the MWIS problem is NP-hard [13]. Thus, along with two exact methods, two approximation methods to find a new assignment are analyzed.

The model of the problem of finding new assignments as a MWIS problem suffers from some drawbacks in that the graph theoretic model of interference neglects the aggregate impact of interference from multiple transmissions. The new assignment obtained from Protocol models may not transmit simultaneously. Thus, this chapter presents techniques for correcting assignments.

Therefore, selecting the initial set $\mathcal{V}^0$, selecting which $\mathbf{v}^+ \notin \mathcal{V}$ to consider, and removing assignments from $\mathcal{V}$ are addressed in sections 5.1, 5.2, and 5.5, respectively. Two methods of correcting the multi-interference for the Protocol model are presented in section 5.3 and section 5.4.

Algorithm 3 Selecting an Initial Set of Assignments

1: Set $\mathcal{V}$ empty, and set $w_x = 0$ for all links x.
2: **repeat**
3: Start an assignment $\mathbf{v}$ with $v_x = 0$ for all x.
4: Randomly select a link x such that $w_x = 0$. Set $w_x = 1$ and $v_x = 1$.
5: **repeat**
6: Randomly select a link y such that $w_y = 0$ and $y \notin \bigcup_{\{x|v_x=1\}} \chi(x)$. Furthermore, check whether each active link in assignment $\mathbf{v}$ satisfies the desired SINR requirement (see Section 5.3).
7: **if** such a y exists **then**
set $w_y = 1$, $v_y = 1$.
8: **end if**
9: **until** No such a y exists
10: Set $\mathcal{V} = \mathcal{V} \cup \mathbf{v}$.
11: **until**
For all x there exists a $\mathbf{v} \in \mathcal{V}$ such that $v_x = 1$.
12: $\mathcal{V}$ is the set of initial assignments.

5.1 Initial Assignments

This initial set of assignments must result in a solution to (3.2) where the flow rates are non-zero. Two types of initial assignments are used. First, assignments that have a single link transmitting are included. Second, assignments composed of links such that any pair of links that are turned on in the assignment are not in conflict. Specifically, as for the $SINR$ protocol model, all active links in the assignment must satisfy the corresponding $SINR$ threshold requirement, and the assignments must be feasible.

The second set of initial assignments is constructed in a greedy way. Each link is active in one assignment, and each assignment contains as many active links as possible. The idea is to randomly select a link and add that link to the assignment $\mathbf{v}$. Then, randomly select next link from the remaining links that are not in conflict with any links in the assignment $\mathbf{v}$, and add it to $\mathbf{v}$. This is repeated until all remaining links not in the assignment are in conflict with at least one link in the

assignment $\mathbf{v}$. This process is repeated until all links are in one assignment. The Details of the greedy selection is given in Algorithm 3. We note that a wide range of techniques could be employed to select $\mathcal{V}(0)$. An examination of the performance of these various techniques is left for future work.

5.2 Searching for New Assignments

In light of the above, the main challenge is finding an assignment $\mathbf{v}$ so that $\sum_x R(\mathbf{v}, x)\,\mu_x > \lambda$ where $\boldsymbol{\mu}$ and λ are the Lagrange multipliers associated with constraints (3.2b) and (3.2c) of Problem (3.2).

One approach to finding such a $\mathbf{v}$ is to solve

$$\max_{\mathbf{v}} \sum_x R(\mathbf{v}, x)\,\mu_x. \tag{5.1}$$

In most cases, it is not necessary to find the assignment that maximizes $\sum_x R(\mathbf{v}, x)\,\mu_x$, but just one that satisfies $\sum_x R(v, x)\,\mu_x > \lambda$. On the other hand, if no assignment exists, then the schedule found from the currently considered assignments yields the optimal throughput. Thus, in order to ensure that the currently considered assignments result in the optimal throughput, (5.1) must be solved.

Unfortunately, in general, solving (5.1) is NP-hard. However, as shown shortly, in the case of the protocol communication model, solving (5.1) is equivalent to a graph theoretic problem known as the maximum weighted independent set (MWIS) problem, which has been extensively studied. Thus, for the cases where there are a moderate number of links, there are a large number of algorithms that efficiently solve (5.1). For networks with more links, there are a large set of algorithms that approximately solve (5.1). And finally, there exists extensive theory regarding the solutions and solvability of (5.1). The richness of the theory of solving (5.1) when the protocol communication model is used is the most significant advantage of the protocol model over the physical model.

5.2.1 Weighted Conflict Graph

The connection between solving (5.1) and the maximum weighted independent set problem is made by the *conflict graph*. As discussed in Section 4.2, for each link x, the protocol communication model defines a set of conflicting links $\chi(x)$. The sets of conflicting links induce the conflict graph as follows. Each link in the network induces a vertex in the conflict graph. Thus, a link x in the network is associated with a vertex in the conflict graph; this vertex is denoted with x, where whether x refers to a link in the network or a vertex in the conflict graph is clear from the context. There is an edge between vertices x and y if $y \in \chi(x)$ (note, the we require that if $y \in \chi(x)$, then $x \in \chi(y)$). In the weighted conflict graph, the vertex associated with link x is assigned weight $R_{\varnothing}(x)\,\mu_x$, where $R_{\varnothing}(x)$ is the data rate across link x for the protocol communication model.

5.2.2 The Maximum Weighted Independent Set

5.2.2.1 MWIS and New Assignment

Let us show the equivalence of MWIS and solving (5.1) in this section. An independent set on the weighted conflict graph is a set of vertices such that no two vertices in the set are neighbors, that is, if x and y are in an independent set, then $x \notin \chi(y)$. To put it another way, if $I = \{x_i : i = 1, 2, ..\}$ is an independent set, then the set of links $\{x_i : i = 1, 2, ...\}$ can simultaneously transmit, and each link can transmit at data rate $R_I(x_i)$. Furthermore, under the Protocol Communication Model, since I is an independent set, we have $R_I(x_i) = R_{\varnothing}(x_i)$, that is, the interference from the other links in the independent set does not impact the sending rate of link x_i. The weight of an independent set is the sum of the vertex weights. Thus, the weight of I is $\sum_{x \in I} R_{\varnothing}(x)\,\mu_x$.

Let I be an independent set and let $\mathbf{v}_I$ be the assignment corresponding to I, then link x is transmitting in $\mathbf{v}_I$ if and only if $x \in I$. Hence, $R(\mathbf{v}_I, x) = R_{\varnothing}(x)$ if and only if $x \in I$, and thus, $\sum_{x \in I} R_{\varnothing}(x)\,\mu_x = \sum_x R(\mathbf{v}_I, x)\,\mu_x$, which is the quantity

we seek to maximize. Therefore, finding a maximum weighted independent set of the weighted conflict graph is equivalent to solving (5.1).

5.2.2.2 Related Work of MWIS

The maximum weighted independent set (MWIS) problem is a fundamental combinatorial problem that has been the focus of a vast amount of research. In general, MWIS problem is NP-hard. Indeed, it is one of the first problems to be shown to be NP-hard [13]. However, it is not always NP-hard. If the conflict graph is a perfect graph, then the MWIS can be found in polynomial time [71]. The class of perfect graphs includes a wide variety of graphs (See pages 279-283 in [71]). For example, the interval graph is a perfect graph. In an interval graph, each vertex is associated with an interval of the real number line. Two vertices are neighbors if their corresponding intervals overlap. It is not hard to show that if the network is restricted to one dimension (e.g., a network along a straight road), then the conflict graph that results from protocol communication models is an interval graph. A 2-D version of the interval graph is the disc graph, which, while not a perfect graph, allows the maximum weighted independent set to be computed in polynomial time [72]. Besides perfect graphs, maximum independent sets can be computed in polynomial time for several other classes of graphs., for example, for "claw-free" graphs [73], for fork-free graphs [74], for trees (see [75] for a linear time algorithm), for sparse random graphs (see [76] for a linear time algorithm), and for circle graphs (see [77] for a linear time/space algorithm).

There has been considerable effort focused on computing the MWIS in the general case. These research efforts tend to take one of two approaches. In one approach, the goal is to develop algorithms that have good worst-case performance. For example, [78] reports an algorithm with worst-case time complexity of $O\left(1.2461^{L}\right)$, while [79] achieves $O(1.2431^{L})$. One drawback of the worst-case analysis is that most graphs can be solved much faster than predicted by the worst-case analysis. And

hence, another approach is to not optimize the worst-case performance, but focus on the time to find the MWIS in graphs that have been deemed interesting. Computational methods that work well in this respect can be found in [80, 81, 82, 83], and [84], where [84] is sometimes cited as the current state-of-the-art.

While there are many cases where the MWIS can be determined in polynomial time, computing the MWIS can be computationally complex. However, criteria (3.9) does not require that the maximum independent set be found, it only requires that an independent set be found with weight that exceeds λ. Thus, approximation schemes can be used to efficiently search for good independent sets. The performance of these algorithms is compared in Section 6.5. On the other hand, more rapid convergence can be expected if the weight of the found independent sets are large. Unfortunately, it is known that there exists $\varepsilon > 0$ such that there is no polynomial approximation that has approximation ratio better than $O\left(n^{\varepsilon}\right)$ [85], where n is the number of vertices. Thus, in general, in the worst-case, the total weight of a polynomial time approximation is a factor of n less than the total weight of the MWIS. However, as discussed above, the worst-case performance may be considerably worse than what typically occurs in conflict graphs that arise from mesh networks.

Finding approximate solutions to the MWIS problems is also an active area of research (e.g., see [86] for a review of some methods). An algorithm suggested by Kako [56] is simple and often provides good results; it achieves a worst-case approximation ratio of $\bar{d}$, where $\bar{d}$ is the average weighted degree, i.e., $\bar{d} = \frac{\sum_x W(x)}{\sum_x R_x \mu_x}$, where $W\left(x\right)$ is the weighted degree of vertex x and is given by $W\left(x\right) := \frac{\sum_{y \in \chi(x)} R_y u_y}{R_x \mu_x}$. Another algorithm that has been found to perform well in realistic mesh networks is the WMIN algorithm developed in [87]. WMIN has approximation ratio Δ, the maximum degree. Other algorithms have approximation ratios $\left(\bar{d}_w + 1\right)/2$, $O\left(\bar{d}_w \log\log \bar{d}_w / \log \bar{d}_w\right)$, and $O\left(\delta_w \log\log \delta_w / \log \delta_w\right)$, where δ_w is the weighted intuitiveness of the graph [56]. See [86] for further discussion of some approximation algorithms for the MWIS problem.

One drawback of using approximate solutions to the MWIS problem is that if the approximation scheme does not result in a better assignment, then, in general, it is not possible to determine whether there does not exist a better assignment (and hence the current schedule is optimal) or there does exist a better assignment, but the approximation algorithm is unable to find it. While in general, such a determination is not possible in polynomial time, as discussed next, in some cases it is possible, and, indeed, in all the realistic mesh networks we have examined, such a determination is possible in polynomial time.

Two important metrics of weighted graphs are ω, the total weight of the MWIS and $\mathcal{K}$, the weighted chromatic number. In general, both numbers are NP-hard to compute. However, the Lovász number, $\mathcal{V}$ can be computed in polynomial time via semi-definite programming [71] and $\omega \leq \mathcal{V} \leq \mathcal{K}$. In some cases (e.g., in the case of a perfect graph), $\omega = \mathcal{V}$. Thus, in such cases it is possible to compute ω in polynomial time. Therefore, in general, if an approximation method fails to find a new assignment with revenue greater than λ, then one can compute the $\mathcal{V}$ in polynomial time and check whether $\lambda = \mathcal{V}$. If $\lambda = \mathcal{V}$, then the optimal schedule has been found. On the other hand, since $\mathcal{V}$ is only an upper bound on ω, $\lambda < \mathcal{V}$ does not imply that the assignment found by the approximation is not optimal. However, we have found that for the realistic mesh networks examined, at convergence (i.e., after Algorithm 1 has converged), we have $\lambda = \mathcal{V}$. Unfortunately, while it is possible to compute $\mathcal{V}$ in polynomial time, it does take a considerable amount of time and we have been unable to confirm if $\omega = \mathcal{V}$ for networks with more than 100 links.

5.2.3 Approximation Algorithms for MWIS

Criteria (3.9) does not require that the maximum independent set be found, it only requires that an independent set be found with weight that exceeds λ. Thus, approximation schemes can be used to efficiently search for good independent sets.

5.2.3.1 Kako's Algorithm

Here we apply a technique to find a good weighted independent set developed by Kako et al [56]. Kako's algorithm is a greedy algorithm, and we have found that Kako's algorithm provides good results based on a large number of computational experiments. Let $\mathcal{L}$ be a subset of the vertices in conflict graph. Hence, the subgraph induced by $\mathcal{L}$ is the set of vertices $\mathcal{L}$ and if $x \in \mathcal{L}$, then the neighbors of x are the set of vertices $\chi(x) \cap \mathcal{L}$.

Kako's algorithm is a greedy algorithm based on the weighted degree. For each non-selected vertex, compute the weighted degree, which is defined as the sum of a vertex's neighbors' weights divided by the vertex's weight. Hence, a vertex has a small weighted degree if its weight is large and it has only a few neighbors and/or its neighbors' weights are small. Once the weighted degrees are computed, the vertex with the smallest weighted degree is selected, and all of the selected vertex's neighbors are removed from the graph. The process is repeated until there are no vertices left in the graph. The performance of Kako's method depends on the inductiveness of the graph, see [56] for details.

Definition 1 *The weighted degree of a vertex in the subgraph induced by* $\mathcal{L}$ *is*

$$W(x, \mathcal{L}) := \frac{\sum_{y \in \chi(x) \cap \mathcal{L}} R_{\varnothing}(y) u_y}{R_{\varnothing}(x) \mu_x}, \tag{5.2}$$

where, $R_{\varnothing}(x) \mu_x$ *is the weight associated with vertex* x*.*

Kako's algorithm is as follows.

1. Set $\mathcal{L}(0)$ to be all the vertices in the conflict graph and set $k = 0$.
2. For each vertex in $\mathcal{L}(k)$, compute the weighted degree.
3. Select the vertex, $x^*(k)$, with the smallest weighted degree and include this vertex into the estimate of the maximum weighted independent set

4. Set $\mathcal{L}(k+1) = \{y \in \mathcal{L}(k) : y \notin \chi(x^*(k))\}$.

5. If $\mathcal{L}(k+1)$ is empty, stop.

6. Otherwise, set $k = k+1$ and go to 2.

This algorithm has approximation ratio $\bar{d}_w$ where $\bar{d}_w$ is the weighted degree of the conflict graph [56]. However, this algorithm has much better performance in practice.

5.2.3.2 WMIN Algorithm

Let us define

$$S(x, \mathcal{L}) = \frac{R_\varnothing(x)\, u_x}{d_{\mathcal{L}}(x) + 1}$$

where $R_\varnothing(x)\, u_x$ is the weight associated with vertex x, and $d_{\mathcal{L}}(x)$ is the degree of vertex x in the subgraph induced by $\mathcal{L}$.

WMIN [87] is a greedy algorithm in which a vertex x maximizing $S(x, \mathcal{L})$ over all $x \in \mathcal{L}$ is selected in each iteration. Therefore, once the $S(x, \mathcal{L})$ for all vertices is computed, the vertex with the largest $S(x, \mathcal{L})$ is selected, and all of the selected vertex's neighbors are removed from the graph. The process is repeated until there are no vertices left in the graph.

WMIN's algorithm is as follows.

1. Set $\mathcal{L}(0)$ to be all the vertices in the conflict graph and set $k = 0$.

2. For each vertex in $\mathcal{L}(k)$, compute $S(x, \mathcal{L}(k))$.

3. Select the vertex, $x^*(k)$, with the largest $S(x, \mathcal{L}(k))$ and include this vertex into the estimate of the maximum weighted independent set

4. Set $\mathcal{L}(k+1) = \{y \in \mathcal{L}(k) : y \notin \chi(x^*(k))\}$.

5. If $\mathcal{L}(k+1)$ is empty, stop.

6. Otherwise, set $k = k + 1$ and go to 2.

This algorithm has approximation ratio Δ where Δ is the the maximum degree of the conflict graph.

5.2.4 Exact Algorithms for MWIS

5.2.4.1 Integer Programming

Another way to compute the MWIS is to use Integer Linear Programming(ILP). Specifically, the MWIS problem can be written as

$$\max_{\mathbf{v}} \sum_{x=1}^{L} R_x \mu_x v_x \tag{5.3}$$

$$\text{such that: } v_x + v_y \leq 1 \text{ if } y \in \mathcal{X}(x) \tag{5.4}$$

$$v_x \in \{0, 1\}.$$

Note that since $v_x \in \{0, 1\}$, this problem can also be solved with binary programming.

However, in large networks, there are many constraints (5.4). The computation time can be dramatically improved if a clique decomposition is used. Specifically, a set of cliques $\{Q_i,\ i = 1, 2, ...M\}$ are found such that if $y \in \chi(x)$, then there is a clique Q_i such that $x \in Q_i$ and $y \in Q_i$. Then, Problem (5.3) becomes

$$\max_{\mathbf{v}} \sum_{x=1}^{L} R_x \mu_x v_x \tag{5.5}$$

$$\text{subject to: } \sum_{x \in Q_i} v_x \leq 1 \text{ for } i = 1, 2, ..., M$$

$$v_x \in \{0, 1\}.$$

There are many commercially available mixed integer and binary programming tools (e.g., CPLEX [88]). Our work has found that this method works well in practice, and is used mostly in our research.

While an optimal clique decomposition might further improve the computation time, a simple greedy clique decomposition results in a factor of ten improvement over (5.3). Algorithm 4 shows the greedy algorithm to find such a set of cliques $\{Q_i\ , i = 1, 2, ...M\}$.

Algorithm 4 Greedy Algorithm of Clique Decomposition

Notations: $\#\left(\chi\left(l\right)\right)$ is the number of elements in the conflict set of link l. $I^l\left(j\right)$ is the index of link j in $\chi\left(l\right)$. $W^l_{I^l(j)} = 1$ means that the conflict between link l and link j has been included in some clique.
Set $i = 1$. Set $\mathbf{W}^l = \vec{0}_{1\times\#(\chi(l))}$ for each link l.
for $l = 1$ to L **do**
  while $(\min_{j\in\chi(l)} W^l_{I^l(j)} = 0)$ **do**
    Randomly select a link j from the links with $W^l_{I^l(j)} = 0$, set $Q_i = [l\ j]$.
    Set $W^l_{I^l(j)} = 1$.
    Set $W^j_{I^j(l)} = 1$.
    while $(\cap_{k\in Q_i}\left\{j|j \in \chi\left(k\right) \text{ and } W^k_{I^k(j)} = 0\right\} \neq \varnothing)$ **do**
      Randomly select a link k from $\cap_{k\in Q_i}\left\{j|j \in \chi\left(k\right) \text{ and } W^k_{I^k(j)} = 0\right\}$.
      Set $Q_i = [Q_i\ k]$
      Set $W^j_{I^j(k)} = 1$ for $j \in Q_i$
      Set $W^k_{I^k(j)} = 1$ for $j \in Q_i$
    end while
    $i++$
  end while
end for
Set $M = i - 1$; The set of cliques is $\{Q_i : i = 1, ..., M\}$.

Let us define some notations first. $\mathbf{W}^l$ is a $1\times\#\left(\chi\left(l\right)\right)$ vector, where $\#\left(\chi\left(l\right)\right)$ is the number of elements in the conflict set of link l. $I^l\left(j\right)$ is the index of link j in $\chi\left(l\right)$, so that $\chi\left(l\right)\left(I^l\left(j\right)\right) = j$ and if $j \notin \chi\left(l\right)$, then $I^l\left(j\right)$ is not defined. If $W^l_{I^l(j)} = 1$, it means that the conflict between link l and link j has been included in some clique.

The idea of Algorithm 4 is as follows. First, initialize $\mathbf{W}^l = \vec{0}_{1\times\#(\chi(l))}$ for each link l and set $l = 1$, since we will start by processing link 1. The first clique,

Q_1, is initialized, $Q_1 = \{l\}$. Then we search for a new link j that is in conflict with all links in Q_1 and that the conflict between l and j has not already been included into some clique, i.e., $W^l_{I^l(j)} = 0$ and $W^j_{I^j(l)} = 0$. We add this j to the clique , i.e., $Q_1 = [Q_1\ j]$, and mark that the conflict between l and j is in some clique, i.e., set $W^l_{I^l(j)} = 1$ and $W^j_{I^j(l)} = 1$. If there does not exist such a link j, Q_1 is a clique. Then, the above procedures are repeated to search for the next clique Q_2. We keep on searching the cliques until $\mathbf{W}^l = \vec{1}$ for each link l. Then, we get the set of cliques $\{Q_i\ , i = 1, 2, ...M\}$.

5.2.4.2 Maximum Weighted Clique

WClique [84] is an exact method to find the maximum independent set by finding the maximum weighted clique in the complement graph. Let G^c be the complement graph of the conflict graph G. In graph theory, the complement or inverse of a graph G is a graph G^c on the same vertices such that two vertices of G^c are adjacent if and only if they are not adjacent in G. That is, to find the complement of a graph, you fill in all the missing edges, and remove all the edges that were already there. It is not the set complement of the graph; only the edges are complemented.

The vertices of the maximum weighted clique of G^c are the vertices of G without edges, which construct an independent set with maximum weight in the conflict graph G. It is easy to see that finding the maximum weighted clique of G^c is equivalent to finding the maximum weighted independent set of G.

To find the maximum weighted clique, we impose an order on the vertices: $V = \{v_1, v_2, ..., v_n\}$ where the total number of vertices is n. The performance of the algorithm depends on the ordering of the vertices. The following ordering was chosen because it outperformed the other orderings that were tried. The vertices are labeled $v_n, v_{n-1}, ...$ in the order they are chosen. For each choice of next vertex to add to this list, we consider the graph induced by the vertices that have not yet

been added. Among the vertices with smallest weight in this graph, we pick the one with the largest sum of weights of adjacent vertices.

In the algorithm, we calculate the values of the function $C(i)$, which denotes the largest weight of a clique in the subgraph induced by the vertices $S_i = \{v_i, v_{i+1}, ..., v_n\}$. Then, $C(n) = w(n)$, where $w(i)$ is the weight of vertex v_i. $C(n-1), C(n-2), ...$ are determined in a backtrack search. To determine $C(i)$, we search for cliques that contain v_i and that have weight greater than $C(i+1)$. We maintain a set of vertices that are adjacent to all vertices fixed so far in the search. This set is called the *working set*. At each level of the search tree, one vertex of the working set is fixed. The new working set consists of the intersection of the vertices adjacent to the fixed vertex and the vertices in the working set. The detailed description of WClique can be found at [84].

5.3 Correcting Protocol Communication Models

The model (3.1) is a binary model in that it only considers conflicts between two links. However, conflicts between more than two links can occur. For example, it is possible that $x \notin \chi(y)$, $x \notin \chi(z)$, and $y \notin \chi(z)$. Thus, according to the binary conflict model, links x, y, and z can all simultaneously active. However, it is possible that the combined interference from y and z, results in enough interference such that transmission across link x fails with high probability. In this case, we say that the links x, y, and z form a multi-conflict. Schedules that use assignments that contain multi-conflicts will have low throughput when deployed. Thus, such assignments should be removed. While the scheme described above removes all binary conflicts, as described next, we remove multi-conflicts only as they arise.

Let $\mathbf{v}^+$ be an assignment found by solving (5.5). $\mathbf{v}^+$ has a multi-conflict if there is a link x with $v_x^+ = 1$ and links $\{y_i : i = 1, 2, ...K\}$ with $v_{y_i}^+ = 1$ and

$$\mathcal{T}(x) > SINR(x, \{y_1, y_2, ..., y_K\}) := \frac{H_{x,x}}{\sum_{i=1}^{K} H_{y_i,x} + \mathcal{N}_0} \tag{5.6}$$

where $\mathcal{T}$ is the SINR threshold for link x. This multi-conflict is defined by the set $C = \{x\} \cup \bigcup_{i=1}^{K} \{y_i\}$. An assignment that maximizes (5.5) and yet does not contain this multi-conflict can be found by solving

$$\begin{aligned} &\max_{\mathbf{v}} \sum_{x=1}^{L} R_x \mu_x v_x \\ &\text{subject to: } \sum_{x \in Q_i} v_x \leq 1 \text{ for } i = 1, 2, ..., M \\ &\qquad \sum_{x \in C} v_x \leq |C| - 1 \\ &\qquad v_x \in \{0, 1\}, \end{aligned} \tag{5.7}$$

where $|C|$ is the number of links in the set C. Intuitively, C should be the smallest set of links that forms a multi-conflict at link x. Solving (5.7) will result in another assignment. If this assignment also has a multi-conflict, then the above problem is further modified. Thus, after N multi-conflicts are found, new assignments are found by solving

$$\begin{aligned} &\max_{\mathbf{v}} \sum_{x=1}^{L} R_x \mu_x v_x \\ &\text{subject to: } \sum_{x \in Q_i} v_x \leq 1 \text{ for } i = 1, 2, ..., M \\ &\qquad \sum_{x \in C_i} v_x \leq |C_i| - 1 \text{ for } i = 1, 2, ..., N \\ &\qquad v_x \in \{0, 1\}, \end{aligned} \tag{5.8}$$

where C_i is the ith multi-conflict.

Note that each time a multi-conflict is found, (5.8) must be resolved. Thus, a large number of multi-conflicts can result in significant computation. The section 6.2.2 finds that only a small number of multi-conflicts arise when forming schedules

in practical mesh networks. Also, note that it is important that the initial set of assignments constructed with Algorithm 3 is free from multi-conflicts.

5.4 The Accuracy of Protocol Models

While the protocol communication models have the significant benefit that the computational algorithms can be analyzed with graph theoretic means, these models suffer from the drawback that they do not accurately represent interference. In order to gauge the impact of these approximations, the physical throughput of the schedules found from the protocol model were computed. That is, suppose that the optimal schedule based on the protocol model resulted in assignment $\mathbf{v}$. In the protocol model, the data rates over link x for this assignment is $R_{\varnothing}(x)$, where the $\varnothing$ denotes that the data rate is based on the assumption that no other node is transmitting.

When there are multiple active links in assignment $\mathbf{v}$, it is unreasonable to ignore the interference because link x may not achieve the normial data rate. Assume the *WithUnsyncACK* physical model is used and the modulation schemes of the data and ACK of link x are (m, n), the actual link data rate is

$$R_{Actual}(SINR^r, SINR^t, m, n) = \frac{PSP_Z(m, SINR^r)\, PSP_{14}(n, SINR^t)}{\frac{1}{Z\times 8}\left(\frac{Z\times 8}{BR(m)} + \frac{14\times 8}{BR(n)} + 2F_{oh}\right)} \tag{5.9}$$

where $SINR^r$, $SINR^t$ are the received $SINR$ at the receiver and transmitter of link x. With these actual link rates, the actual flow rate from the gateway(s) to each destination can be determined.

5.4.1 Adjust Active Link Rate for Optimal Scheduling

Although adding multi-conflicts constraints can solve the multi-conflicts problem, it requires solving more MWIS problems and increases the corresponding computation time. It is possible to correct the multi-conflicts by adjusting the link data rate after we ignore the multi-conflicts and compute the optimal schedule.

Since the $SINR^r$, $SINR^t$ of each active link are known, we can adjust the link modulation schemes and eliminate the multi-conflicts. The modulation adjustment scheme $R_{Opt}(SINR^r, SINR^t)$ is defined as

$$R_{Opt}(SINR^r, SINR^t) = \max_{m,n} R_{Actual}(SINR^r, SINR^t, m, n).$$

The flow rate can be determined from the new adjusted link rates.

5.5 Removing Redundant Assignments from the Set of Considered Assignments

The motivation of this is that if $\mathcal{V}$ is small, then problem (3.2) can be solved quickly. However, as more assignments are added to $\mathcal{V}$, then its size will grow, defeating this goal. Thus, it is useful to remove assignments if they are guaranteed to never be used as active assignments. To see how this is done, suppose that $\mathcal{V}$, the set of considered assignments, yields a set of considered link bit-rates, R, and let r be an element in R. Then, to see if the assignment r can be removed from R, we check whether

$$r \in \text{interior of } Co\left(R \backslash r\right), \tag{5.10}$$

where $Co\left(R \backslash r\right)$ is the convex hull of $R \backslash r$ and $R \backslash r$ is the set R with r removed. If (5.10) holds, then the link bit-rates achieved by r can also be achieved by using a set of bit-rates in $R \backslash r$. Hence, the assignment r can be removed without impacting $Co\left(R\right)$. Figure 3.1 illustrates a redundant assignment that can be removed.

From Theorem 1, we know that the optimal vector of bit-rates is the convex sum of no more than L assignments. Similarly, the vector of bit-rates that solves (3.2) for any set of assignments is also the convex sum of no more than L assignments. Therefore, the set of active assignments, $\mathcal{V}^*\left(\boldsymbol{\mu}^*\right)$, should contain no more than L elements. If $\mathcal{V}^*\left(\boldsymbol{\mu}^*\right)$ does contain more than L elements, then there must be some redundancies in the set of considered assignments, and hence some assignments can be removed.

Determining if (5.10) is true for some $r \in R$ can be determined by solving a linear programming problem. Hence, a check for redundancy requires solving $\#\mathcal{V}$ LP problems. See [89] for details.

5.6 Summary

It is well known that finding an assignment is equivalent to solving a maximum weighted independent set problem in the case of protocol models. In the worst case, MWIS problem is NP-hard. Therefore, this chapter presents two exact methods and two approximation methods to solve the MWIS problem. As we will see in chapter 7, MWIS problem that arises from optimal scheduling in wireless mesh networks can be solved quickly.

Protocol models have a drawback in that the aggregate interference from multiple transmissions is neglected. Thus, two methods of correcting the multi-conflicts are presented. Also, this chapter presents a greedy approach to constructing an initial set of assignments and the techniques to remove the redundant assignments.

Chapter 6

NUMERICAL EXPERIMENTS FOR OPTIMAL SCHEDULING

One of the important aspects of throughput optimization is that in theory, determining the optimal throughput has a theoretical worst-case computational complexity that makes computing throughput even for small networks (e.g., 30 links) intractable with today's computing abilities. However, the theoretical worst-case performance provides little insight into the typical performance that occurs in mesh networks. Thus, it is imperative that the performance be examined in realistic mesh networks. For this reason, this examination employed the UDel Models [90]. Along with a realistic mobility simulator, the UDel Models include a map builder, a realistic propagation simulator, and large collection of data and trace files. The propagation simulator is based on ray-tracing and accounts for reflections off of the ground and off of buildings, transmissions through building walls, and diffraction around and over buildings [91]. It also accounts for the impact that different materials have on reflections off of walls and transmission through walls. Data sets for several urban areas are available online.

This chapter is organized as follows. The methods to generate different kind of network topologies are denoted in section 6.1. Section 6.2 presents several basic experiments for optimal scheduling, such as the number of iterations to converge and the number of multi-conflicts etc. To evaluate the performance of different communication models, a large set of simulations are executed in Section 6.3. Also,

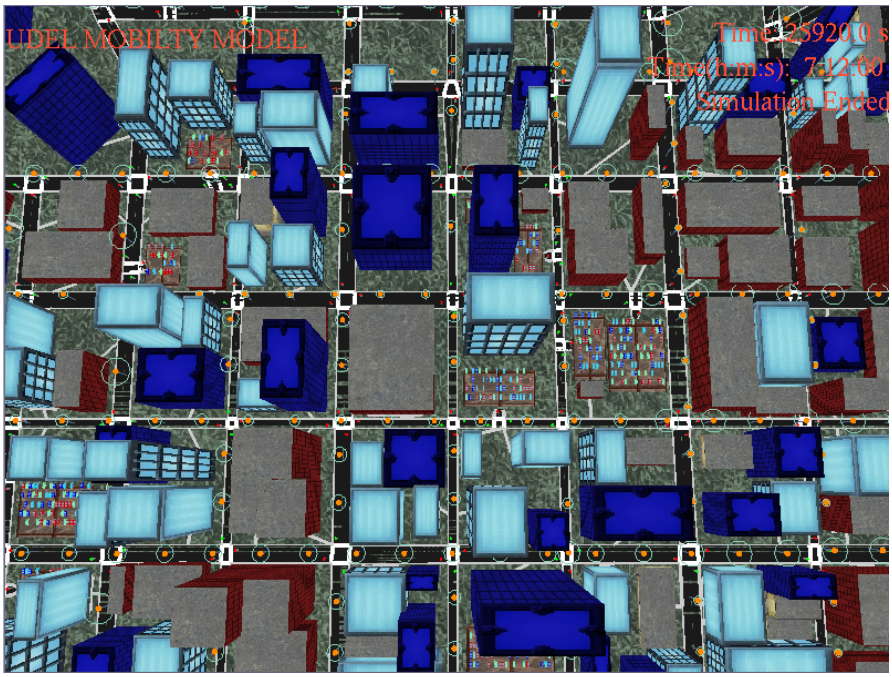


Figure 6.1: A portion of map of simulated region used for determine the performance and behavior of throughput maximization techniques. Mesh routers are shown as orange dots with green circles. The full region is 13x9 blocks.

the techniques to fix the multi-conflict for the SINR Protocol Model are examined. In the section 6.4, the throughput of optimal scheduling is compared with that of 802.11a, and the performance has a great improvement. Section 6.5 compares the performance of different methods to finding the MWIS. Finally, Section 6.6 discusses the impact of the topology on throughput.

6.1 Topology Generation

In order to examine the throughput of urban mesh networks, a large set of simulated urban mesh networks were generated for the future simulations.

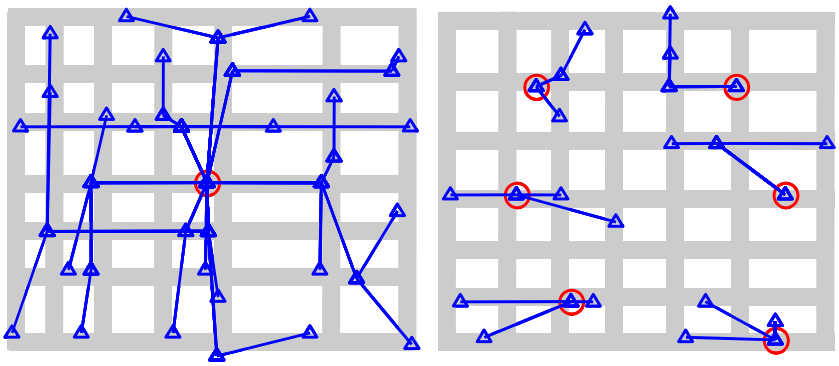


Figure 6.2: 6×6 block region of Downtown Chicago. The mesh routers are displayed as triangles and the gateways are triangles with a circle. The 6×6 block region is randomly chosed from $2km^2$ region of downtown Chicago. The left frame is a network with 1 gateway and 36 mesh routers. The right frame is a network with 6 gateways and 18 mesh routers.

6.1.1 6×6 Block Region of Downtown Chicago - Outdoor Nodes

Many of the results shown or referred to here are derived from a simulation of a 2 km^2 (13×9 block) region of downtown Chicago. Figure 6.1 shows part of the region along with the placement of the outdoor mesh nodes. There are 500 outdoor mesh nodes in the entire region. Each network was based on a 6×6 block region of downtown Chicago that was randomly selected from a 2 km^2 region. Figure 6.2 shows two examples of the 6×6 block region of downtown Chicago. The radio propagation was determined with the UDelModels [90].

In order to estimate the typical performance of an urban mesh network, several different types of topologies were generated, and ten trial topologies were generated for each topology type. The topology types are characterized by the number of wireless mesh routers and the number of wired gateways. In these experiments, all traffic flowed from gateways to destinations, where each mesh router in the topology was a destination of a flow. Mesh routers and gateways were assumed to be

placed on lampposts. Apart of this restriction, mesh routers were uniformly spread throughout the region. The simulated area was partitioned into equal size regions where the number of regions is the same as the number of gateways. A gateway was randomly located within each region. Mesh routers within a region receive packets from the gateway within the same region.

Finally, the number of nodes ranged from 18 to 90 so that the average number of nodes per block ranged from 0.5 to 2.5 in steps of 0.5. The number of gateways ranged from 1 to 6. Since 10 samples of each topology were generated, a total of 300 topologies were used.

Packets were forwarded to their destination over least hop paths. Among paths with the same number of hops, the path selected was the one that had the highest minimum link channel gain, where the minimization is over each hop along the path. Each flow originates at the gateway such that the best route from the gateway to the destination of the flow is no worse than any route from any other gateway.

6.1.2 $2km^2$ Region of Downtown Chicago - Indoor and Outdoor Nodes

We consider the $2km^2$ (13×9 block) region of downtown Chicago with indoor and outdoor nodes. The topology generations are given in details in Section 6.8. In this experiment, all traffic flowed from gateways to destinations, where each mesh router in the topology was a destination of a flow.

For the urban propagation model, nodes were placed to mimic a large infrastructure network. Specifically, outdoors, nodes were placed on lampposts throughout the city, and indoors, enough nodes were placed on each floor so that the entire floor was covered. In all, the baseline set of nodes included over 7000 nodes positioned throughout the city and over 10000 topologies are examined. Here, max-flow routing is deployed to find the path for each flow.

Algorithm 5 Max-Flow routing for a nonzero flow

while Select a destination n such that $F_n > 0$. **do**
  while $F_n > 0$ **do**
    Set $i = 1$, $n_i = n$.
    repeat
      Select an ingress link x_i of router n_i with link data rate $S_{x_i} > 0$.
      if The transmitter n_t of link x_i is not a gateway **then**
        Set $i = i + 1$, Set $n_i = n_t$.
      end if
    until The transmitter n_t of link x_i is a gateway
    Set $r = \min(F_n, S_{x1}, S_{x2}, ...)$. Set $S_{x_i} = S_{x_i} - r$. Set $F_n = F_n - r$.
  end while
end while

6.1.2.1 Max-Flow Routing

Interference aware, multi-path max-flow routing is found by solving

$$\max_{\mathbf{S},F} F \tag{6.1a}$$

$$\sum_{\{x:x_t=w\}} S_x - \sum_{\{y:x_r=w\}} S_y + F = 0 \text{ for } w \notin GW \tag{6.1b}$$

$$\frac{S_x}{r(x)} + \sum_{y\in\chi(x)} \frac{S_y}{r(y)} \leq 1 \text{ for all } x, \tag{6.1c}$$

where S_x is the flow over link x. Note this optimization problem approximates the impact of interference. Specifically, $\frac{S_x}{r(x)}$ is the fraction of time that link x transmits, and hence (6.1c) ensures that the fraction of time that link x transmits and the fraction of times that all links that interfere with link x transmit sum to no more than one. Of course, it is possible that some links that interfere with x can transmit simultaneously. But (6.1c) does not account for this possibility. Thus, (6.1) provides a lower bound on the throughput. It should be pointed out that while problem (6.1) is polynomial, solving (6.1) was, by far, the computational bottleneck of this investigation.

Algorithm 6 Greedy Method to Select Single Path

Let S_x be the optimal flow rates that solve (6.1) and set $\mathcal{W} = \varnothing$.
repeat
  Randomly select $w \in \mathcal{N}$ and $w \notin \mathcal{W}$.
  Set $\mathcal{W} = w \cup \mathcal{W}$.
  $\boldsymbol{P}(w) = \arg\max_{\{p \in \mathcal{P} | p \text{ is a path to } w\}} \min_{x \in p} S_x$, i.e., $\boldsymbol{P}(w)$ is the path that results in the highest flow to w.
  $\boldsymbol{S}(w) = \min\left(F, \min_{x \in P(w)} S_x\right)$.
  Set $S_x = S_x - \boldsymbol{S}(w)$ for each $x \in \boldsymbol{P}(w)$.
until
  $\mathcal{W} = \mathcal{N}$.

Problem (6.1) results in multipath routing. There are many different routings that accommodate a given link flow rate S_x and the connection rates F. One simple approach to finding paths from the gateways to mesh routers is given in Algorithm 5. Note that the above scheme may result in multiple paths between a single source-destination pair.

Single path routing can be formed by quantization as follows. Define $\mathcal{P}(w)$ to be the set of paths from some gateway to node w. Then the greedy algorithm shown in Algorithm 6 is used to construct $\boldsymbol{P}(w)$, a path from some gateway to node w.

6.2 Results for Optimal Scheduling

Topologies generated in Section 6.1.2 and the SINR Protocol Model with *Unsynchronized Acks* are deployed to evaluate the performance of optimal scheduling. These topologies included the outdoor lamppost-mounted nodes along with indoor infrastructure nodes. Nodes were randomly selected so that the network was connected and each node had approximately six neighboring nodes with which it can communicate at 24 Mbps using 802.11a. This node density resulted in the conflict graph having a degree of between 15 and 20. Once the nodes were selected, a set of gateways was selected so that the number of gateways equals the number of nodes

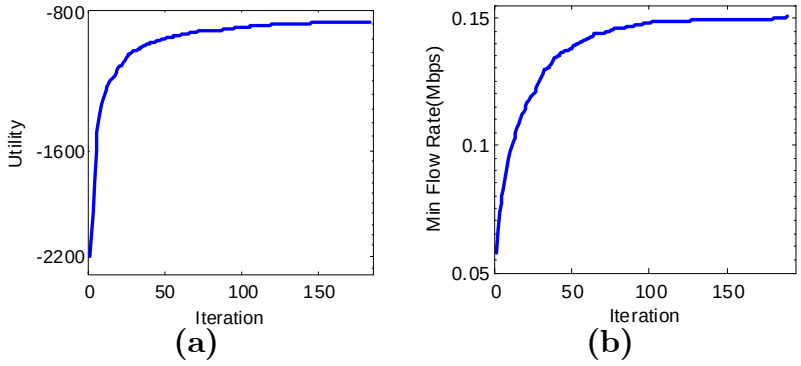


Figure 6.3: Variation in the computed throughput as assignments are added. In (a) the throughput is the total utility, i.e., $\sum_{\phi\in\Phi}\log(f_\phi)$. In (b) the throughput is $\min_{\phi\in\Phi} f_\phi$. These plots are for a 1024 node (992 link) topology.

divided by 32. The gateways were selected such that they were uniformly distributed. In this way, topologies were made with 64, 128, 256, 512, 768, 1024, and 2048 nodes. For each number of nodes, 40 sample topologies were generated.

6.2.1 Number of Iterations until Algorithm 1 Stops

Figure 6.3 shows how, in the 1024 node (992 link) topology, the throughput increases as the more assignments are added. The point of maximum throughput occurs when the solution to the ILP (5.5) does not satisfy (3.9) or the stopping condition specified in Algorithm 1 is met. Thus, in this case, Algorithm 1 stopped after 186 iterations when the throughput metric was $G(\mathbf{f}) = \sum_{\phi\in\Phi}\log(f_\phi)$, and after 191 iterations when the throughput metric was $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$. When $G(\mathbf{f}) = \sum_{\phi\in\Phi}\log(f_\phi)$, the stopping condition used $\rho = 0.15$, while for the case of $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$ we used $\rho = 0.05$. Note that since the objective functions are different, the values of ρ should not be compared.

As can be observed, the number of iterations is approximately the same for both objective functions. Figure 6.4 explores this behavior in more detail and shows that the average number of iterations over 40 topology samples is approximately

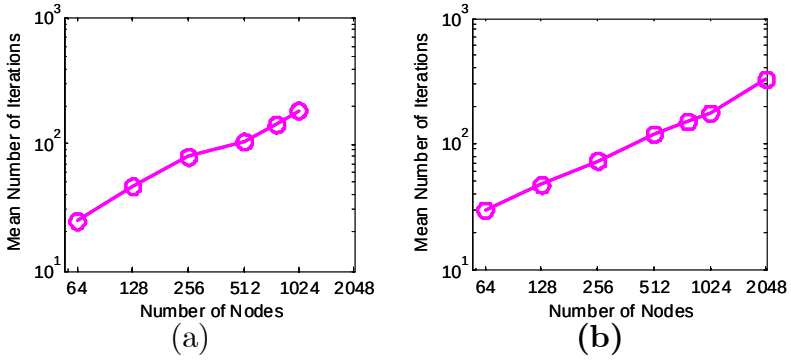


Figure 6.4: Number of iterations until Algorithm 1 stopped. In (a) $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$ and $\rho = 0.15$ In (b) $G(\mathbf{f}) = \min_{\phi\in\Phi} f_\phi$ and $\rho = 0.05$.

the same for both objective functions. Moreover, since the log-log scale is used, Figure 6.4 indicates the number of iterations increases polynomially with the number of links. Note that Figure 6.4 only shows the case of $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$ for topologies up to 1024 nodes. Due to numerical difficulties, we were not able to solve (3.2) for 2048 nodes even for a small number of assignments. Thus, we conclude that when $G(\mathbf{f}) = \sum_{\phi\in\Phi} \log(f_\phi)$, the computational bottleneck is not finding new assignments, but solving the basic nonlinear optimization (3.2).

Note that only one assignment is added at each iteration. Thus, the maximum number of elements in $\mathcal{V}$ is the number of assignments found in Algorithm 3 plus the number of iterations required by Algorithm 1. Hence, we have achieved the goal of determining the solution to (3.2) for $\mathcal{V} = \bar{\mathcal{V}}$ by computing the solution to (3.2) for a small set $\mathcal{V}$. The complexity of solving linear and nonlinear optimization problems is well known, and is not investigated here.

6.2.2 The Number of Multi-Conflicts

As mentioned in Section 5.3, in order for the throughput found by solving (3.2) to match the actual throughput when the schedule is deployed, the assignments used in the schedule must not have any multi-conflicts. The scheme discussed in Section 5.3 can be used to remove the multi-conflicts.

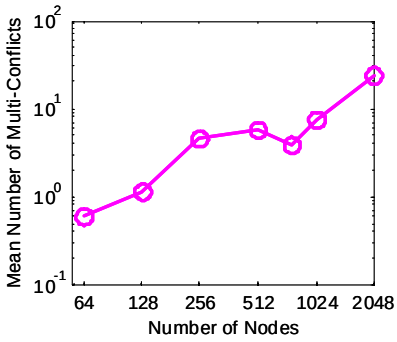


Figure 6.5: The average number of multi-conflicts detected and removed for topologies of different sizes.

However, each time a multi-conflict is detected and removed, an ILP problem (5.8) must be solved, increasing the overall computation time. Figure 6.5 shows the average number of multi-conflicts found (and removed) for various sizes of networks. Roughly, the number of multi-conflicts grows with the number of nodes and the number of gateways. Comparing Figure 6.5 to Figure 6.4 we observe that the number of multi-conflicts is much smaller than the total number of iterations. On the other hand, failing to remove multi-conflicts can severely impact the throughput when the schedule is deployed.

6.2.3 Time to Perform Clique Decomposition

As discussed in Section 5.2.4.1, the time to find a new assignment is greatly reduced if a clique decomposition is performed first. Figure 6.6 shows that the time required to perform this decomposition is on the order to the time it takes to perform one iteration of Algorithm 1. Since Algorithm 1 requires that tens or hundreds of iterations are performed, the time to compute a single clique decomposition is negligible. However, we do not recompute the clique decomposition every time a multi-conflict is found.

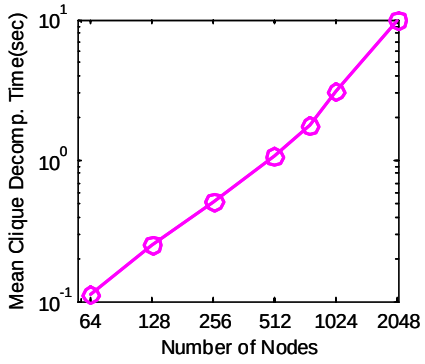


Figure 6.6: Time to compute a clique decomposition as a function of the number of nodes in the network.

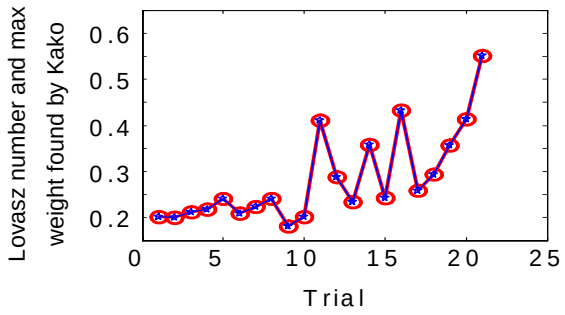


Figure 6.7: The Lovasz number $\mathcal{V}(G, w)$ and the weight of the largest found independent set after Algorithm 1 has converged. Since these two values are the same, only a single set of points can be seen. The equality these numbers indicates that the computed schedules are optimal. The results shown are for 90 node networks.

6.3 Results for Communication Models

6.3.1 Optimality of Schedules Based on Protocol Models.

90 node topologies generated in Section 6.1.1 are examined in this subsection. As mentioned in Section 5.2.2, the Lovasz number, $\mathcal{V}(G, w)$, is an upper bound on $\alpha(G, w)$, the weight of the maximum weighted independent set. Thus, if maximum assignment cannot be found that satisfies the inequality of (3.9) and has a weight that is the same as $\mathcal{V}(G, w)$, then the optimal schedule has been found. On the one hand, in general, $\mathcal{V}(G, w)$ is only an upper bound, and hence, the equality of $\mathcal{V}(G, w)$ and weight of a found independent set is only a sufficient condition for the schedule being optimal. On the other hand, there are a large class of graphs (e.g., perfect graphs) where the $\mathcal{V}(G, w) = \alpha(G, w)$. In these cases, the equality of the Lovasz number and weight of the found independent set is a necessary and sufficient condition for the schedule being optimal. It is not known whether the graphs that arise in urban mesh networks are such that $\mathcal{V}(G, w) = \alpha(G, w)$.

However, Figure 6.7 shows the $\mathcal{V}(G, w)$ and the weight of the independent set found once Algorithm 1 had converged (i.e., we were unable to find an assignment such that $\sum \mu_x R(\mathbf{v}, x) > \lambda$). As can be observed, we have $\mathcal{V}(G, w) = \alpha(G, w)$, ensuring that the Algorithm 1 has indeed converged to the optimal schedule.

6.3.2 Performance of Communication Models

Figure 6.8 shows the theoretical and actual throughput for Node Exclusive, 2-hop Node Exclusive, and Sensing models. The theoretical and actual throughputs are the results from the optimal schedule that uses nominal link bit rate and actual link bit rate, respectively. The throughput is averaged over 40 samples for each topology, and the number of gateways is equal to the number of nodes divided by 32. One complication with the Sensing model is that the *Channel Sensing Threshold* must be determined. In Figure 6.8, *Channel Sensing Threshold*$= -90$dBm. All of the models provide high theoretical throughputs. However, since these models

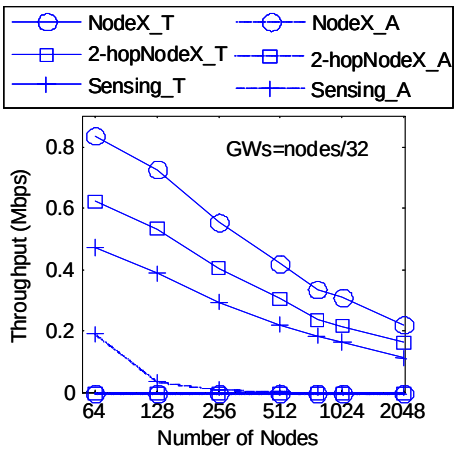


Figure 6.8: Computed theoretical and actual throughputs for NodeX, 2-hop NodeX and Sensing models where the number of Gateways is equal to number of nodes divided by 32.

do not accurately represent the interference, it is not surprising that the actual throughputs are very poor. Specifically, Node Exclusive and 2-hop Node Exclusive models always have zero actual throughput, and Sensing model shows a small actual throughput when the network size is small and zero otherwise. The schedules from the Node Exclusive and the 2-hop Node Exclusive Models consistently result in no data traversing some links. Hence, the high theoretical throughput offered by the node exclusive model is fictitious.

Figure 6.9 shows the theoretical and actual throughputs for SINR Protocol Model with *Unsynchronized ACKs* , *Synchronized ACKs* and *Without ACKs*, where the multi-conflicts are ignored and the MinAck selection scheme is used if ACK applies. As compared to using ACKS, *Without ACKs* eliminates overhead of ACK transmission, reduces the interference induced from ACK, and hence, achieves more spatial multiplexing. For these reasons, *Without ACKs* outperforms the schemes using ACKs. The *Synchronized ACKs* case slightly outperforms the *Unsynchronized ACKs* because the synchronization eliminates the interference between data and

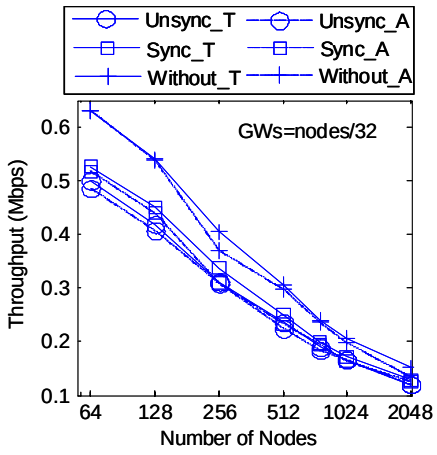


Figure 6.9: Computed theoretical and actual throughputs for SINR Protocol Model with *UnsyncAck*, *SyncAck* and *WithoutAck*.

ACK packets. The actual throughput of SINR Protocol Model is very close to the theoretical throughput. The reason for this behavior is that SINR Protocol Model accurately represents the interference.

6.3.3 Performance of Correcting Multi-conflicts and Adjusting Bit-Rates

There are two methods to correct the multi-conflicts for SINR protocol model. One way is to add multi-conflicts constraint, named as fixing multi-conflicts, during the process to find the optimal schedule. The other way is to ignore multi-conflicts during the schedule optimization, but then adjust the link bit rates once the schedule and SINRs are known.

We consider three options to choose data and ACK transmission rates, namely MinAck, which uses the minimum ACK rate, SameAck, which uses the same data and ACK rate, and OptDataAck which might use the different data and ACK rates and selects the bit-rates to minimize the transmission time across a link (including retransmissions). These different methods correspond to the different ways to select the bit-rate for ACKs used in (4.2).

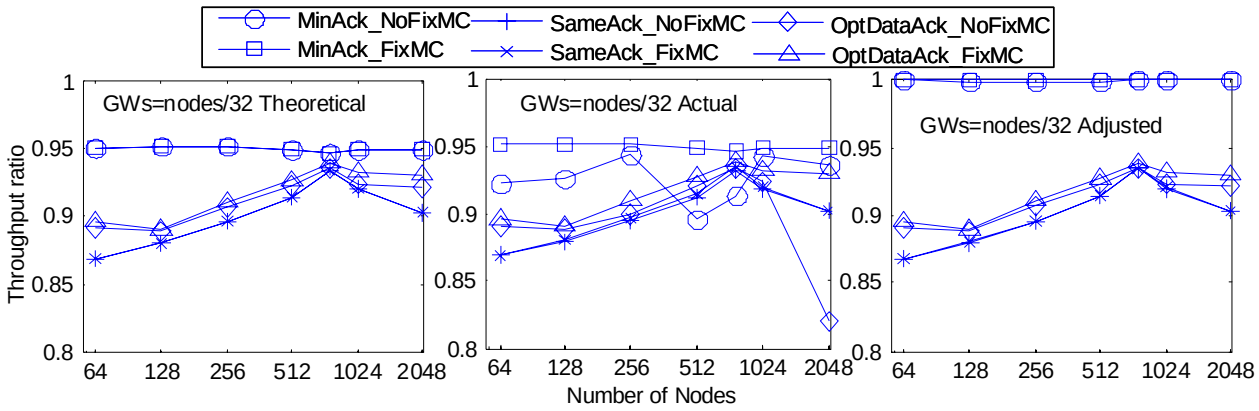


Figure 6.10: (a) The theoretical throughput for modulation selection schemes MinAck, SameAck and OptDataAck with fixing multi-conflicts or not (b) The actual throughput (c) The adjusted throughput

It should be emphasized that in this section we consider adjusting the bit-rates at two different stages. First, before the schedule optimization, the nominal data and ACK bit-rates are selected as described in Section 4.2.4.2. Then, once the schedule has been computed, the bit-rates can be adjusted for each link and each assignment. In the first case, the bit-rates are selected based on SNR; while in the second case, the bit-rates are selected based on SINR. Also, in the first case, three possible ways of selected data and ACK bit-rates are considered, while in the second case, only the OptDataAck scheme is used. The reason for considering different schemes in the first case is that different schemes might result in different amounts of spatial multiplexing. However, once the schedule is determined, the spatial multiplexing is fixed.

Figure 6.10 shows the theoretical, actual, and adjusted throughputs for the three link modulation selection schemes where for each scheme multi-conflicts are either eliminated or not. The throughputs are normalized as follows. For each topology, the maximum actual throughput found over all six schemes but where the bit-rates are adjusted after the schedule has been determined. The right-hand frame shows the throughputs after the bit-rates have been adjusted. Thus, due to

normalization, some of the throughputs are one.

The left-hand frame in Figure 6.10 shows the theoretical throughput is not affected by whether multi-conflicts are fixed or not. On the other hand, the middle frame in Figure 6.10 shows that the actual throughput of the schemes when multi-conflicts are ignored can be smaller than the theoretical throughput. For example, for the MinACK scheme, the actual throughput is 3-5% less than the theoretical, but for the OptDataACK scheme, the actual throughput is reduced by 12% when the topology has 2048 nodes. Also, as expected, the actual throughout is same as the theoretical throughout if we fix the multi-conflicts.

The right-hand frame of Figure 6.10 shows that the throughput after the bit-rates are adjusted. In this case, there is minimal difference between accounting for multi-conflicts or not. Thus, in terms of throughput, as long as bit-rates are adjusted after the schedules are computed, multi-conflicts can be ignored. On the other hand, according to the algorithm described in Section 5.3, every time a multi-conflict is discovered, the MWIS problem must be changed and resolved. This can greatly increase the computational complexity. Thus, we conclude that multi-conflicts can be ignored.

Note that in the case of MinACK, the actual throughput after adjusting the bit-rates is about 5% larger than the theoretical throughput. This behavior is due to suboptimal selection of the nominal bit-rates, which is corrected after the schedule is computed. This shows the utility of the simple procedure of adjusting bit-rates after schedules are computed.

Finally, note that the MinACK scheme achieves higher throughput than the other schemes. This behavior is expected since increasing the ACK rate does not greatly impact the effective data rate, but may significantly impact the set of conflicting links, and hence impact spatial multiplexing.

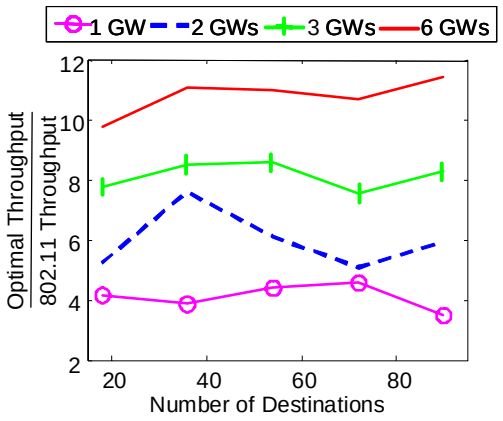


Figure 6.11: Comparison between optimal scheduling and 802.11 in mesh networks covering 6×6 block regions of downtown Chicago.

6.4 Comparison with 802.11 CSMA/CA

With the ability to compute optimal schedules, the impact of optimal schedules on the throughput as compared to 802.11 with CSMA/CA can be investigated. Figure 6.11 shows the ratio of the optimal throughput to the throughput that 802.11 CSMA/CA can achieve. Here the throughput metric is $\min_{\phi \in \Phi} f_\phi$ and the SINR protocol model with *Without Acks* is used. Qualnet was used to estimate the throughput of 802.11. RTS/CTS and Qualnet's automatic rate fallback scheme were used. The 802.11 CSMA/CA throughput was determined by sending data to each destination at a constant rate (1000 B packets were used). The sending rate was adjusted until the maximum of $\min_{\phi \in \Phi} f_\phi$ was found. Confidence intervals were generated via bootstrapping [92] to ensure that the estimated throughput was accurate within 10%.

The topologies generated in Section 6.1.1 are used in this section. Figure 6.11 shows that for networks with a large number of gateways, optimal scheduling can have a dramatic improvement in the throughput. On the other hand, the simulations used a standard version of 802.11 CSMA/CA. It is conceivable that if 802.11 is better

tuned (e.g., by tuning CCA [93]) and a better version of the ARF is used, then the throughput with 802.11 CSMA/CA could be improved and the relative improvement provided by optimal scheduling would be reduced.

Figure 6.11 provides a baseline for the range of improvement in throughput that optimal scheduling can achieve. While it is expected that scheduling results in a higher throughput, the degree of the improvement and the dependence on the topology have been unknown. Figure 6.11 indicates when there are a large number of gateways, scheduling tends to provide tremendous improvements in throughput over 802.11 with CSMA/CA. The scale of this improvement motivates further research on scheduling for networks with many gateways and perhaps supports the extra cost required to deploy hardware capable of performing scheduled transmissions. For example, improvements of this size are large enough that optimal scheduling will likely still provide considerably higher throughput when factors like overhead and errors due to synchronization are accounted for and CSMA/CA is well tuned.

On the other hand, Figure 6.11 indicates potential difficulties with improving the throughput on networks with few gateways. For example, [20] developed a scheme that achieves at least 1/3 of the optimal throughput (under the condition that co-channel interference does not arise). Figure 6.11 indicates that such a scheme will only slightly improve the throughput on networks with a small number of gateways. However, improvements of that size might also be possible by tuning CSMA/CA.

6.5 Performance of Searching Algorithms for MWIS

In this investigation, the approximation algorithms Kako and WMIN are implemented with Matlab. One of the exact methods, maximum weighted clique developed by Östergård, is implemented with C code. As for the other exact method, integer programming, two solvers from the Tomlab toolbox are examined. One solver is called binary programming, and the other solver is called CPLEX which is a mixed

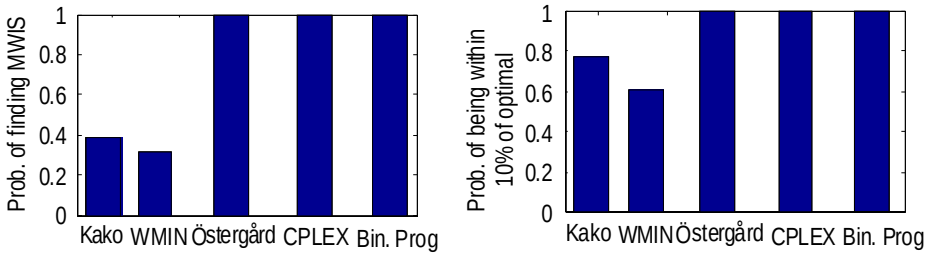


Figure 6.12: Performance of techniques to compute the maximum weighted independent set. Left: The probability of finding the MWIS. Östergård's methods, CPLEX, and binary programming are exact methods, and hence always compute the exact MWIS. Right: The probability of the found independent set having a weight within 10% of optimal.

integer programming solver. Also, the SINR protocol model with *Without Acks* is used here.

Figure 6.12 compares the performance of techniques to compute an (approximate) maximum weighted independent set for the realistic networks. The left frame of Figure 6.12 shows that all exact methods always find the optimal MWIS and the approximation methods have less than 40% probability to compute the optimal MWIS. Furthermore, the method Kako is a little better than the method WMIN. When we consider the probability of finding the weighted independent set within 10% of optimal, Kako and WMIN have almost 80% and 60% probabilities, respectively, which are shown in the right frame of Figure 6.12. Therefore, Kako performs better than WMIN in searching the MWIS.

The average computation time of MWIS is another factor that impacting the choice of the searching algorithm. Since the results from Chapter 7 show that the MWIS problem can be solved quite fast with CPLEX method for moderate sized networks, for example, it takes about 1 sec for the topology with 2048 nodes, it is not necessary to consider the approximation methods anymore. Therefore, the CPLEX method is deployed in the following investigation.

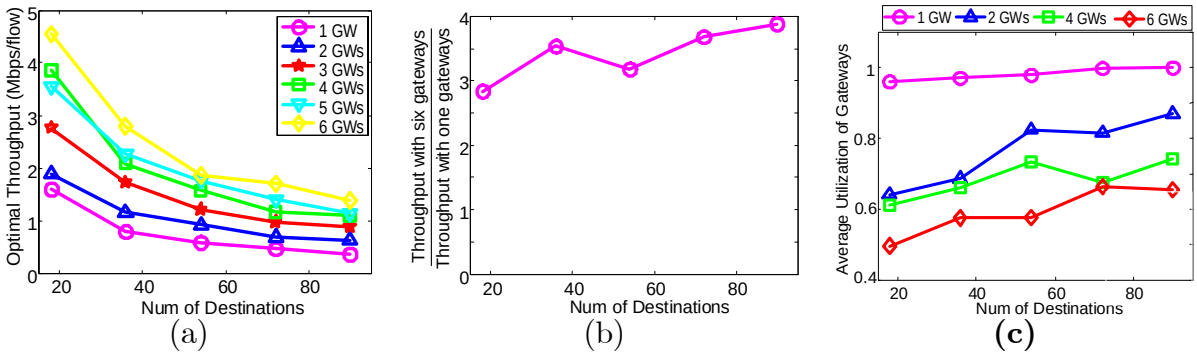


Figure 6.13: (a) Throughput of mesh network infrastructure in a 6x6 block region for various numbers of wired gateways and various numbers of mesh routers/destinations. (b) Ratio of the throughput with six wired gateways and the throughput with one wired gateway. (c) Average utilization of each gateway.

6.6 The Impact of the Topology on Throughput

Here, the topologies generated in Section 6.1.1 are deployed and the SINR protocol model with *Without Acks* is used. Figure 6.13 (a) shows the optimal throughput (Mbps/destination) for a wide range of topologies. The general trends are expected. As the number of destinations increases, the data rate per destination decreases and as the number of gateways increases, the data rate per destination increases. Recall that there are 36 blocks in the simulated region. Thus, Figure 6.13 (a) shows that if there is one gateway for every six blocks, and 2.5 mesh routers/destinations for each block (i.e., 90 destinations in 36 blocks), then the infrastructure can provide approximately 1.5Mbps per destination.

Also, Figure 6.13 (a) shows that the throughput increases as gateways are added. A more detailed view of this behavior is given in Figure 6.13 (b). This figure shows that as more destinations are added, the impact of adding more gateways increases, however, it increases rather slowly.

Figure 6.13 (c) shows the average utilization of each gateway in the network. Ideally, each gateway transmits at all times. When there is only a single gateway, this ideal is nearly achieved. Recall that the communication model used here does

not utilize ACKs; only one-way communication is required. Note that with TDMA, even when packets must be transmitted over multiple hops, the gateway is able to nearly continuously transmit. For example, suppose that the gateway transmits to node A during the first time slot and to node B during the second time slot. Furthermore, during the second time slot, node A forwards the packet received during the first time slot to its neighbor, node C. If node C is far enough away from the gateway and node B is far enough away from node A, then all transmissions will succeed. Of course, if node A requires an ACK from node C, this ACK would likely not be received since the interference from gateway's transmission to node B would likely be substantial. Thus, ACKs may decrease the network throughput.

6.7 Conclusion

This chapter evaluates the performance of practical techniques for computing optimal schedules in multihop wireless networks even when co-channel interference arises. The algorithms can compute optimal schedules within a few minutes for networks with 2048 nodes and within a few seconds for networks with 128 nodes.

Node exclusive model, 2-hop node exclusive model and sensing model, which are widely used in the research, exhibit poor performance because of neglecting co-channel interference. The SINR protocol model proposed in this work accurately models the interference and the multi-conflicts can be removed easily.

The performance improvement provided by the optimal scheduling is significant if there are a large number of gateways. For example, as compared to 802.11's CSMA/CA, optimal scheduling improves performance by a factor between 3 and 11, with the improvement increasing as the density of gateways increases.

6.8 Appendix: Construction of Random Wireless Networks

In order to investigate the average computational complexity of the MWIS problem for optimal scheduling in practical wireless networks, statistics must be generated from a large number of networks. This investigation focuses on topologies that might arise in large scale wireless mesh networks. Such infrastructure networks are composed of wireless routers and gateways, which have both wired and wireless interfaces. Such networks have densely distributed wireless routers while gateways are more lightly distributed. The routing forms a forest, where gateways are roots of the trees.

In this investigation, five parameters are used to characterize a mesh network. Four of these parameters are the number of the nodes, the density of nodes (i.e., how many neighbors a node has), the density of the gateways, and the target bit-rate of links. The propagation environment also plays an important role in the performance of a network. Thus, in order to fully explore the computational complexity, we consider three popular propagation environments, namely, the two-ray model, the two-ray with shadow fading model, and a realistic urban propagation model. Thus, the propagation model is a fifth parameter that controls the topology. The next subsections detail the generation of random topology based on these five parameters.

6.8.1 Propagation Models

Propagation is a key aspect of wireless networks. In the two-ray propagation model, the received signal strength (in dB) at a node that is d meters from a transmitting node is

$$P_{\text{2Ray}}(d) = 20\log_{10}\left(\frac{\lambda}{4\pi}\right) + P_{\text{TransmitPower}}\,[\text{dB}] - \begin{cases} 20\log_{10}(d) & \text{for } d \le C \\ 40\log_{10}(d/C) + 20\log_{10}(C) & \text{for } d > C \end{cases},$$

where C is a parameter that depends on the node height. In the case of hand-held radios, the height is approximately 1.5m, and $C = 225m$. Throughout this work, $P_{\text{TransmitPower}} = 18$ dBm and $\lambda = 0.125$m, as is the case for 802.11b/g. When shadow fading is added, the received signal strength (in dB) at a node that is d meters from a transmitting node is

$$P_{\text{2RayAndShadowing}}(d) = P_{\text{2Ray}}(d) + X$$

where X is a Gaussian random variable with mean 0 and standard deviation 4 dB [118]. We assume that nodes are spaced far enough apart that the random part of the propagation are independent. However, propagation is symmetric [118].

Due to the difference between indoor and outdoor propagation, and due to wave guide effects of streets, urban propagation is distinct from the random propagation models. Thus, in order to investigate the performance of the complexity of optimal scheduling in urban areas, the UDel Models Propagation Simulator [90] was employed. Specifically, for this study, ray-tracing was performed on a 2 km^2 region of downtown Chicago. This computation provided the received signal strength between any pair of nodes.

Topologies were randomly generated by selecting a subset of nodes from a large *baseline set of nodes*. In the case of the two-ray propagation model and the two-ray with shadowing model, the baseline set of nodes were densely distributed so that within a 15 km^2 region 5000 nodes where distributed. However, for the urban propagation model, nodes were placed to mimic a large infrastructure network. Specifically, outdoors, nodes were placed on lampposts throughout the city, and indoors, enough nodes were placed on each floor so that the entire floor was covered. In all, the baseline set of nodes included over 7000 nodes positioned throughout the city.

6.8.2 Random Topology Generation

Node Selection

Beyond the propagation model, four parameters are used to construct a topology, namely, n the number of nodes, r^* the target bit-rates, Δ the target number of neighbors, and NGW the number of gateways. The target bit-rate corresponds to a specific received signal strength. Letting $RSS(r)$ be the minimum required received signal strength to decode a transmission at data rate r, then using 802.11g's coding and modulation, typical values of RSS are

$$RSS(6) = -90\text{dBm};\ RSS(12) = -87\text{dBm};$$
$$RSS(18) = -84\text{dBm};\ RSS(24) = -81\text{dBm};$$
$$RSS(36) = -78\text{dBm};\ RSS(48) = -74\text{dBm};$$
$$RSS(54) = -72\text{dBm},$$

where the data rates are in Mbps. We say that two nodes are neighbors if the propagation model results in a received signal strength that is above $RSS(r^*)$.

Let $\mathcal{N}$ denote the set of nodes in the topology. Initially, $\mathcal{N}$ is a single node selected at random. Then, a node is selected at random among all the nodes that satisfy 1.) the node has between 1 and Δ neighbors in $\mathcal{N}$, and 2.) adding the node to $\mathcal{N}$ will not make any node in $\mathcal{N}$ have more than Δ neighbors in $\mathcal{N}$. If no such node exists, then the process is restarted. If suitable nodes do exist, the process continues until $\mathcal{N}$ has n elements.

Next, gateways are selected. The objective is that the gateways are uniformly spread throughout the network in the sense that the average distance from a node to the closest gateway is minimized. This is formulized as follows. Given a set of gateways, $\mathcal{G}$, a new gateway is added by finding the node, w, that minimizes $D(u)$ where

$$D(u) = \sum_{w \in \mathcal{N}} \min\left(d(u,w), \min_{g \in \mathcal{G}} d(g,w)\right)$$

Algorithm 7 Selecting the Gateways

1: Let $\mathcal{G}$ be a randomly selected set of NGW nodes from $\mathcal{N}$.
2: **repeat**
3: Set $\mathcal{G}' = \mathcal{G}$.
4: Remove the node from $\mathcal{G}$ that has been in $\mathcal{G}$ for the most iterations.
5: Set $\mathcal{G} = \mathcal{G} \cup \arg\min_{w \in \mathcal{N} \setminus \mathcal{G}} D(w)$.
6: **until**
7: $\mathcal{G}' = \mathcal{G}$.

where $d(u, w)$ is the distance in hops from node u to node w. Thus, the gateways are selected in Algorithm 7.

Note that the above is not a convex optimization and hence the final set of gateways might depend on the initial selection of gateways. Thus, the above algorithm was run ten times and the set of gateways that resulted in the smallest value of $\sum_{w \in \mathcal{N}} \min_{g \in \mathcal{G}} d(g, w)$ was used.

Routing

Once the wireless routers and gateways have been selected, the routing was determined. As mentioned above, the routing forms a forest, where each gateway is a root of a tree and each wireless router is in exactly one tree. While there are several approaches for routing, this investigation used a max-flow-based, interference aware routing.

The first step in forming routes is to identify the set of potential links, their bit-rates, and the links that they interfere with. Let x denote a link with transmitter x_t and receiver x_r and let P_x be the received signal strength at the receiver. The bit-rate used by link x is denoted $r(x)$ and is given by

$$r(x) := \max\{r : P_x - PGuard > RSS(r)\},$$

where $PGuard$ is used as a buffer to reduce sensitivity to interference. This study used $PGuard = 3$ dB. If no such bit-rate exists (i.e., $P_x - PGuard < -90$dBm),

then the link is removed from consideration. Given the bit-rates, for each link x, the set of conflicting links, $\chi(x)$ can be found, as described in Section 4.2.4.2. The maximum-flow routing is detailed in Section 6.1.2.1.

Chapter 7

PRACTICAL COMPLEXITY OF SOLVING MAXIMUM WEIGHTED INDEPENDENT SET

7.1 Introduction

Optimal scheduling requires solving a graph theoretic problem known as the maximum weighted independent set (MWIS) problem. Thus, the complexity of optimal scheduling is tied to the complexity of MWIS problem. This chapter examines the computational complexity of solving MWIS in practical wireless networks.

In general, the MWIS problem is NP-hard [94]. Moreover, it is NP-hard to approximate the MWIS with an approximation ratio of $n^{1-\varepsilon}$, for $\varepsilon > 0$, where n is the number of vertices in graph [95]. On the other hand, there are many classes of graphs where the MWIS problem has polynomial complexity. For example, MWIS can be found in polynomial time of perfect graphs[71], disk graphs [72], circle graphs [77], trees [75] and as well as many families of graphs that are free of particular subgraphs [73, 96]. The MWIS problem is also solvable in polynomial time on line graphs. In wireless scheduling, line graphs arise when there is no co-channel interference.

Beside the restrictive case where there is no co-channel interference and the case of networks restricted to one-dimension (e.g., roads), it is unknown whether the MWIS problems that arise in practical wireless scheduling have any special properties that make them solvable in polynomial time. Nonetheless, through extensive computational experiments we have found that the MWIS that arises in practical wireless scheduling can be solved quickly. As shown in Section 7.3, the MWIS that

arises when computing the optimal schedule for a wireless network with 2048 nodes can be computed in approximately one second. This work will also demonstrate that the number of nodes and the average degree of the conflict graph (defined in Section 5.2.2) are good predictors of the computation time. Other factors such as node density and bit-rate impact the average degree of the conflict graph, and hence do not need to be considered beyond their impact on the degree of the conflict graph. Moreover, computational evidence indicates that the computation time grows polynomially with the size of the network if the mean degree of the conflict graph is fixed.

The remainder of this Chapter proceeds as follows. The next section provides a brief overview of problems with worst-case exponential complexity that have been found to be easily solved in practice. Sections 7.3 - 7.5 present the results of computational experiments involving over 10000 topologies. Finally Section 7.6 provides concluding remarks. Note that the topologies generated in Section 6.1.2 are deployed, the SINR protocol model with *Unsynchronized ACKs* is used, and the ILP with CPLEX solver described in Section 5.2.4.1 is used to solve the MWIS in this investigation.

7.2 Worst-Case and Average Complexity

The results of the computational experiments presented below indicate that optimal scheduling is practical. In particular, the MWIS problem that arises in wireless scheduling can be quickly solved in practice. These results are not in contradiction with earlier proofs of NP-hard, but rather are well aligned with the practical computational complexity in a wide range of other NP-hard or exponential problems. For example, let us consider linear programming. In [97], it is shown that in the worst-case, the computational complexity of the simplex algorithm is exponential in the size of problem. On the other hand, there is an abundance of evidence that in practice, the computational complexity of the simplex algorithm is $m^2 \times n$ where

m is the number of constraints and n is the number of variables [70]. Moreover, interior point methods have been developed that have polynomial complexity for the worst case. However, despite the fact that interior point methods have a better worst-case performance, state-of-the-art solvers such as CPLEX [88] and XPress [98] use the simplex method. In summary, there may be a substantial difference between worst-case computational complexity and practical computational complexity.

While there are many ways to quantify practical computational complexity, one common approach is to use the average complexity [99]. By this definition of complexity, several problems that are NP-hard in the worst case, are polynomial on average. For example, in graph theoretic problems, average complexity is the complexity averaged over solving the problem over random graphs. A random graph is one where an edge between any two vertices exists with probability p. Under this definition, it has been shown that the average complexity of finding Hamilton Cycles and solving the edge coloring problem are polynomial [100, 101]. In the case of the maximum independent set (MIS) problem, there exists an algorithm with average complexity of $\sum_{k=1}^{n} \binom{n}{k} 2^{-k(k-1)/2} = O\left(n^{\log(n)}\right)$ on random graphs with $p = 1/2$, where there are n nodes in the graph [102].

The SAT problem, which is NP-hard in general, is another example of a significant difference between practical computational complexity and worst-case computation complexity. While the authors are not aware of any proofs on average complexity, several researchers have developed algorithms that can quickly solve randomly generated problems [103, 104, 105, 106]. Due to the importance of the SAT problem, the average complexity of the SAT problem has been extensively studied. One finding is that the distribution of the problems plays an important role in the average complexity of SAT problem [107].

The impact of the distribution is troublesome since it indicates that the complexity averaged over a specific distribution of problems might be significantly

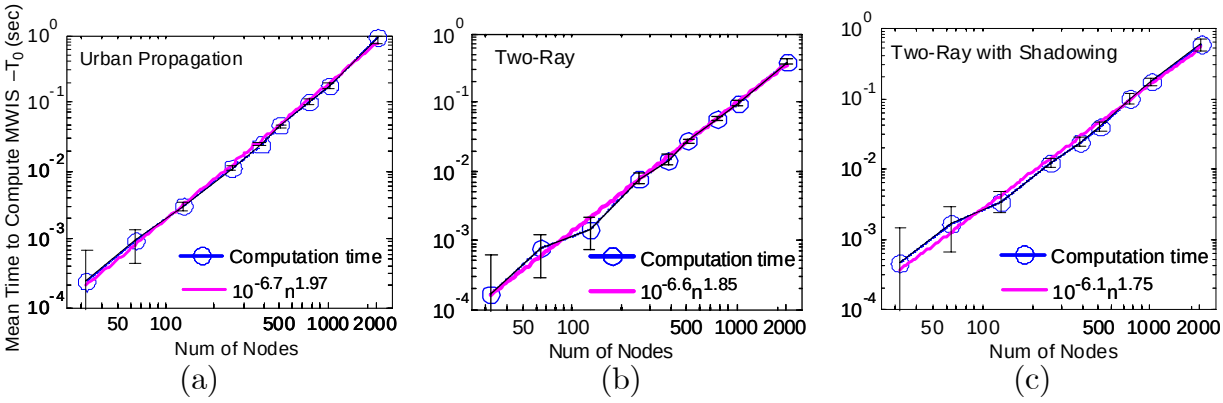


Figure 7.1: The time to compute the MWIS versus the number of nodes in the network. (a), (b), and (c) show this relationship when the propagation is the urban propagation, the two-ray model, and the two-ray with shadow fading model, respectively. In all cases, the target number of neighbors is 6, the target bit-rate is 24 Mbps, and the number of gateways is the number of nodes divided by 16.

different from the average complexity when averaged over problems that appear in practice. For this reason, this work investigates the practical complexity of the MWIS problems that arise in computing optimal schedules for wireless networks.

7.3 Computation Time as a Function of the Number of Nodes in the Network - The Low Degree Case

Figure 7.1 (a), (b), and (c) show the average time to solve (5.1) for urban propagation, the two-ray propagation model, and the two-ray with shadowing propagation model, respectively. These computation times[1] were averaged over each iteration of Algorithm 1 and averaged over 40 randomly generated topologies. For small topologies, the computation time is quite small. Randomness due to memory

[1] All computations were run on a machine with two Intel E5440 CPUs and 16GB RAM using Matlab v 7.2, and CPLEX v 10 with the Tomlab interface to CPLEX. However, at all times, 8 computations were solved simultaneously. Hence, the computation times shown correspond to the time on a single core (i.e., the MWIS problem was not parallelized across cores).

management and other high priority tasks, small computation times are significantly influenced by noise. Therefore, for topologies with fewer than 256 nodes, each time that the ILP program was to be solved, it was repeatedly solved ten times. The computation time was then the average of these ten times. Figure 7.1 also shows the 95% confidence intervals, which were found with bootstrapping. When generating these topologies, the target number of neighbors, Δ, was equal to 6, the target bit-rate was set to 24Mbps, and the number of gateways was the number of nodes in the network divided by 16.

Three conclusions can be made from Figure 7.1. First, the time to solve the MWIS is quite small, with 2048 node topologies taking approximately one second. Clearly, the statement that the MWIS can only be solved for trivial networks is incorrect. Second, it appears that in practice, the time to compute the MWIS grows polynomially with the size of the network. Specifically, for the topologies shown in Figure 7.1, we have

$$\begin{aligned} & \text{Time to find a MWIS for } \Delta = 6 \\ \approx \quad & A \times n^B + T_o \text{ sec.} \end{aligned} \tag{7.1}$$

where (A, B, T_o) is $(10^{-6.7}, 1.97, 0.0095)$, $(10^{-6.7}, 1.85, 0.0095)$, $(10^{-6.1}, 1.75, 0.0095)$ for the urban propagation, the two-ray model, and the two-ray with shadowing model, respectively. This relationship between computation time and topology size is also shown in Figure 7.1. However, as will be shown in the following sections, this behavior is unique to networks that have a low degree (i.e., Δ is small). A third conclusion drawn from Figure 7.1 is that the computational complexity does not greatly depend on the propagation model. Specifically, the computation times for different propagation models are within 10%.

Note that in (7.1), $T_o = 0.0095$ sec., for all types of propagation. We suspect that it takes approximately 9.5 msec to load the CPLEX optimizer (which is a DLL) and to begin solving the MWIS problem.

7.4 Impact the Mean Degree of the Conflict Graph

Figure 7.1 shows a relationship between the computation time and the size of the network under a specific type of topologies. We seek to understand this relationship for a wider range of types of topologies. This section will present results from computational experiments that indicate that

$$\frac{\text{Time to find a MWIS} - T_o}{\text{Mean degree of the conflict graph}} \approx \alpha \times n^{\beta}, \tag{7.2}$$

where α and β only depend on the propagation environment. This result implies that in terms of the time to find a MWIS, the mean degree of the conflict graph encapsulates many of the parameters used for generating topologies, namely, the number of gateways, the target bit-rate, r^*, and the target number of neighbors, Δ.

We take two steps to show that (7.2) is a good model for the computation time. First, we fix n, in which case (7.2) implies that

$$\begin{aligned} &\text{Time to find a MWIS} - T_o \\ \approx\ & K \times \text{Mean degree of the conflict graph,} \end{aligned} \tag{7.3}$$

where the constant K is dependent of the number of nodes, but K is independent of the other parameters of the topology, namely, the number of gateways, the target bit-rate, r^*, and the target number of neighbors, Δ. The next two subsections will explore the relationship between K and these other topology parameters.

In the second step to demonstrating (7.2), we will show that K is polynomial in n, that is, $K = \alpha \times n^{\beta}$ for some α and β, where α and β depend on the propagation model. This relationship is explored in Section 7.4.3.

7.4.1 The Mean Degree of the Conflict Graph, the Number of Gateways, and the Number of Neighbors

Figure 7.2 shows the relationship between the mean degree of the conflict graph and the computation time for different numbers of gateways and different

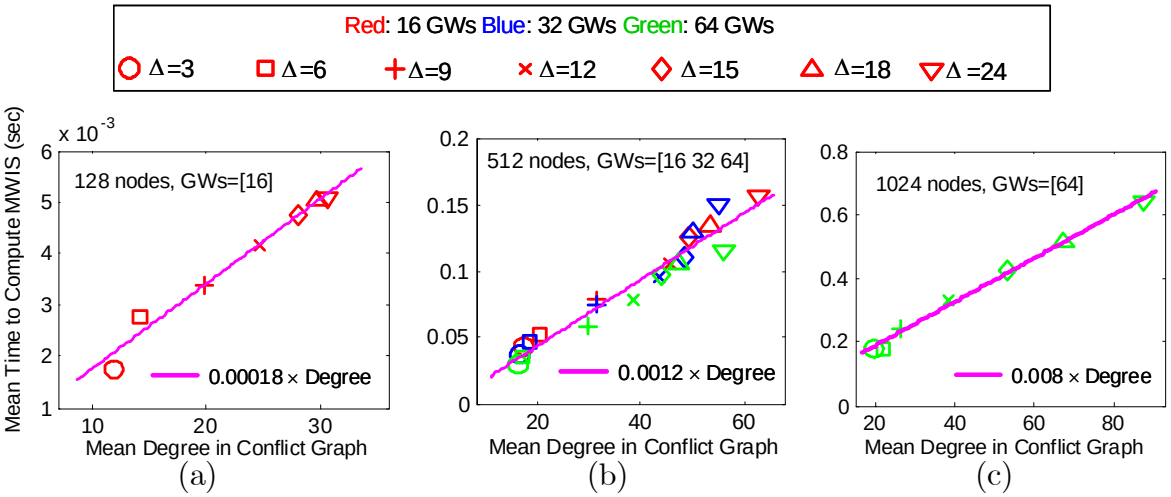


Figure 7.2: The mean time to compute the MWIS versus the mean degree of the conflict graph for several topologies. (a) Shows the case where the topologies have 128 nodes, 16 gateways, and the target number of neighbors, Δ vaires from 3 to 24. (b) Shows the case where the topologies have 512 nodes, 16, 32, and 64 gateways, and Δ vaires from 3 to 24. (c) Shows the case where the topologies have 1024 nodes, 64 gateways, and Δ vaires from 3 to 24.

numbers of target neighbors, Δ, but with the target bit-rate fixed at 24 Mbps. Here, urban propagation is used and, as above, each point is averaged over 40 topologies (both the mean degree of the conflict graph and mean computation time are averaged over 40 topologies). As can be observed, there is an approximately linear relationship between the computation time and the mean degree of the conflict graph. Specifically, the computation time is approximately given by (7.3), where K is 0.00018, 0.0012, and 0.008 for 128, 512, and 1024 node networks, respectively. These models are also shown in Figure 7.2.

Figure 7.2 also shows the computation time for particular values of the topology parameters. As expected, as Δ, the target number of neighbors in the wireless network, increases, the mean degree of the conflict graph increases. Figure 7.2 (b) shows the computation time for different numbers of gateways. As can be observed,

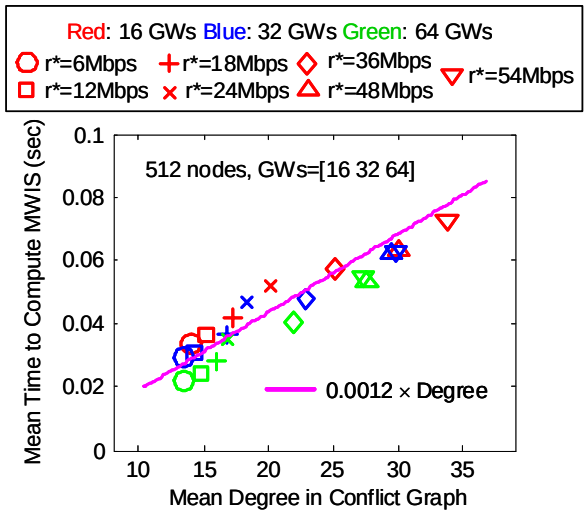


Figure 7.3: The mean time to compute the MWIS as a function of the mean degree of the conflict graph for different topologies where the topologies are generated with different target bit-rates.

as the number of gateways increases, the mean degree of the conflict graph slightly decreases, leading to a slight reduction in the computation time. Moreover, notice that the computation times for a larger number of gateways tends to be slightly below the linear fit, while a smaller number of gateways cases tend to be slightly above the linear fit. Nonetheless, the linear relationship between the mean degree of the conflict graph and the computation time provides a reasonable approximation for a wide range of gateways.

7.4.2 The Mean Degree of the Conflict Graph, the Target Bit-Rate, and the Number of Neighbors

The topologies shown in Figure 7.2 used a target bit-rate of 24Mbps. Figure 7.3 shows the relationship between the mean degree of the conflict graph and the time to compute the MWIS for a wide range of target bit-rates and different numbers of gateways. In this case, where urban propagation was used, there were 512 nodes

in the topology, and the target number of neighbors was set to 6. This figure shows that the mean degree (and computation time) increases with the target bit-rate. The reason for that the mean degree of the conflict graph increases with the target bit-rate is that higher bit-rates are more susceptible to interference. For example, 54 Mbps requires 23 dB of SINR, while 6 Mbps only requires 5 dB of SINR. Consequently, transmissions that are several hops away will interfere with high bit-rate transmissions, whereas only nearby transmissions will impact low bit-rate transmissions. Since the target number of neighbors is held fixed, links in topologies where a high target bit-rate is used will interfere with considerably more links than do links in topologies with low target bit-rates.

The linear fit shown in Figure 7.3 is the same one shown in Figure 7.2, i.e., (7.3) with $K = 0.0012$. This further confirms the linear relationship between the mean degree of the conflict graph and the time to compute the MWIS.

7.4.3 Time to Compute a MWIS and the Mean Degree of the Conflict Graph

The previous sections provide a strong indication that (7.3) is a good model for the computation time, where K only depends on the number of nodes. While Figures 7.2 and 7.3 only show the case for urban propagation. However, the plots are similar for the two-ray propagation model and the two-ray with lognormal shadowing propagation model.

Figure 7.4 (a), (b), and (c) show K as a function of the number of nodes for the urban propagation model, the two-ray propagation model, and the two-ray with lognormal shadowing propagation model, respectively. These plots also show the model $K = \alpha n^{\beta}$, where $(\alpha, \beta) = (1.77 \times 10^{-8}, 1.88), (1.09 \times 10^{-7}, 1.64)$, $(7.78 \times 10^{-8}, 1.75)$, for the three types of propagation, respectively. As can be observed, this model provides a high quality of fit. Thus, we conclude that with the computers and algorithms used in this investigation, the time to solve the MWIS

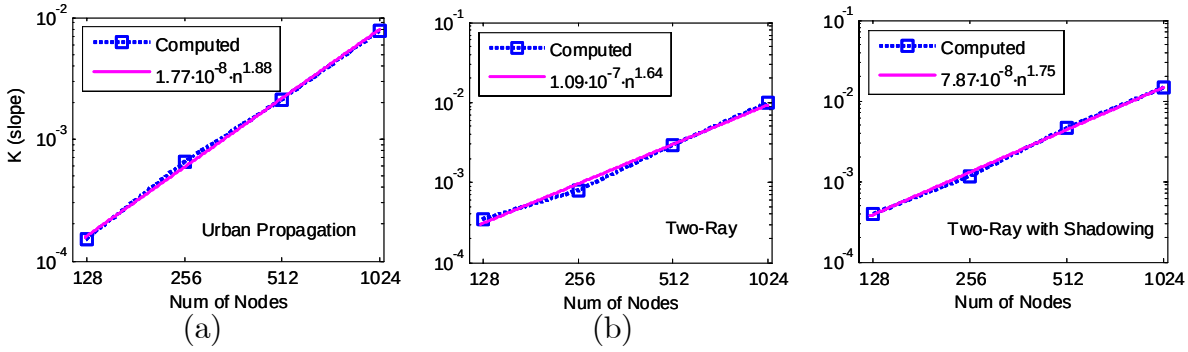


Figure 7.4: The relationship between K, the parameter in (7.3), and the number of nodes in the network. (a) is for urban propagation (b) is for the two-ray propagation model, and (3) is for the two-ray propagation model with lognormal shadow fading.

problem associated with optimal scheduling in practical wireless networks can be modeled with (7.2).

7.5 The Mean Degree of the Conflict Graph

It is important to note that (7.2) does not imply that the time to find a MWIS grows like n^β. Specifically, as Figures 7.2 and 7.3 show, the mean degree of the conflict graph also varies with the number of nodes. Thus, the scaling of the computation time depends on how the network is scaled, or more specifically, it depends on how this scaling impacts the mean degree of the conflict graph. In the case of the urban propagation, the time to compute the MWIS grows like n^β with $\beta = 1.88$ only if the mean degree of the conflict graph is somehow held constant as the size of the network grows. However, the mean degree of the conflict graph varies in a complicated way, and hence there does not appear to be any simple relationships between the number of nodes and the mean degree of the conflict graph.

Figure 7.5 shows how, in the case of the urban propagation, the mean degree of the conflict graph varies as the number of nodes increases, but the gateway density is held constant and Δ, the target number of neighbors, is held constant. For this

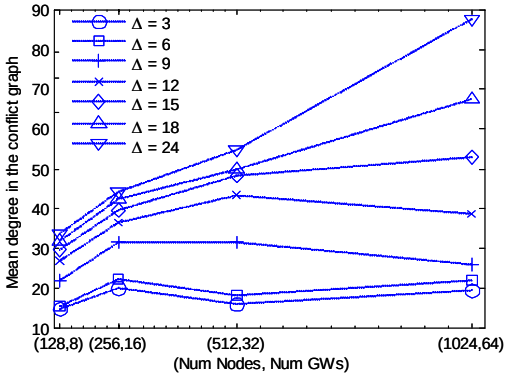


Figure 7.5: Relationships between the number of nodes and the mean degree of the conflict graphs.

type of scaling of the topology, it is difficult to draw any definitive conclusions about the relation between the number of nodes and the mean degree of the conflict graph. For example, for $\Delta = 24$ and $\Delta = 18$, the mean degree clearly increases with the number of nodes. Thus, the time to find a MWIS increases faster than $n^{1.88}$. However, for other values of Δ, the mean degree of the conflict graph appears to reach a plateau.

Note that the models shown in Figure 7.1 do not have the same exponent as the one given in the previous section. These can be reconciled by recognizing that the mean degree of the conflict graph can vary with n. For example, in the urban propagation case shown in Figure 7.1, the computation time grows like $n^{1.97}$, thus in order for (7.2) to hold with the parameters given above, we must have that for the topologies used to generate Figure 7.1 (a)

$$\frac{A \times n^{1.97}}{\text{mean degree of the conflict graph}} = 1.77 \times 10^{-8} n^{1.88},$$

which implies that the mean degree of the conflict graph must grows like $n^{0.09}$. Similarly, for the two-ray model and the two-ray with shadowing model, the models shown in Figure 7.1 (b) and (c) can be reconciled with (7.2) if the mean degree of

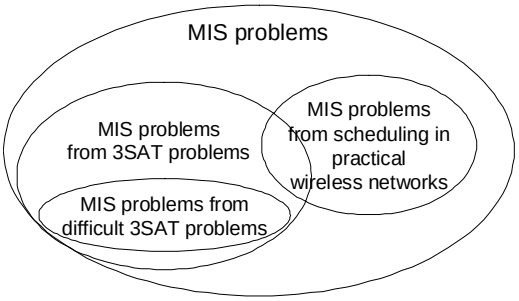


Figure 7.6: Possible Venn diagram of maximum independent set (MIS) problems, 3-SAT problems, and scheduling problems.

the conflict graph is approximately $O\left(n^{-0.21}\right)$ and $O\left(n^{0.0}\right)$, respectively. Considering Figure 7.5, such a variation is plausible, especially, when Δ is small, as it is in the case of Figure 7.1.

7.6 Conclusions

This work studied the practical computational complexity of the maximum weighted independent set (MWIS) problem that arises in optimal scheduling in wireless networks. In contrast to folklore, the MWIS problem is solvable in many practical wireless scheduling problems. Specifically, by examining over 10000 randomly generated topologies, it was found that the time to compute the MWIS grows polynomially with the number of nodes and linearly with the mean degree of the conflict graph. Moreover, the mean time to solve the MWIS problem for networks with 2048 nodes was approximately one second.

While this result might appear to be in conflict with prior research on the complexity of scheduling, it is not. First, there are a wide range of problems that have a worst-case complexity that grows exponentially with the size of the problem, and yet in practice grow polynomially with the size of the problem. Second, prior research on the complexity of scheduling relied on the relationship between the 3-SAT problem and the MWIS problem. However, it is well known that in many cases

the 3-SAT problem can be quickly solved [103, 104, 105, 106]. Moreover, the MWIS problem that arises in practical scheduling is particular subset of MWIS problems. The relationship between practical scheduling and the 3-SAT problem is not clear. Figure 7.6 show a possible Venn diagram of the set of problems. Note that it is unknown if there is any overlap between difficult 3-SAT problems and the MWIS problems that arise in practical wireless networks. The computational experiments in this work indicate that there is not a significant overlap.

An important consequence of this work is that the ability to quickly solve MWIS problems allows optimal schedules to be quickly found. In previous work, the perceived practical intractability had been circumvented by using suboptimal methods or by making strong assumptions about interference.

Chapter 8

OPTIMAL ROUTING

8.1 Introduction

Researchers have been investigating scheduling techniques for at least 25 years [30], and it was shown that in the worst case determining the optimal schedule is NP-hard [108]. Consequently, joint optimal scheduling and optimal routing has appeared to be impractical. However, we propose an algorithm that can efficiently compute optimal schedules even while accommodating co-channel interference.

With the ability to quickly compute optimal schedules, optimal routing (with optimal scheduling) can be investigated. In the worst-case, the number of routes between a source and destination increases exponentially with the size of the network. Thus, it is not tractable to directly optimize over all routes. Instead, an iterative scheme is developed where paths are added only if they are guaranteed to improve the throughput. However, guaranteed optimal routing remains computationally complex. Thus, an approximate algorithm is developed. The performance of this algorithm is compared to the optimal algorithm for several topologies and it is found to result in the same throughput as the optimal routing.

We also explore several aspects of the behavior of the proposed routing in realistic mesh networks. For example, it is found that when compared to a least hop routing algorithm, the approximate algorithm increases throughput by as much as a factor of two. However, the typical improvement is approximately 60%. Considering that scheduling (with least hop routing) improves throughput over 802.11's MAC

by a factor between 4 and 10 (depending on the density of gateways), we conclude that optimal routing and scheduling typically provide an improvement of a factor between 6 and 15 over a 802.11-based deployment.

As a comparison, we consider a max-flow based routing algorithm that is similar to the one developed in [40] (except optimal scheduling is used, whereas in [40], a greedy scheduling algorithm is used). The approximate routing algorithm typically yields between 20% and 35% improvement in throughput over this max-flow based algorithm. Other aspects of routing explored include the path length and the performance of a single path quantization of optimal multipath routing, and the performance when subjected to time varying offered load. It is important to note that the topologies considered in this work were developed with the UDelModels urban wireless network simulator, which realistically models wireless propagation in urban areas [90].

The remainder of the chapter proceeds as follows. In the next section some notation and the problem definition are given. Section 8.3 develops an algorithm for optimal routing while Section 8.4 develops an approximation of optimal routing. Then, Section 8.5 and 8.6 explore the behavior of these routing algorithms in realistic mesh networks. Finally, concluding remarks are provided in 8.7. We must notice that Section 6.1.2.1 develops a max-flow based routing algorithm, which is clearly suboptimal, but computationally efficient.

8.2 Notation and Problem Definition

A router-to-router connection is denoted by ϕ, with Φ denoting the set of all connections. A connection may make use to several flows with each flow using a different path; the kth path for connection ϕ is denoted by (ϕ, k). The data rate of flow (ϕ, k) is denoted by $f_{\phi,k}$, and the path followed by flow (ϕ, k) is denoted by $P(\phi, k)$. The set of paths used by connection ϕ is denoted with $P(\phi)$, i.e., $P(\phi) = \{P(\phi, k) : k = 1, ..., |P(\phi)|\}$, where $|P(\phi)|$ is the number of paths used by

connection ϕ. The set of all considered paths is $\mathcal{P}$. Using this notation, the total data rate over connection ϕ is $\sum_{k=1}^{|P(\phi)|} f_{\phi,k}$, and the total data rate sent over link x is $\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k}$, where $\{(\phi,k)\,|x \in P(\phi,k)\}$ is the set of flows that cross link x.

A schedule is a convex combination of assignments. Specifically, a schedule is a set $\{\alpha_{\mathbf{v}} : \mathbf{v} \in \mathcal{V}\}$ where $\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1$ and $\alpha_{\mathbf{v}} \geq 0$. With this notation, the total data rate that the schedule $\{\alpha_{\mathbf{v}} : \mathbf{v} \in \mathcal{V}\}$ provides over link x is $\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R_x v_x$. Then, the throughput optimization problem is

$$\max G\left(\vec{f}\right) \tag{8.1}$$

subject to:

$$\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k} \leq \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R\left(\mathbf{v}, x\right) \text{ for each link } x \tag{8.2}$$

$$\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1 \tag{8.3}$$

$$0 \leq \alpha_{\mathbf{v}} \text{ for each } \mathbf{v} \in \mathcal{V}, \tag{8.4}$$

where $\vec{f}$ is the vector of flow rates. Several different functions G are possible. Popular ones include $G\left(\vec{f}\right) = \sum_{\phi\in\Phi} w_\phi \log\left(\sum_k f_{\phi,k}\right)$ and $G\left(\vec{f}\right) = \min_{\phi\in\Phi} w_\phi \sum_k f_{\phi,k}$. In both cases, w_ϕ is the administrative weight for router-to-router connection ϕ.

8.3 Optimal Routing

A naive approach to optimal routing is to let each connection ϕ use all possible routes. That is, for each ϕ, $P\left(\phi\right) = \{P\left(\phi,k\right) : k = 1, 2, ...\}$ is the set of all possible paths between the source and the destination of connection ϕ. With this set of paths, the optimal schedule can be found. The schedule will determine the portion of the flow $\sum_{k=1}^{|P(\phi)|} f_{\phi,k}$ that uses path $P\left(\phi,k\right)$. In this way, optimal scheduling also performs optimal routing. While such an approach is possible in theory, in practice, it is not feasible since the number of all possible paths between a source and destination grows exponentially with the size of the network.

As an alternative, we consider adding paths only when they will improve the performance. Specifically, suppose that connection ϕ uses the set of paths $P(\phi)$. Now consider the candidate path $P(\phi, k^+) \notin P(\phi)$. This path should be added to the set of considered paths only if the addition of this path improves the throughput. The question of whether including the path $P(\phi, k^+)$ into the set of considered paths will increase the network throughput is answered by the following.

Proposition 23 *Let $P(\phi, k^+) \notin P(\phi)$ be the path of a candidate flow potentially to be added for connection ϕ. Furthermore, let $P(\phi, k^o) \in P(\phi)$ be such that $f_{\phi,k^o} > 0$. Then, adding the candidate path $P(\phi, k^+)$ to $P(\phi)$ will increase the throughput if and only if*

$$\sum_{x \in P(\phi,k^+)} \mu_x < \sum_{x \in P(\phi,k^o)} \mu_x, \tag{8.5}$$

where μ_x is the Lagrange multiplier associated with constraint (8.2) that arises from solving Problem (8.1) without the inclusion of the path $P(\phi, k^+)$.

The proof of Proposition 23 is addressed in Section 8.8.1. Employing the economic interpretation of Lagrange multipliers, Proposition 23 says that path $P(\phi, k^+)$ should be included into the set of considered paths if its end-to-end cost $\sum_{x \in P(\phi,k^+)} \mu_x$ is less than the end-to-end cost of the currently used flows.

Following the approach of the proof of Proposition 23, it is straightforward to prove that data is only sent down paths where the cost is minimum.

Proposition 24 *If $\sum_{x \in P(\phi,k_1)} \mu_x < \sum_{x \in P(\phi,k_2)} \mu_x$, then $f_{\phi,k_2} = 0$.*

Corollary 25 *If $f_{\phi,k_1} > 0$ and $f_{\phi,k_2} > 0$, then $\sum_{x \in P(\phi,k_1)} \mu_x = \sum_{x \in P(\phi,k_2)} \mu_x$.*

Since Problem (8.1) is convex, it is straightforward to show that if there does not exist a path that improves the throughput, then the current routing is optimal.

Algorithm 8 Optimal Routing

1: Set $m = 0$, then select an initial set of paths $\mathcal{P}(0)$ and a empty new path $P(\phi, k^+)$.
2: **repeat**
3: Add this path $P(\phi, k^+)$ to the set of considered paths for connection ϕ, and set $m = m + 1$.
4: Remove any unecessary paths from consideration.
5: Compute the optimal schedule for the set of Considered paths $\mathcal{P}(m)$ and get the Lagrange multiplers μ_x associated with Constraint (8.2).
6: Search for a path $P(\phi, k^+) \notin \mathcal{P}(m)$ that satisfies (8.5).
7: **until**
8: No such path exists.

Thus, Algorithm 8 can be used to incrementally add paths until no further improvement is possible, that which point the optimal throughput has been determined. Also, Figure 8.1 shows the flow chart for Algorithm 8.

There are several challenges in applying Algorithm 8. A critical one is related to determining μ_x for all links. This issue is addressed in the next section. Other important issues are related to removing unnecessary paths as discussed in Step 4. One approach is to remove paths that carry no data. That is, if the optimal schedule dictates that $f_{\phi,k} = 0$, then the path $P(\phi, k)$ can be removed from consideration without impacting the throughput. However, as discussed in the next sections, in some cases, the removal of paths is more complicated.

In [109], the idea of updating routing based on the sum of the end-to-end Lagrange multipliers was investigated for wired networks. However, the focus there was on single path routing and on the conditions under which single path routing is optimal.

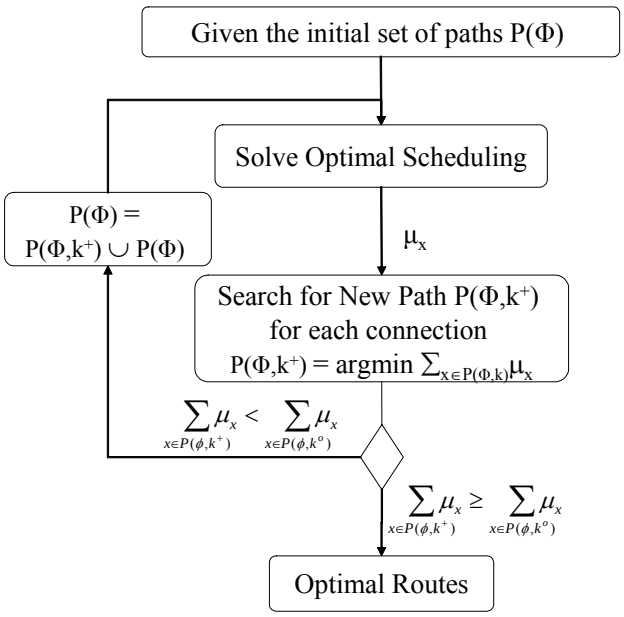


Figure 8.1: Algorithm for optimal routing

8.4 Evaluating the Path Cost

8.4.1 Approaches to Evaluating the Path Cost

One significant drawback of Algorithm 8 is that it requires μ_x to be known for each link x. However, the scheduling algorithm will only compute μ_x if the set of all considered paths $\mathcal{P}$ is such that for each x there exists a path $P \in \mathcal{P}$ such that $x \in P$. There are three ways to address this drawback. First, Algorithm 8 can be applied but neglecting links that are not used by any path in $\mathcal{P}$. The result will be suboptimal. Specifically, only links considered by the initial routing will be used when the algorithm terminates. Based on realistic simulations (see Section 8.5 for a discussion of the simulation set-up), we have found that this approach performs poorly, especially when the network is dense (i.e., when there are many links that have high channel gain and are not included into the routing).

A second approach to accommodating Algorithm 8's need for μ_x for all links is to ensure that the routing at each iteration is such that there is at least one route crossing each link. This brute force approach suffers from high computational complexity since there may be a large number of links. Since some links have very low channel gain, and hence, it can be assumed that such links would not be utilized in optimal routing. Thus, the number of unidirectional links is considerably smaller than $N(N-1)$, which is the maximum number of directional links. Nonetheless, in dense mesh networks, there are a very large number of links, limiting the applicability of the brute force method. On the other hand, this approach is guaranteed to yield the optimal routing.

A third approach is to find an upper bound, $\bar{\mu}_x$ such that $\mu_x \leq \bar{\mu}_x$ for the links that are not included in any path and $\bar{\mu}_x = \mu_x$ for links that are included in at least one path. If such a bound is known (one is given in the next section), then from Proposition 23 we have the following.

Corollary 26 (Sufficient Condition to Include a Path) *Let $\mu_x \leq \bar{\mu}_x$. Let $P(\phi, k^+) \notin P(\phi)$ be the path of a candidate flow potentially to be added to connection ϕ. Furthermore, let $P(\phi, k^o) \in P(\phi)$ be such that $f_{\phi,k^o} > 0$. Then, adding the candidate path $P(\phi, k^+)$ to $P(\phi)$ will increase the throughput if*

$$\sum_{x \in P(\phi,k^+)} \bar{\mu}_x < \sum_{x \in P(\phi,k^o)} \mu_x. \tag{8.6}$$

Note that since the path $P(\phi, k^o)$ is already included in $\mathcal{P}$, μ_x is known for all $x \in P(\phi, k^o)$, and hence, the right-hand side of (8.6) can be computed. Of course, the drawback of using a bound $\bar{\mu}_x$ is that some paths $P(\phi, k^+)$ might satisfy (8.5), but not (8.6). Such paths would not be included into $\mathcal{P}$. Therefore, replacing condition (8.5) with condition (8.6) in Algorithm 8 results in suboptimal routing. On the other hand, at each iteration of Algorithm 8, more paths are added to $\mathcal{P}$.

Algorithm 9 Heuristic for Removing Paths from Consideration

The path $P(\phi, k)$ is not removed from the set of considered paths if any of the following hold.

0: The path $P(\phi, k)$ was added to the set of considered paths within the past M iterations.

1: $f_{\phi,k} \neq 0$ during at least one of the past M iterations.

2: There exists a link $y \in P(\phi, k)$ such that $y \notin P(\varphi, j)$ for any other flow (φ, j) and μ_y has been used in computing the upper bound $\bar{\mu}_z$ for some link z at least once during the past M iterations.

Algorithm 10 (Sub)Optimal Routing

Use Algorithm 8 with Condition 8.5 replaced with 8.6, where $\bar{\mu}_x$ is computed with 8.7, and Algorithm 9 in Step 4 of Algorithm 8.

Thus, if a path is incorrectly not added due to the inaccuracy of $\bar{\mu}_x$, it might be added at later iterations once μ_x is determined for more links.

The performance of Algorithm 8 with condition (8.6) can be improved by adjusting when paths are removed in Step 4. On the one hand, if more links that are included in the $\mathcal{P}$, then μ_x is known for more links x, and hence the decision of which paths to include can be made more accurately. On the other hand, the complexity of the scheduling increases with the number of links included into the routing. Thus, Algorithm 9 is developed to determine when paths should be removed from considerations. Roughly speaking, Algorithm 9 removes paths after they have not been used by the routing algorithm in any way for more than M iterations. As demonstrated in Section 8.5, $M = 5$ works well. In summary, as an alternative to ensuring that each link is used by some route, Algorithm 10 is used.

8.4.2 An Upper Bound on μ_x

Let $\mathcal{L}$ be the set of links used in any path, i.e., $\mathcal{L} = \{x : \text{there exists a } P \in \mathcal{P} \text{ such that } x \in P\}$. Thus, for each $x \in \mathcal{L}$, μ_x is determined by the scheduling. Denote by $\mathcal{CG}[\mathcal{L}]$ to be the conflict graph induced by the links $\mathcal{L}$. Furthermore, the sub-graph of the conflict graph induced by removing link x is denoted with $\mathcal{CG}[\mathcal{L} \backslash x]$. R_x

is the data rate over link x when there are no links $y \in \chi(x)$ are transmitting. Let $MWIS(\mathcal{CG}[\mathcal{L}])$ denote the links in the maximum weighted independent set. Similarly, let $MWIS(\mathcal{CG}[\mathcal{L}\backslash(\{z\} \cup \chi(z))])$ be the set of links in a maximum weighted independent set with links $\mathcal{L}$ but excluding link z and all links in conflict with z.

Employing this notation, we have the following.

Proposition 27 *Let $z \notin \mathcal{L}$, then $\bar{\mu}_z \geq \mu_z$ where*

$$\bar{\mu}_z = \left(\lambda - \sum_{x \in MWIS(\mathcal{CG}[\mathcal{L}\backslash(\{z\} \cup \chi(z))])} R_x \mu_x \right) / R_z. \tag{8.7}$$

The proof of Proposition 27 is in Section 8.8.2. The economic interpretation of Lagrange multipliers can be used to intuitively explain (8.7). Specially, μ_x is the price to transmit a bit across link x. Let $\mathbf{v} \in \{0,1\}^L$ be an assignment without conflicts (i.e., $v_x = 1$ implies $v_y = 0$ for all $y \in \chi(x)$). Then the revenue per second generated by this assignment is $\sum_x R_x \mu_x v_x$. It can be shown that the best assignments are those that achieve $\lambda = \sum_x R_x \mu_x v_x$ [110]. Thus, the network is "willing" to multiplex between any assignments as long as its revenue per second is λ. Thus, the network is willing to allow transmission over z if the price per bit to transmit across link z is (8.7).

While Proposition (27) yields a useful bound on μ, it requires the computation of a maximum weighted independent set. While we have found that in realistic networks, MWIS problem can be computed quickly.

8.5 Computational Experiments

In this section the throughputs provided by different routing algorithms are compared. In all cases, optimal scheduling is used. Hence, only the routing is changed. In the following computational experiments, the throughput metric is $G\left(\vec{f}\right) = \min_{\phi_{\in\Phi}} w_\phi \sum_{k=1}^{|P(\phi)|} f_{\phi,k}$. However, the results for $G\left(\vec{f}\right) = \sum_{\phi\in\Phi} w_\phi \log(\sum_{k=1}^{|P(\phi)|} f_{\phi,k})$ have been examined and yield qualitatively similar results.

8.5.1 Simulated Urban Mesh Networks

The topologies generated in Section 6.1.1 are deployed in this investigation. The 6×6 city block regions were randomly located in a 2 km^2 region. Various nodes densities were investigated. Specifically, the number of gateways was 1, 2, 3, 6 and the number of wireless routers was 18, 36, 54, 72, and 90. The wireless routers and gateways were uniformly distributed throughout the 6×6 block region. Ten samples of each topology were generated (hence, 200 topologies in total).

In these experiments, all traffic flow from gateways to destinations (i.e., downstream traffic), where each mesh router in the topology was a destination of a flow. The baseline routing uses single path of least hop where each link had a receiver signal strength of at least 55 dBm. Among paths with the same number of hops, the path selected was the one that had the highest minimum link channel gain, where the minimization is over each hop along the path. Each flow originates at the gateway such that the best route from the gateway to the destination of the flow is no worse than any route from any other gateway in terms of the minimum channel gain along the route.

8.5.2 Comparison of Algorithms

As discussed in Section 8.4, Algorithm 8 yields optimal routing if the Lagrange multiplier for each possible link in the network is always included into some path. Typically, there are many possible links in the network, hence, this approach is computational complex. Algorithm 10 is less computational complex, but is suboptimal. On the other hand, since Algorithm 9 is conservative in removing a path from the set of considered paths, Algorithm 10 tends to include a fairly large number of links, and hence has the potential to accurately determine which paths to include.

Figure 8.2 compares the throughput found by these algorithms for topologies with 18 and 36 mesh routers and with 1 and 3 gateways. For each number of routers and gateways, ten samples were considered (hence, 40 topologies in total

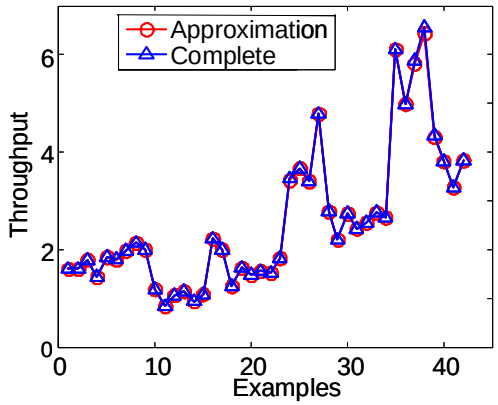


Figure 8.2: Throughput found from optimal and suboptimal routing for various urban topologies.

were considered). As can be observed, Algorithm 10 yields the same throughput as the optimal algorithm. We have found that the Algorithm 9 is critical to the performance of Algorithm 10. On the other hand, Algorithm 10 with Algorithm 9 tends to include a large number of links. Nonetheless, it is considerably more efficient than including all possible links, and hence optimal routing with all possible links is not considered in the remainder of this work. Nonetheless, since we have not verified that Algorithm 10 is indeed optimal for all topologies, the throughput found by Algorithm 10 is referred to as (sub)optimal.

8.5.3 The Impact of Optimal Routing

The left-hand side of Figure 8.3 shows the improvement in the throughput of (sub)optimal routing as compared to the initial least hop routing. As can be observed, optimal routing can significantly improve the throughput. However, we note that this improvement is far less than the improvement of optimal scheduling over 802.11 (which is between a factor of four and ten on the same topologies). Nonetheless, a 20 - 75% improvement in throughput is significant.

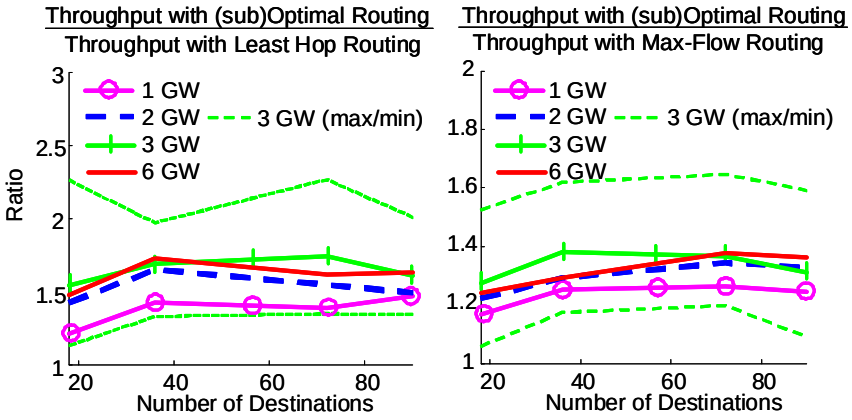


Figure 8.3: The median of the ratio of the throughput with (sub)optimal routing and the throughput with the least hop routing (left) and max-flow routing (right). Also shown is the maximum and minimum value of the ratio when there are three gateways. The maximum and minimum for other numbers of gateways are similar.

The right-hand side of Figure 8.3 shows the throughput improvement of (sub)optimal throughput and the throughput that results from using max-flow based routing described in Section 6.1.2.1. Comparing the left and right-hand sides of Figure 8.3, it can be observed that max-flow routing provides significantly more throughput than the least hop routing. However, (sub)optimal routing typically provides 20 - 35% higher throughput than max-flow based routing. On the other hand, max-flow based routing (with even optimal scheduling via Algorithm 1) is quite efficient and can be computed in tens of seconds for the moderate size of topologies considered here.

As a brief aside, consider Figure 8.4, which we compare the optimal scheduling of Algorithm 1 and the greedy scheduling presented in [40] along with max-flow based routing. It can be observed that in the topologies considered, optimal scheduling increases the throughput by a factor between 50% and 2.5.

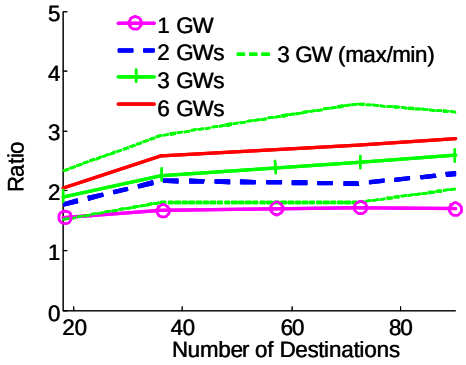


Figure 8.4: Median ratio of the throughput with max-flow routing and optimal scheduling and the throughput with max-flow routing and greedy scheduling from [40]. The maximum and minimum ratio is shown in the case of three gateways.

8.5.4 Path Length

It is often claimed that high performance can be achieved by taking many short hops where, since each hop is short, and hence has utilized a good channel, each hop can support high bandwidth. While a single short hop can proceed at a high data rate, if a connection requires many short hops, then it will require many transmissions, which will increase self-interference and interference with other connections. In this way, perhaps shorter paths and longer hops might be preferable [111].

Figure 8.5 shows the average number of hops for different routing schemes. In general, least hop routing uses the shortest paths (as expected), and (sub)optimal routing uses the longest paths. In the case of one gateway in a 6×6 city block region, (sub)optimal routing uses significantly longer paths than least hop paths. As the density of gateways increases, all path lengths decrease (since routers are closer to some gateway). In order to better compare the path lengths at high gateway densities, Figure 8.6 shows the mean ratio of the path lengths. Observe that the path difference in path lengths decreases as the gateway density increases. However, (sub)optimal routing still uses considerably longer paths than least hop routing.

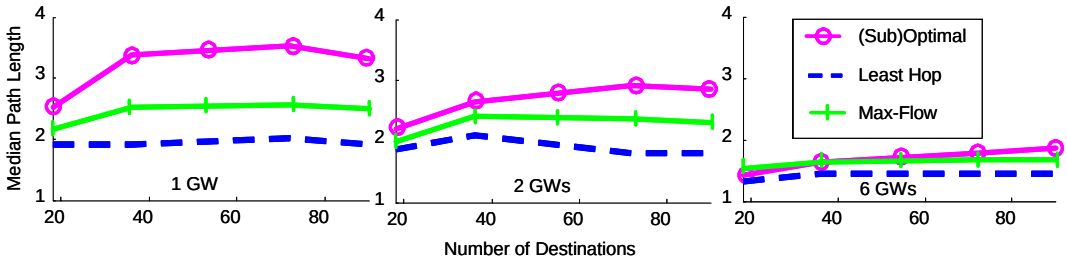


Figure 8.5: Median path length for different topologies and different routing algorithms.

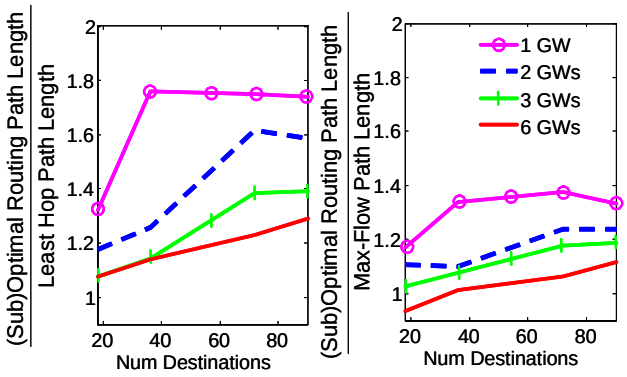


Figure 8.6: Left: Median of the ratio of the path lengths from least hop routing and path lengths from (sub)optimal routing. Right: Median of the ratio of the path lengths from of max-flow based routing and path length from (sub)optimal routing.

Note that since multipath routing is used and since the scheduling may assign small data flow to some paths, the path lengths used in Figure 8.5 and 8.6 are weighted by the amount of data flow that crosses the path. Specifically, the path length for connection ϕ is $\frac{1}{\sum_{k=1}^{|P(\phi)|} f_{\phi.k}} \sum_{k=1}^{|P(\phi)|} f_{\phi,k} \left(|P\left(\phi,k\right)|-1\right)$, where $|P\left(\phi,k\right)|-1$ is the path length of path $P\left(\phi,k\right)$.

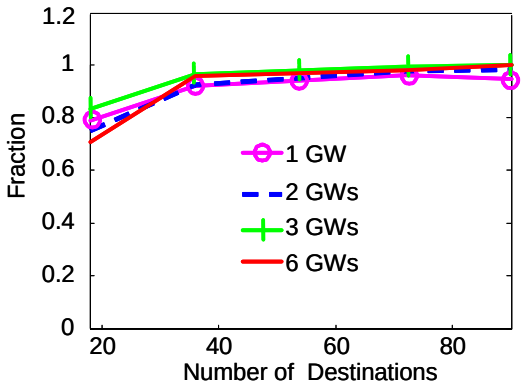


Figure 8.7: Fraction of connections that use multiple paths.

8.5.5 Multipath versus Single Path Routing

There has been extensive work on multipath routing (e.g., [112] and [113]). One motivation for using multipath routing is that multiple paths can increase robustness to disconnection and improve load balancing. It is expected that load balancing can increase throughput. While there is some evidence of this behavior in single connection ad hoc routing [114], the behavior in mesh networks at full throughput is unclear. In theory, optimal multipath routing is more general than optimal single path routing, and hence multipath routing cannot provide any lower throughput than single path. However, it is unknown the degree to which multipath routing is required for maximum throughput. Figure 8.7 shows the fraction of connections that use multiple paths. Note that Algorithm 10 might generate multiple paths, however, the scheduling algorithm might not allocate data flow to some paths. Thus, we say that a connection ϕ uses multiple paths if there exist k and j with $k \neq j$ and $f_{\phi,k} > 0$ and $f_{\phi,j} > 0$. As can be observed in Figure 8.7, a large majority of the connections require multiple paths. Note that a significant number of the paths were only a single hop. Thus, Algorithm 10 chooses to even augment paths that are one hop from the gateway with multi-hop paths.

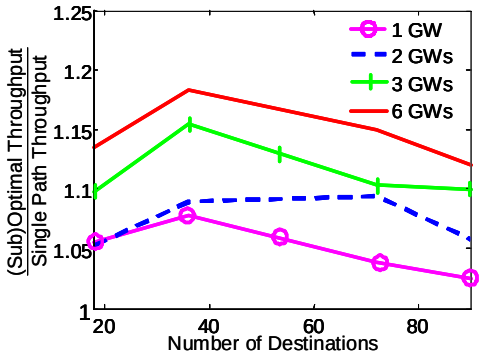


Figure 8.8: A comparison of the (sub)optimal throughput and the throughput provided by a single path routing that is a quantized version of (sub)optimal routing.

While Figure 8.7 indicates that optimality makes significant use of multiple paths, it does not indicate the impact of allowing multipath routing has on routing. Since optimal single path routing is a non-convex optimization [109], a precise comparison between optimal multipath routing and optimal single path routing is computationally complex. Nonetheless, an idea of the performance of single path routing can be gleaned by considering a single path routing that is a "quantized" version of multipath routing as follows. As mentioned above, even when there are multiple paths between a single source-destination pair, the scheduling algorithm need not allocate data flow to all available paths. For the paths that do have non-zero data flow, each path does not necessarily have the same data flow. Thus, for each connection ϕ, we can select a single path $P(\phi, k)$ such that $f_{\phi,k} \geq f_{\phi,j}$ for all j. If there are multiple paths with the same maximal data rate, then one is selected arbitrarily. Once the paths have been selected, the scheduling algorithm is used to compute the optimal throughput under the single path routing. Figure 8.8 shows the ratio of the throughput given by this single path routing and the (sub)optimal routing. As can be observed, the (sub)optimal routing provides no more than 20% higher throughput and generally less than 15% higher throughput than the single path routing. This is considerably less than the improvement of (sub)optimal

routing over least hop routing and max-flow routing (See Figure 8.3).

Note that when multipath routing is used, the address of the destination does not specify the route (or next hop). Hence, a labeling scheme or full source routing must be used. Moreover, if a single TCP flow is split over multiple paths, packet reordering is likely to occur. If packets are reordered by more than three, then TCP will assume a packet has been dropped and will decrease the congestion window and sending rate. While solutions to this problem exist [115], they are not widely deployed. Thus, when including the reduction in throughput due to the overhead to support, the difficulty in that multipath routing has on TCP, and the minimal increase in throughput provided by multipath routing, there appears to be little motivation to deploy multipath routing. However, in the next section we will see that multipath routing does provide significant advantages of single path routing algorithm described above.

8.6 Dynamic Routing and Scheduling

As discussed in Section 8.2, popular throughput metrics include $G\left(\vec{f}\right) = \sum_{\phi\in\Phi} w_\phi \log\left(\sum_k f_{\phi,k}\right)$ and $G\left(\vec{f}\right) = \min_{\phi\in\Phi} w_\phi \sum_k f_{\phi,k}$. Both approaches include administrative weights, $\{w_\phi : \phi \in \Phi\}$. One simple approach to setting the weights is to set the weights statically and for each *end-host to end-host* (e2e) connection that uses router-to-router connection ϕ to equally share the throughput allocated to ϕ. Consider the case where e2e connections start at Poisson distributed times and the mean connection size is β. Let

$$\begin{aligned} C\left(\vec{w}\right) &= \max \min_{\phi\in\Phi} w_\phi \sum_{k=1}^{|P(\phi)|} f_{\phi,k} \\ &\quad \text{subject to: (8.2)-(8.4).} \end{aligned}$$

Then, allocating each connection ϕ throughput $C\left(\vec{1}\right)$, where $\vec{1}$ is the vector of all ones, supports the maximum e2e connection arrival rate with the finite expected

number of active e2e connections. More generally, if the average e2e connection size over ϕ is β_ϕ, then setting $w_\phi = 1/\beta_\phi$ allows for the fastest arrival rate of e2e connections.

As compared to statically set weights, dynamically setting weights may improve the average time to complete an e2e connection (but, of course, will not change the maximum arrival rate such that the expected number of e2e connections is finite). One approach is to set $w_\phi(t) = 1/N_\phi(t)$ where $N_\phi(t)$ is the number of active e2e connections that are using connection ϕ at time t. In this case, each e2e connection is allocated throughput $C(w(t))$.

It is also possible to adjust the routing dynamically. As above, the weights could be set according to $w_\phi(t) = 1/N_\phi(t)$ and Algorithm 10 is applied to compute routing and scheduling. However, considering the computational complexity of Algorithm 10, it is not practical to recompute the routing sufficiently quickly enough to accommodate the high rate of arrivals of e2e connections. On the other hand, max-flow based routing is highly efficient, and hence, the scheme described in Section 6.1.2.1 could be applied to accommodate time-varying offered load [1].

Thus, three approaches are considered.

1. Compute the optimal multipath routing via Algorithm 10 with $w_\phi = 1$ for all ϕ. This routing is left fixed, but the weights are set according to $w_\phi(t) = 1/N_\phi(t)$ and a new schedule is computed whenever a new flow starts or ends.

2. Compute the optimal multipath routing via Algorithm 10 with $w_\phi = 1$ for all ϕ and then convert this multipath routing into single path routing as described in Section 8.5.5. This routing is left fixed and the weights are set according to

[1] Since many flows in the Internet consists of very few packets, it is not practical to recompute anything every time a new connection starts. However, there are many options for accommodating very small flows. Hence, this issue is left for future work.

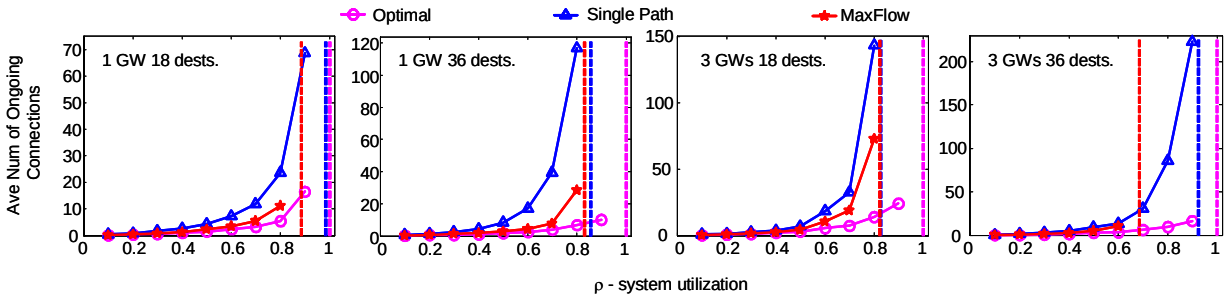


Figure 8.9: Average number of ongoing connections when subjected to Poisson connection arrivals and exponentially distributed connection size. Utilization of 1 corresponds to the maximum capacity provided by optimal routing

$w_\phi(t) = 1/N_\phi(t)$ and a new schedule is computed whenever a new flow starts or ends.

3. Whenever a new e2e connection starts or an old connection finished, set $w_\phi(t) = 1/N_\phi(t)$ and compute new routing and scheduling with the max-flow based scheme described in Section 6.1.2.1.

Figure 8.9 shows the average number of ongoing connections in the network for different topologies and different levels of network utilization. The connection interarrival time was exponentially distributed as was the connection size. The x-axis is in terms of the utilization under optimal routing. The vertical lines in Figure 8.9 indicate the maximum throughput achievable by the different schemes. (Hence, the vertical line at $\rho = 1$ corresponds to the maximum utilization that optimal routing can support.) As discussed in Sections 6.1.2.1 and 8.5.3, max-flow based routing provides less throughput than optimal multipath routing and single path routing developed in Section 8.5.5. However, max-flow routing dynamically varies the routing in response to variations in demand, it is able to significantly outperform single path routing, until the offered load nears the maximum throughput that max-flow routing can support. On the other hand, even with static routing, optimal

multipath routing significantly outperforms max-flow based routing. and, of course, optimal routing also provides the highest throughput.

Therefore, we suggest the following. Since optimal multipath routing provides the maximum throughput, it should be used. However, since the computation time of multipath routing is considerable, when a topology change occurs, max-flow routing should be used until multipath routing can be recomputed.

Note that there are many alternatives that were not investigated here. Indeed, it is clear that maximizing throughput does not necessary provide the best performance even in terms of closely related metrics such as the average time to complete a connection. On the other hand, while well planned wired network rarely reaches full throughput, and hence performance at lower utilization might be more relevant than peak throughput. On the other hand, since wireless bandwidth is scarce, it is possible that performance near maximum throughput is critical.

8.7 Conclusion

Employing a recently developed algorithm to compute optimal schedules under realistic propagation models, this work explores joint optimal routing and scheduling. An iterative algorithm for optimal routing is developed along with an approximation of the optimal algorithm. However, in the networks examined, the approximation yields the same throughput as the optimal algorithm. Several aspects of the behavior of the approximating algorithm are explored. For example, it is found that in realistic topologies, the proposed algorithm improves throughput by 60% over least hop routing and 20-35% over routing based on max-flow. Behavior of the routing in the face of time-varying offered load is also examined. It is found that a routing algorithm that provided higher throughput does not necessarily translate into better performance even in terms of metrics that are closely related to throughput. Future work will consider these aspects of performance since they are more aligned with user perceived performance.

8.8 Appendix

8.8.1 Proof of Proposition 23

Proof. We consider the optimization with nonlinear objective function, i.e., $G(\mathbf{f}) = \sum_{\phi\in\Phi} U_\phi\left(\sum_k f_{\phi,k}\right)$. The case of $G(f) = \min_{\phi\in\Phi} w_\phi \sum_k f_{\phi,k}$ follows the same approach.

Consider the optimization problem

$$\max \sum_{\substack{\phi\in\Phi \\ \phi\neq\phi^+}} U_\phi\left(\sum_k f_{\phi,k}\right) + U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \tag{8.8}$$

$$\text{subject to: } \sum_{\{(\phi,k)|x\in\mathcal{P}(\phi,k)\}} f_{\phi,k} \leq \sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} R(\mathbf{v},x) \;\; \forall x$$

$$f_{\phi^+,k^+} \leq b_{\phi^+,k^+}$$

$$0 \leq f_{\phi,k}$$

$$\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1$$

$$0 \leq \alpha_{\mathbf{v}} \text{ for each } \mathbf{v}\in\mathcal{V}.$$

The Lagrangian for (8.8) is

$$\begin{aligned}
& L\left(\overrightarrow{f}, \overrightarrow{\alpha}, \gamma_{\phi^+,k^+}, \overrightarrow{\mu}, \lambda, \overrightarrow{\sigma}\right) \\
= \; & -U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \\
& + \sum \mu_l \left(\sum_{\{(\phi,k)|x\in P(\phi,k)\}} f_{\phi,k} - \sum \alpha_{\mathbf{v}} R(\mathbf{v},x)\right) \\
& + \gamma_{\phi^+,k^+}\left(f_{\phi^+,k^+} - b_{\phi^+,k^+}\right) - \sigma_{\phi^+,k^+} f_{\phi^+,k^+} \\
& + \lambda\left(\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} - 1\right) - \sum_{\substack{\phi\in\Phi \\ \phi\neq\phi^+}} U_\phi\left(\sum_k f_{\phi,k}\right) \\
& - \sum_{(\phi,k)\neq\left(\phi^+,k^+\right)} \sigma_{\phi,k} f_{\phi,k}
\end{aligned}$$

where all Lagrange multipliers, $\gamma_{\phi^+,k^+}, \boldsymbol{\mu}, \lambda, \boldsymbol{\sigma}$ are non-negative. The dual function is

$$\begin{aligned} & q\left(\gamma_{\phi^+,k^+}, \boldsymbol{\mu}, \lambda, \boldsymbol{\sigma}\right) \\ = & \inf_{f_{\phi^+,k^+}} -U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \\ & + f_{\phi^+,k^+}\left(\sum_{l\in P\left(\phi^+,k^+\right)} \mu_l + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+}\right) + D \end{aligned} \tag{8.9}$$

where D represents other terms that do not depend on f_{ϕ^+,k^+}. From (8.9), f_{ϕ^+,k^+} is such that

$$\begin{aligned} 0 = & \frac{d}{df_{\phi^+,k^+}}\left(-U_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right)\right. \\ & \left. + f_{\phi^+,k^+}\left(\sum_{x\in P\left(\phi^+,k^+\right)} \mu_x + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+}\right)\right) \end{aligned}$$

or

$$\begin{aligned} & U'_{\phi^+}\left(\sum_{k\neq k^o} f_{\phi^+,k} + f_{\phi^+,k^+}\right) \\ = & \sum_{l\in P\left(\phi^+,k^+\right)} \mu_l + \gamma_{\phi^+,k^+} - \sigma_{\phi^+,k^+}, \end{aligned} \tag{8.10}$$

where $U'_{\phi^+}(f) = \frac{d}{df} U_{\phi^+}(f)$.

Consider the set of problems with different values of $b_{\phi^+k^+}$, specifically, consider the sequence of problems indexed by n with $b_{\phi^+,k^+}(n) > 0$ and $\lim_{n\to\infty} b_{\phi^+,k^+}(n) = 0$. In the limit, flow $\left(\phi^+, k^+\right)$ not being used (i.e., $f_{\phi^+,k^+} = 0$), and the solution is the same as the original optimization (e.g., the multipliers $\boldsymbol{\mu}$

and λ are unchanged). Since $f_{\phi^+,k^+} = 0$, we have $U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k} + f_{\phi^+,k^+}\right) = U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k}\right)$. Thus, from (8.10), if

$$U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k}\right) < \sum_{x\in P\left(\phi^+,k^+\right)} \mu_x \tag{8.11}$$

then $\gamma_{\phi^+,k^+} > 0$. Thus, by sensitivity analysis of Lagrange multiplier theory, if and only if (8.11) holds, then the throughput will increase if b_{ϕ^+,k^+} is increased. That is, the throughput will increase if data is allowed to be sent over path $P\left(\phi^+,k^+\right)$.

Next we show that

$$U'_{\phi^+}\left(\sum_{k\neq k^+} f_{\phi^+,k}\right) = \sum_{l\in P\left(\phi^+,k^o\right)} \mu_l. \tag{8.12}$$

To see this, the same analysis as above can be done but with k^+ replaced with k^o. Furthermore, since data is allowed to flow over flow $\left(\phi^+,k^o\right)$ we have $b_{\phi^+,k^o} = \infty$. Hence, the constraint $f_{\phi^+,k^o} \leq b_{\phi^+,k^o}$ is not active, and therefore, $\gamma_{\phi^+,k^o} = 0$. Moreover, since $f_{\phi^+,k^o} > 0$, the constraint $f_{\phi^+,k^o} \geq 0$ is not active, and hence $\sigma_{\phi^+,k^o} = 0$. Hence, (8.10) implies (8.12). Finally, substituting the right-hand side of (8.12) into the left-hand side of (8.11) results in (8.5). ■

8.8.2 Proof of Proposition 27

Proof. In Problem (8.1), not all flows are considered. We denote the set of considered paths $\mathcal{P}$. We define a shadow problem where all possible paths are included, but the data rate along any path not in $\mathcal{P}$ is restricted so that the data rate crossing these flows is no greater than ε. Hence, as $\varepsilon \to 0$, the solutions to the shadow problem and the original problem are the same. However, in the case of the shadow problem, the Lagrange multiplier for each link is known. Specifically, let $\mathcal{L}$ be the set of links used by any path in $\mathcal{P}$ and let $\mathcal{A}$ be the set of all links. Then, by computing the optimal schedule when the paths $\mathcal{P}$ are considered, we can determine

μ_x for $x \in \mathcal{L}$. And, from the optimal schedule where all paths are considered, μ_x is known for all links $x \in \mathcal{A}$. It is straightforward to verify that for $x \in \mathcal{L}$, the μ_x is the same in both cases. Moreover, λ is also the same.

Let $z \in \mathcal{A} \backslash \mathcal{L}$. An assignment that includes z is $\{z\} \cup MWIS\left(\mathcal{CG}\left[\mathcal{A} \backslash \left(\{z\} \cup \chi\left(z\right)\right)\right]\right)$. By Proposition 2,

$$\mu_z R_z + \sum_{x \in MWIS(\mathcal{CG}[\mathcal{A} \backslash (\{z\} \cup \chi(z))])} \mu_x R_x \leq \lambda.$$

Since

$$\begin{aligned} & \sum_{x \in MWIS(\mathcal{CG}[\mathcal{L} \backslash (\{z\} \cup \chi(z))])} \mu_x R_x \\ \leq & \sum_{x \in MWIS(\mathcal{CG}[\mathcal{A} \backslash (\{z\} \cup \chi(z))])} \mu_x R_x \end{aligned}$$

we have

$$\mu_z R_z \leq \lambda - \sum_{x \in MWIS(\mathcal{CG}[\mathcal{L} \backslash (\{z\} \cup \chi(z))])} \mu_x R_x$$

which is the desired result. ■

Chapter 9

OPTIMAL SCHEDULES WITH POWER CONTROL AND MULTIPLE BIT RATES

9.1 Introduction

Much of the previous research on optimal scheduling has only considered a single transmission power and a single bit-rate[1]. However, today's implementations of 802.11a/b/g/n and 802.16 support a wide range of bit-rates and transmission powers. This chapter develops techniques to accommodate a set of transmission powers and a set of bit-rates into optimal scheduling; thus, the schedule not only specifies when a link should transmit, but also specifies which bit-rate and transmission power are used for the transmission. This chapter also explores the impact that transmission power and bit-rates have on the performance of scheduling, which includes the computation time and the computed throughput.

The schedule optimization over a set of bit-rates and a set of transmission powers results in a higher dimension problem than when the optimization is over only a single bit-rate and a single transmission power. In most cases, this added flexibility results in an increase in the computation time as well as an increase in the computed throughput. Thus, this work examines heuristic schemes that consider various subsets of bit-rates and transmission powers in hope of increasing the computed throughput without greatly increasing the computation time.

[1] Important exceptions include [47] and [8].

The main conclusion is that there is no need to consider a continuum of transmission powers, rather only two transmission powers for each link need to be considered. Considering a continuum of transmission powers greatly increases the computation time, but only slightly increases the computed throughput. Similarly, optimizing over an increasing number of bit-rates increases the computed throughput. As compared to using a single bit-rate, using two bit-rates results in a moderate increase in throughput and a relatively small increase in computation time. However, as more bit-rates are considered, the increase in throughput is smaller and the increase in computation time is larger. Roughly speaking, the "sweet spot", where considering more bit-rates and/or considering more transmission powers greatly increases computation time and only incrementally improves throughput occurs with two or three bit-rates and two transmission powers. The results in this work provide guidelines for making such a trade-off between computed throughput and computation time.

The remainder of this chapter is as follows. In the next section, notation and the problem definitions are given. There are two classes of techniques, namely, one where the transmission power can take a continuum of values and one where the transmission power can only take values from a discrete set. Section 9.3 presents techniques to compute optimal schedules which are extended to support multiple bit rates and multiple transmission powers. Various subsets of bit-rates and transmission powers over which the optimization is performed are developed in Section 9.4. Section 9.5.1 discusses the set-up for the computational experiments and Sections 9.5.2 to 9.5.6 present the results from the computational experiments. Finally, concluding remarks are in Section 9.6.

9.2 System Model and Problem Formulation

When considering multiple powers and bit-rates, the concept of a logical link arises. For example, a physical link may support M modulation/coding schemes.

This can be modeled as M logical links, each with a different bit-rate, but between the same pair of physical nodes. In this case, the logical link x_m would refer to the physical link x transmitting at the mth modulation/coding scheme. We drop the terms physical and logical unless it is unclear from context.

A schedule is defined as a convex combination of assignments, where an assignment specifies which logical links are transmitting along with the transmission power and bit-rates used by the transmitting links. Since links are permitted to use different modulation schemes and different transmission powers, the definition of an assignment must be generalized. The next two sections develop the case where the transmission powers can take a continuum of values and the case where the transmission powers can only take values in a discrete set.

9.2.1 The Continuous Power Case

While today's radios only support a relatively small number of bit-rates, they could support nearly a continuum of transmission powers. In the case of multiple bit-rates and variable transmission power, it is convenient to write an assignment in terms of a pair $(\mathbf{v}, \mathbf{p})$ where $\mathbf{v} \in \{0,1\}^{L \times M}$ and $v_{x,m} = 1$ implying that link x is transmitting with modulation scheme m. The vector $\mathbf{p}$ specifies the transmission powers, i.e., p_x is the transmission power of link x. Of course, we require $p_x = 0$ if $v_{x,m} = 0$ for all m. Thus, $(\mathbf{v}, \mathbf{p}) \in \{0,1\}^{L \times M} \times [0, p_{\max}]^L$, where $p_{\max}$ is the maximum allowable transmission power. Since the space of all pairs of assignments and transmission powers is very large, we often consider a subset $\mathcal{VP} \subset \{0,1\}^{L \times M} \times [0, p_{\max}]^L$, which is referred to as the *set of considered assignments.*

We define the SINR at the receiver of link x as

$$SINR(x, \mathbf{p}) := \frac{H_{x,x} p_x}{\sum_{y \neq x} H_{y,x} p_y + \mathcal{N}_0},$$

where $H_{x,x}$ is the channel gain across link x, $H_{y,x}$ is the channel gain from the transmitter of link y to the receiver of link x, and $\mathcal{N}_0$ is the channel noise. Then,

the data rate across link x during assignment $(\mathbf{v}, \mathbf{p})$ is approximated as

$$R\left(\left(\mathbf{v}, \mathbf{p}\right), x_m\right) = \begin{cases} R_{x_m} \text{ if } SINR\left(x, \mathbf{p}\right) > \mathcal{T}\left(x, m\right) \\ \qquad\qquad \text{and } v_{x_m} = 1,\ v_{x_n} = 0 \text{ for } n \neq m \\ 0 \text{ otherwise,} \end{cases} \tag{9.1}$$

where R_{x_m} is link x's mth nominal data rate and $\mathcal{T}\left(x, m\right)$ is the SINR required to achieve this data rate. Specifically,

$$\mathcal{T}\left(x, m\right) = \min\left\{ \mathcal{T} \left| \frac{PSP\left(\frac{H_{x,x} p_{\max}}{\mathcal{N}_0}, m\right) - PSP\left(\mathcal{T}, m\right)}{PSP\left(\frac{H_{x,x} p_{\max}}{\mathcal{N}_0}, m\right)} < G1 \right. \right\},$$

where $PSP\left(\mathcal{T}, m\right)$ is the probability of successfully transmitting a packet when the SINR is $\mathcal{T}$ and the mth modulation scheme is used, $p_{\max}$ is the maximum allowable transmission power, and $G1 = 0.01$.

A schedule is a convex combination of assignments. Specifically, a schedule is a set $\left\{\alpha_{(\mathbf{v},\mathbf{p})} : (\mathbf{v}, \mathbf{p}) \in \mathcal{VP}\right\}$ where $\sum_{(\mathbf{v},\mathbf{p})\in\mathcal{VP}} \alpha_{(\mathbf{v},\mathbf{p})} \leq 1$ and $\alpha_{(\mathbf{v},\mathbf{p})} \geq 0$. With this notation, the total data rate that the schedule $\boldsymbol{\alpha}$ provides over link x is $\sum_{(\mathbf{v},\mathbf{p})\in\mathcal{VP}} \alpha_{(\mathbf{v},\mathbf{p})} \sum_{m=1}^{M} R\left(\left(\mathbf{v}, \mathbf{p}\right), x_m\right)$. In this scenario, the throughput maximization problem is

$$\max_{\boldsymbol{\alpha}, \mathbf{f}} G\left(\mathbf{f}\right) \tag{9.2a}$$

subject to:

$$\sum_{\{\phi | x \in P(\phi)\}} f_\phi \leq \sum_{(\mathbf{v},\mathbf{p})\in\mathcal{VP}} \alpha_{\mathbf{v},\mathbf{p}} \sum_{m=1}^{M} R\left(\left(\mathbf{v}, \mathbf{p}\right), x_m\right) \text{ for each link } x \tag{9.2b}$$

$$\sum_{(\mathbf{v},\mathbf{p})\in\mathcal{VP}} \alpha_{(\mathbf{v},\mathbf{p})} \leq 1 \tag{9.2c}$$

$$0 \leq \alpha_{(\mathbf{v},\mathbf{p})} \text{ for each } (\mathbf{v}, \mathbf{p}) \in \mathcal{VP}, \tag{9.2d}$$

where $\mathbf{f}$ is the vector of flow rates. The function G is referred to as the *throughput metric*, and $G\left(\mathbf{f}\right) = \min_{\phi\in\Phi} w_\phi f_\phi$ is considered in this work.

9.2.2 The Discrete Power Case

Problem (9.2) can be simplified by considering a discrete set of transmission powers instead of a continuum. While this approach is similar to the approach above, it requires slightly different notation.

Suppose that there are M modulation/coding schemes and S transmission powers available for each link, then associated with each physical link x is the set of logical links $x_{m,s}$ with $1 \leq m \leq M$ and $1 \leq s \leq S$. In this case, the assignment specifies which links are transmitting, their bit-rate, and their transmission power. Specifically, an assignment $\mathbf{v} \in \{0,1\}^{L \times M \times S}$, where $v_{x_{m,s}} = 1$ implies that the physical link x is transmitting at bit-rate m and with power $p_{x_{m,s}}$. Note that, $p_{x_{m,s}}$ need not be the same as $p_{y_{m,s}}$ or $p_{x_{n,s}}$, that is, the set of transmission powers depend on the link and the modulation scheme. Due to the large size of $\{0,1\}^{L \times M \times S}$, we consider subsets $\mathcal{V} \subset \{0,1\}^{L \times M \times S}$, where $\mathcal{V}$ is referred to as the *set of considered assignments*.

The data rate across logical link $x_{m,s}$ during assignment $\mathbf{v}$ is denoted by $R(\mathbf{v}, x_{m,s})$. As above, a simple binary approximation is used to define $R(\mathbf{v}, x_{m,s})$. Specifically,

$$R(\mathbf{v}, x_{m,s}) = \begin{cases} R_{x_{m,s}} \text{ if } v_{y_{n,t}} = 0 \text{ for all } y_{n,t} \in \chi(x_{m,s}), \\ \qquad y_{n,t} \neq x_{m,s} \text{ and } v_{x_{n,t}} = 0 \text{ for } x_{n,t} \neq x_{m,s} \\ 0 \text{ otherwise,} \end{cases} \tag{9.3}$$

where $\chi(x_{m,s})$ is the set of logical links that are in conflict with $x_{m,s}$, i.e., $y_{n,t} \in \chi(x_{m,s})$ if simultaneous transmissions over logical link $x_{m,s}$ and logical link $y_{n,t}$ are not possible. $R_{x_{m,s}}$ is the nominal data rate over logical link $x_{m,s}$. The details of the communication model for multi-bit rates and multi-powers are provided in Section 4.2.5.

With this notation, the throughput maximization problem with discrete transmission powers is

$$\max_{\boldsymbol{\alpha},\mathbf{f}} G(\mathbf{f}) \tag{9.4a}$$

subject to:

$$\sum_{\{\phi|x\in P(\phi)\}} f_\phi \leq \sum_{\mathbf{v}\in\mathcal{V}} \sum_{s=1}^{S} \sum_{m=1}^{M} \alpha_{\mathbf{v}} R(\mathbf{v}, x_{m,s}) \text{ for each link } x \tag{9.4b}$$

$$\sum_{\mathbf{v}\in\mathcal{V}} \alpha_{\mathbf{v}} \leq 1 \tag{9.4c}$$

$$0 \leq \alpha_{\mathbf{v}} \text{ for each } \mathbf{v} \in \mathcal{V}. \tag{9.4d}$$

9.3 Optimal Scheduling

9.3.1 Algorithms

The main challenge to solve (9.2) and (9.4) is that the space $\mathcal{VP} = \{0,1\}^{L\times M} \times [0, p_{\max}]^L$ and $\mathcal{V} = \{0,1\}^{L\times M\times S}$ are very large. Thus, the size of the space over which the optimization is performed must be reduced. In Chapter 3, a technique was developed that focuses on finding a small set of considered assignments $\mathcal{VP}$ (or $\mathcal{V}$) such that the throughput is optimal when (9.2) (or (9.4)) is solved over $\mathcal{VP}$ (or $\mathcal{V}$). Algorithms 12 and 11 find such a set of assignments iteratively. It was shown that these algorithms converge geometrically when $G(\mathbf{f}) = \min_{\phi\in\Phi} w_\phi f_\phi$ and algebraically when $G(\mathbf{f}) = \sum_{\phi\in\Phi} w_\phi \log(f_\phi)$. However, these algorithms require solving (9.6) or (9.5) each iteration.

9.3.2 Finding New Assignments - The Discrete Power Case

9.3.2.1 Basic Approach

It is well known that solving (9.5) is equivalent to solving the graph theoretic problem known as the maximum weighted independent set (MWIS) problem. The

Algorithm 11 Computing an Optimal Schedule (Discrete Powers Case)

1: Set $k = 0$, then select an initial set of assignments $\mathcal{V}(0)$ and a empty assignment $\mathbf{v}^+$.
2: **repeat**
3: Set $\mathcal{V}(k+1) = \mathcal{V}(k) \cup \mathbf{v}^+$, and set $k = k + 1$.
4: Solve (9.4) for $\mathcal{V} = \mathcal{V}(k)$ and compute $\mu(k)$ and $\lambda(k)$, the Lagrange multipliers associated with constraints (9.4b) and (9.4c), respectively.
5: Search for an assignment $\mathbf{v}^+ \notin \mathcal{V}(k)$ that solves

$$\mathbf{v}^+ \in \arg\max_{\mathbf{v} \notin \mathcal{V}(k)} \sum_{x=1}^{L} \mu_x(k) \sum_{m=1}^{M} \sum_{s=1}^{S} R(\mathbf{v}, x_{m,s}). \tag{9.5}$$

6: **until**
7: No such an assignment is found or $\sum_{x=1}^{L} \mu_x(k) \sum_{m=1}^{M} \sum_{s=1}^{S} R(\mathbf{v}^+, x_{m,s}) \leq \lambda(k)$.

exact method described in Section 5.2.4.1 can be extended. Specifically, a set of cliques $\left\{Q_i,\ i = 1, 2, ...\overline{Q}\right\}$ are found such that if $y_{n,t} \in \chi(x_{m,s})$, then there is a clique Q_i such that $x_{m,s} \in Q_i$ and $y_{n,t} \in Q_i$. Then, MWIS problem can be formulated as

$$\begin{aligned} \max_{\mathbf{v}} & \sum_{x=1}^{L} \mu_x \sum_{m=1}^{M} \sum_{s=1}^{S} R_{x_{m,s}} v_{x_{m,s}} \\ \text{subject to:} & \sum_{x_{m,s} \in Q_i} v_{x_{m,s}} \leq 1 \text{ for } i = 1, 2, ..., \overline{Q} \\ & v_{x_{m,s}} \in \{0, 1\}. \end{aligned} \tag{9.7}$$

9.3.2.2 Removing Multi-Conflicts

Binary communication models do not account for the impact of the aggregate of interference from several simultaneously transmitting links. Removing multi-conflicts described in Section 5.3 can be extended as follows.

Let $\mathbf{v}^+$ be a new assignment found by solving (9.7). Check whether this assignment has any multi-conflicts, by determining whether there is a $x_{m,s}$ with

Algorithm 12 Computing an Optimal Schedule (Continuous Power Case)

1: Set $k = 0$, then select an initial set of assignments $\mathcal{VP}(0)$ and a empty assignment-transmission power pair $(\mathbf{v}^+, \mathbf{p}^+)$.
2: **repeat**
3: Set $\mathcal{VP}(k+1) = \mathcal{VP}(k) \cup (\mathbf{v}^+, \mathbf{p}^+)$, and set $k = k+1$.
4: Solve (9.2) for $\mathcal{VP} = \mathcal{VP}(k)$ and compute $\mu(k)$ and $\lambda(k)$, the Lagrange multipliers associated with constraints (9.2b) and (9.2c), respectively.
5: Search for an assignment-transmission power pair $(\mathbf{v}^+, p^+) \notin \mathcal{VP}(k)$ that solves

$$\left(\mathbf{v}^+, \mathbf{p}^+\right) \in \arg \max_{(\mathbf{v},\mathbf{p}) \notin \mathcal{VP}(k)} \sum_{x=1}^{L} \mu_x(k) \sum_{m=1}^{M} R\left((\mathbf{v}, \mathbf{p}), x_m\right). \tag{9.6}$$

6: **until**
7: No such an assignment is found or $\sum_{x=1}^{L} \mu_x(k) \sum_{m=1}^{M} R\left((\mathbf{v}^+, \mathbf{p}^+), x_m\right) \leq \lambda(k)$.

$v^+_{x_{m,s}} = 1$ and

$$\frac{PSP\left(\frac{H_{x,x} p_{x_{m,s}}}{\mathcal{N}_0}, m\right) - PSP\left(\frac{H_{x,x} p_{x_{m,s}}}{\sum_{y=1, y\neq x}^{L} \sum_{n=1}^{M} \sum_{t=1}^{S} H_{y,x} p_{y_{n,t}} v_{y_{n,t}} + \mathcal{N}_0}, m\right)}{PSP\left(\frac{H_{x,x} p_{x_{m,s}}}{\mathcal{N}_0}, m\right)} > G1.$$

If such a $x_{m,s}$ exists, then add the found multi-conflict into the ILP (9.7) as follows. Let $\mathcal{C} = \left\{x_{m,s} | v^+_{x_{m,s}} = 1\right\}$, that is, $\mathcal{C}$ is the set of links that make up the multi-conflict. Intuitively, $\mathcal{C}$ should be the smallest set of links that form the multi-conflict at link $x_{m,s}$. This multi-conflict can be removed from consideration by using the following ILP problem to search for new assignments,

$$\max_{\mathbf{v}} \sum_{x=1}^{L} \mu_x \sum_{m=1}^{M} \sum_{s=1}^{S} R_{x_{m,s}} v_{x_{m,s}} \tag{9.8a}$$

$$\text{subject to: } \sum_{x_{m,s} \in Q_i} v_{x_{m,s}} \leq 1 \text{ for } i = 1, 2, ..., \overline{Q} \tag{9.8b}$$

$$\sum_{x_{m,s} \in \mathcal{C}} v_{x_{m,s}} \leq |\mathcal{C}| - 1 \tag{9.8c}$$

$$v_{x_{m,s}} \in \{0, 1\}, \tag{9.8d}$$

where $|\mathcal{C}|$ is the number of elements in $\mathcal{C}$. Again, it is determined whether the assignment found by solving (9.8) has multi-conflicts and if so, another constraint

such as (9.8c) is added to the ILP. This process continues until an assignment is found that does not have a multi-conflict.

9.3.3 Finding New Assignments - The Continuous Power Case

Here we seek to find a new assignment-transmission power pair that solves (9.6), where R is given by (9.1). Thus, the desired assignment-transmission power pair is the solution to

$$\begin{aligned} &\max_{(\mathbf{v},\mathbf{p})} \sum_{x=1}^{L} \mu_x \sum_{m=1}^{M} R_{x_m} v_{x_m} \\ &\text{such that: } \frac{H_{x,x}p_x}{\sum_{y\neq x} H_{y,x}p_y + \mathcal{N}_0} > \mathcal{T}(x,m) \text{ if } v_{x_m} = 1. \end{aligned} \tag{9.9}$$

In theory, constraint (9.9) is equivalent to

$$H_{x,x}p_x - \mathcal{T}(x,m)\left(\sum_{y\neq x} H_{y,x}p_y + \mathcal{N}_0\right) > \infty \times (v_{x_m} - 1), \tag{9.10}$$

since, when $v_{x_m} = 0$, (9.10) is always true (assuming that $\infty \times 0 = 0$). Of course, in practice, ∞ is replaced with some large number. While further tuning is possible, we have found that new assignment-transmission power pairs can be found by solving the following mix-integer programming problem.

$$\max_{\mathbf{v}} \sum_{x=1}^{L} \mu_x \sum_{m=1}^{M} R_{x_m} v_{x_m}$$

$$\text{subject to: } H_{x,x}p_x - \mathcal{T}(x,m)\left(\sum_{y\neq x} H_{y,x}p_y + \mathcal{N}_0\right) > \tag{9.11a}$$

$$- \Gamma_{x_m}(1 - v_{x_m}) \quad \text{for each link } x \text{ and modulation } m = 1, 2, ..., M$$

$$\sum_{m=1}^{M} v_{x_m} \leq 1 \text{ for each link } x \tag{9.11b}$$

$$\sum_{m=1}^{M} v_{x_m} + \sum_{m=1}^{M} v_{y_m} \leq 1 \text{ if } y \in NE(x) \tag{9.11c}$$

$$0 \leq p_x \leq p_{\max} \text{ for each link } x$$

$$v_{x_m} \in \{0, 1\},$$

where

$$\Gamma_{x,m} = \mathcal{T}(x,m)\left(\sum_{y \neq x} H_{y,x} P_{\max} + \mathcal{N}_0\right) \tag{9.12}$$

and where $NE(x)$ is the set of links y such that y's transmitter or receiver is the same as either x's transmitter or receiver.

9.4 A Prior Bit-Rate and Transmission Power Selection

A critical challenge facing throughput maximization when multiple powers (or variable power) and multiple bit-rates are available is the large optimization space. Algorithms 12 and 11 dramatically reduce the optimization space. However, if a large number of bit-rates and transmissions powers are available, then the problem remains computationally complex. The size of these problems can be reduced by considering only a subset of the possible bit-rates and only a subset of all possible transmission powers. The subset of bit-rates and transmission powers are referred to as *the set of considered bit-rates* and *the set of considered transmission powers*, respectively. There is no surprise that such a prior selection might reduce the computed throughput. Also, it should be noted that these sets were designed after extensive experimentation, but are purely based on heuristics.

9.4.1 Sets of Considered Bit Rates

Assume that different modulation/coding schemes are indexed by an integer between 1 and M, where we assume that a larger index implies a higher bit-rate. To make the problem concrete, we will focus on the modulation/coding used by 802.11a. Figure 9.1 shows the relationship between the probability of successful packet transmission (PSP) and the SINR where 1000B packets are assumed. This relationship is assumed to be the same as the relationship between PSP and SNR. Note that 802.11a supports eight bit-rates, and yet Figure 9.1 only shows seven bit-rates, 9 Mbps is missing. The reason is that the relationships between SNR and

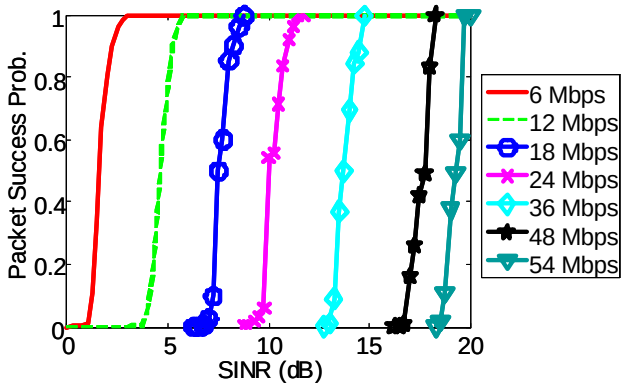


Figure 9.1: Relationship between packet success probability (PSP) and SINR for 802.11a

PSP at 9 Mbps and at 12 Mbps are the same. Thus, there is no need to consider the 9 Mbps bit-rate.

For a particular SNR, a link can support a maximum data rate $m^*(x,p)$ which is derived from

$$m^*(x,p) \quad := \arg\max_m PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0}, m\right) BR(m)$$
$$\text{subject to} \frac{PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0}, m\right) - PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0} - G2 - G3, m\right)}{PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0}, m\right)} \leq G1,$$

where p is the maximum allowable power. For simplicity of presentation, we use the following notation.

Let $m^* = m^*(x, p_{\max})$, $m^* - k = \max(m^*(x, p_{\max}) - k, 1)$, and $m^*/i = round(m^*(x, p_{\max})/i)$ where k and i are integers.

Once the maximum bit rate is selected, a wide range of possible approaches are available for selecting the set of bit rates. For example, a set of two bit rates $\{m^*, m^* - 1\}$ of each link can be considered. Note that m^* depends on the link. Thus, if the set of considered bit-rates is $\{m^*, m^* - 1\}$, then link x uses $m^*(x, p_{\max})$

and $\max(m^*(x, p_{\max}) - k, 1)$, and thus, different links do not necessarily use the same modulation/coding schemes.

9.4.2 Sets of Considered Transmission Powers

Once the set of considered bit-rates is selected, the set of transmission powers can be selected. Two obvious options are

- **Max Power** $\mathcal{P}(x, m) = \{p_{\max}\}$
- **Continuum of Powers** $\mathcal{P}(x, m) = [0, p_{\max}]$.

Between these approaches is a wide range of techniques to select a finite set of transmission powers. However, for a particular modulation/coding scheme, if the transmission power is too low, then transmission is not possible. Thus, we define the minimum transmission power $p_{\min}(x, m)$ via

$$p_{\min}(x, m) := \min \left\{ p : \frac{PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0}, m\right) - PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0} - G2 - G3, m\right)}{PSP\left(\frac{H_{x,x}p}{\mathcal{N}_0}, m\right)} \leq G1 \right\}.$$

With this, a third approach to selecting a set of candidate transmission powers is

- **Two-Powers** $\mathcal{P}(x, m) = \{p_{\min}(x, m), p_{\max}\}$

While there are many ways that two transmission powers can be selected, this work only considers the above two powers. Hence, the term "two transmission powers" refers to this specific selection of transmission powers. While other schemes are also possible (e.g., three transmission powers), as discussed in Section 9.5.4, in the topologies explored here, the above three sets of power provide a sufficiently rich spectrum of techniques.

9.5 Numerical Experiments

9.5.1 Computational Experimental Set-up

The topologies generated in Section 6.1.1 and the SINR protocol model with *Without Acks* were considered. Each topology was based on a different 6×6 city block region that was randomly located within a 2 km^2 region of downtown Chicago. Various node densities were investigated. Specifically, the number of gateways was 1, 3, and 6, and the number of wireless routers was 18, 36, 54, 72, and 90. The wireless routers and gateways were uniformly distributed throughout the 6×6 city block region. Ten samples of each topology were generated (hence, 150 topologies in total).

As mentioned in Section 9.4.1, 802.11a data rates were used, and the throughput metric used is $G(\mathbf{f}) = \min_{\phi \in \Phi} f_\phi$. The computations below were performed on a 2.4MHz AMD FX-53 processor with 8GB RAM.

9.5.2 Selecting a Set of Considered Bit-Rates and Considered Transmission Powers

Clearly, increasing the number of considered bit-rates and transmission powers might increase the computed throughput, but also might result in an increase in the computation time. Thus, a single best set of bit-rates and transmission powers does not exist. Instead, this section will present a sequence of sets of bit-rates and transmission powers that result in increasing the computed throughput while limiting the increase in computation time. Furthermore, we seek to understand the trade-off between the computed throughput and the computation time. This information allows one to select a set of bit-rates and transmission power so as to make a rational trade-off between the computed throughput and the computation time.

The first step toward this objective is to select sets of considered bit-rates that provide a good trade-off between the computed throughput and the computation time. To this end, we use the following notation.

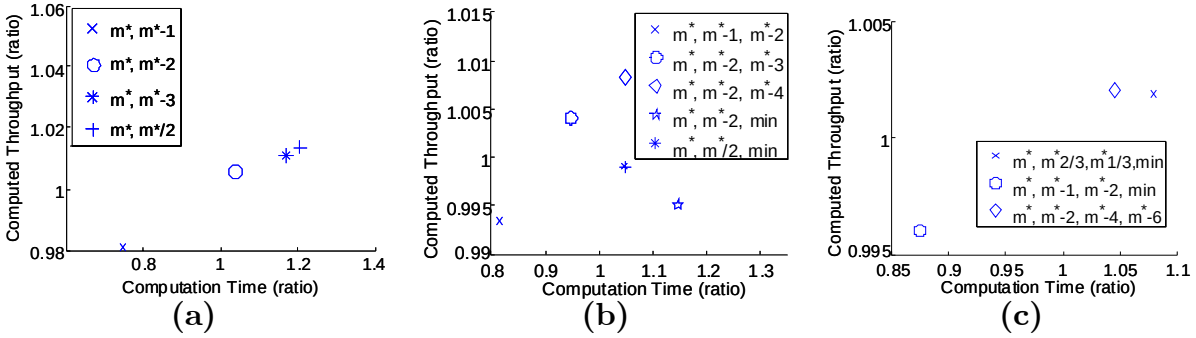


Figure 9.2: Relative Performance of Set of Considered Bit-Rates. (a), (b), and (c) show the relative performance of sets of bit-rates with 2, 3, and 4 elements, respectively. The performance is relative to the average performance of other sets of bit-rates with the same number of elements and is averaged over all topologies and all three sets of considered transmission powers.

$C^{\tau}_{\sigma,\omega}$ is the throughput computed for the τth topology with σ transmissions powers, and ω is the set of considered bit-rates. $\sigma = 1$ implies that $\mathcal{P}(x,m) = \{p_{\max}\}$, $\sigma = 2$ implies that $\mathcal{P}(x,m) = \{p_{\min}(x,m), p_{\max}\}$, and $\sigma = \infty$ implies that $\mathcal{P}(x,m) = [0, p_{\max}]$.

$T^{\tau}_{\sigma,\omega}$ is the time required to compute the optimal schedule for the τth topology with σ transmission powers and the ωth set of considered bit-rates.

Let $\Omega(i)$ be a set of different sets of considered bit-rates where each set of considered bit-rates has i elements. Suppose that ω has i elements, then we compare ω to the other sets of considered bit-rates in $\Omega(|\omega|)$, where $|\omega|$ is the number of elements in ω, i.e., the number of bit-rates considered. Specifically, for a particular ω, the vector

$$\left(\frac{1}{\sum_{\sigma\in\{1,2,\infty\}}\sum_{\tau=1}^{10} 1} \sum_{\sigma\in\{1,2,\infty\}}\sum_{\tau=1}^{10} \frac{T^{\tau}_{\sigma,\omega}}{\frac{1}{|\Omega(|\omega|)|}\sum_{\varphi\in\Omega(|\omega|)} T^{\tau}_{\sigma,\varphi}}, \right. \tag{9.13}$$

$$\left. \frac{1}{\sum_{\sigma\in\{1,2,\infty\}}\sum_{\tau=1}^{10} 1} \sum_{\sigma\in\{1,2,\infty\}}\sum_{\tau=1}^{10} \frac{C^{\tau}_{\sigma,\omega}}{\frac{1}{|\Omega(|\omega|)|}\sum_{\varphi\in\Omega(|\omega|)} C^{\tau}_{\sigma,\varphi}} \right)$$

	Reduced Computation Time	Maximized Throughput
1 Bit-rate	$\omega^*(1) := \{m^*\}$	$\omega^+(1) = \{m^*\}$
2 Bit-rates	$\omega^*(2) := \{m^*, m^*-1\}$	$\omega^+(2) = \{m^*, m^*/2\}$
3 Bit-rates	$\omega^*(3) := \{m^*, m^*-1, m^*-2\}$	$\omega^+(3) = \{m^*, m^*-2, m^*-4\}$
4 Bit-rates	$\omega^*(4) := \{m^*, m^*-1, m^*-2, \min\}$	$\omega^+(4) = \{m^*, m^*-2, m^*-4, m^*-6\}$
7 Bit-rates	$\omega^*(5) :=$all supported bit-rates	$\omega^+(5) :=$all supported bit-rates

Table 9.1: Good Performing Sets of Bit-Rates

gives the ratio of the computation time and the ratio of the throughput achieved by ω as compared to the sets of considered bit-rates with the same number of elements. Moreover, (9.13) is averaged over all topologies and all sets of considered transmission powers. Figure 9.2 shows these points when the sets of considered bit-rates have two, three, and four elements. Note that in Figure 9.2, the set $\Omega(i)$ does not include all possible sets with i elements, but only the ones shown in the figure.

Figure 9.2 shows that, to a large extent, all the sets of considered bit-rates with the same number of elements perform similarly. While selecting a set of considered bit-rates depends on the relative importance of maximizing throughput as compared to reducing computation time, Figure 9.2 does provide some guidelines for selecting sets of considered bit-rates. Table 9.1 summarizes the sets of bit-rates that either reduce the computation time or maximize the computed throughput, where the considered bit-rates ω^* tend to reduce the computation time at the expense of the computed throughput, whereas ω^+ tends to maximize the throughput. Since selecting bit-rates according to ω^* can reduce the computation time by up to 20% without having a significant impact of the computed throughput, we focus on the performance achieved when using ω^*.

Figure 9.3 shows the relative increase in the computation time and the computed throughput for different sets of considered bit-rates. Specifically, Figure 9.3 shows $\left(\frac{1}{10}\sum_{\tau=1}^{10} T^{\tau}_{\sigma,\omega^*(i)}/T^{\tau}_{1,\omega^*(1)}, \frac{1}{10}\sum_{\tau=1}^{10} C^{\tau}_{\sigma,\omega^*(i)}/C^{\tau}_{1,\omega^*(1)}\right)$, where the summation

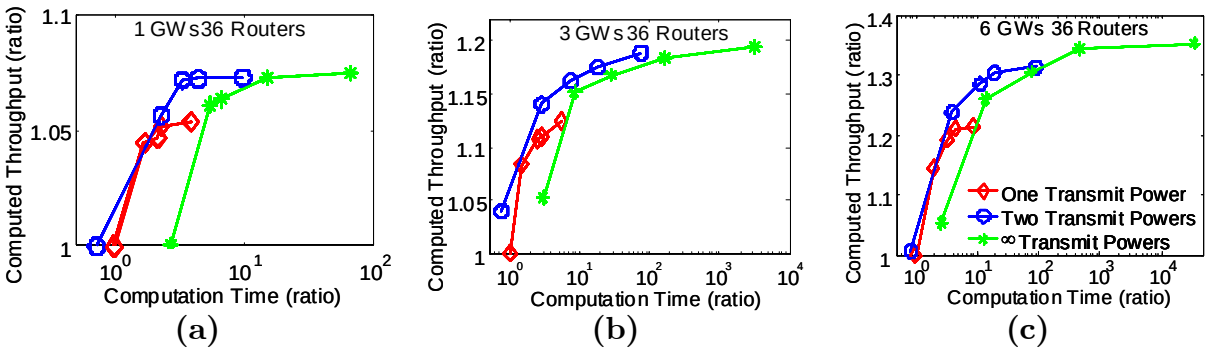


Figure 9.3: Relative increase in the computed throughput and the computation time for various topologies when the bit-rate selection scehemes listed in the column labeled "Reduced Computation Time" are used.

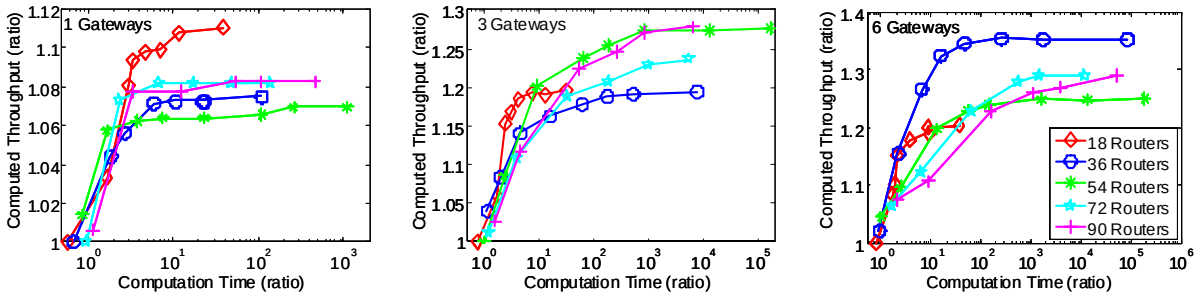
over τ is over all ten topology samples. Figure 9.3 shows the performance for the topologies with 36 wireless routers and with 1, 3, and 6 gateways. The behavior for other topologies is similar. Note that $T^{\tau}_{1,\omega^*(1)}$ and $C^{\tau}_{1,\omega^*(1)}$ are the computation time and the computed throughput when only maximum allowable bit-rate and maximum transmission power are considered. Thus, Figure 9.3 shows the performance relative to the baseline case where a single bit-rate and a single transmission power are used.

Figure 9.3 indicates Pareto optimal sets of bit-rates and transmission powers that increase the computed throughput and the computation time. Specifically, letting $\sigma \mathrm{P}i\mathrm{B}$ represent the case where σ transmission powers are considered and $\omega^*(i)$ is the set of considered bit-rates, Table 9.2 provides a progression of schemes of increasing complexity that result in increasing computed throughput and increasing computation time. Thus, a trade-off between the computation time and the computed throughput is achieved by selecting a stage from this ordering. This progression results in a similar Pareto optimal sequence for the other topologies

Using this progression, Figure 9.4 shows the relationship between $\frac{1}{10}\sum_{\tau=1}^{10} T^{\tau}_{\sigma,\omega^*(i)}/T^{\tau}_{1,\omega^*(1)}$, the average increase in the computation time, and $\frac{1}{10}\sum_{\tau=1}^{10} C^{\tau}_{\sigma,\omega^*(i)}/C^{\tau}_{1,\omega^*(1)}$,

Stage	Scheme	Stage	Scheme
1	2P1B	5	2P4B
2	1P2B	6	2P*All*B
3	2P2B	7	∞P4B
4	2P3B	8	∞P*All*B

Table 9.2: Progression of Sets of Considered Bit-Rates and Transmission Power



(a) (b) (c)

Figure 9.4: The relationship between the computation time and computed throughput for different topologies when the set of considered transmissions powers and bit-rates vary according to Table 9.2 and the bit-rate selection schemes are given by the column labeled "Reduced Computation Time" in Table 9.1.

the average increase in the computed throughput. These metrics are averaged over ten topology samples. It was beyond our computational abilities to compute the optimal schedule for Stages 7 and 8 when there were 72 or 90 nodes in the topology. Thus, Figure 9.4 only shows the first six stages for 72 and 90 wireless routers. A few comments are in order.

- It can be observed that the first few stages provide considerable improvement in throughput, while the last few stages provide little improvement of throughput and require significantly more computation time. Specifically, the improvement in throughput tends to converge as the stage number increases.
  - Convergence is reached or is nearly reached by Stage 4 (two transmission powers and three bit-rates). Specifically, in half of the topologies

examined, using a higher stage than Stage 4 increases the throughput by no more than 2%, and in 95% of all topologies examined, using a higher stage than Stage 4 increases the computed throughput by no more than 6%.

  - The computation time to reach convergence is large. In half of the topologies, Stage 4 increases the computation time by more than a factor of 20, and in 5% of the topologies, Stage 4 increases the computation time by more than a factor of 500.

- Stage 3 also achieves high throughput; in 90% of the topologies, the throughput achieved by Stage 3 is within 10% of the maximum throughput achieved by any stage. In none of the topologies is the maximum throughput achieved by any stage more than 15% larger than the throughput achieved by stage 3. The mean and median of the computation time of Stage 4 was about twice that of Stage 3.

- Considering more than two transmission powers results in only a slight improvement in throughput, but might dramatically increase the computation time (This point is examined in more detail in Section 9.5.4).

- For a fixed number of gateways, the relative improvement in throughput for different numbers of wireless routers is approximately the same. While it is not shown in Figure 9.4, the absolute throughputs for different numbers of wireless routers and the same number of gateways vary by more than a factor of two.

- In many cases, the average Stage 1 gives a higher relative computed throughput that is greater than one and the relative computation time is less than one. Thus, in most cases, 2P1B provides higher throughput and lower computation

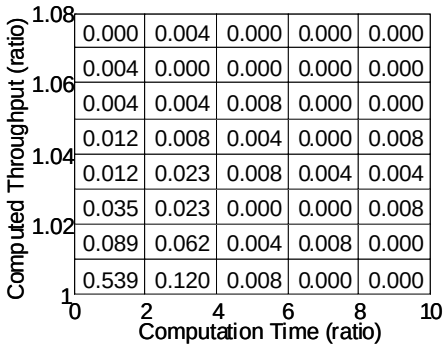


Figure 9.5: The two-dimensional probability distribution of the improvement in the throughput and the increase in the computation time when increasing the number of considered bit-rates from four to seven. Each box covers a range of increases in the throughput and the computation time. The number in each box is the probability of that range of increases occurring.

time than 1P1B. As compared to 1P1B, 2P1B results in a larger optimization problem, that is, since 2P1B has twice as many logical links as 1P1B, (9.4) and (9.8) are larger. However, in the case of 2P1B, Algorithm 11 converges faster, and hence the total computation time is less.

9.5.3 The Need for a Finer Set of Bit-Rates

The set of bit-rates provided by 802.11 a/g are separated by a factor of 1.5 and 1.33. Here we investigate whether it is possible that higher throughput can be achieved by considering a finer set of bit-rates than those supported by 802.11 a/g. To this end, we examine the performance impact of changing from bit-rates that are separated by a factor of 2 to the more finely spaced bit-rate supported by 802.11a/g. For each topology τ and each of the three sets of considered transmission powers σ, the vector $\left(T^{\tau}_{\sigma,\omega^+(5)}/T^{\tau}_{\sigma,\omega^+(4)},\, C^{\tau}_{\sigma,\omega^+(5)}/C^{\tau}_{\sigma,\omega^+(4)}\right)$ is the relative increase in the computation time and the computed throughput when changing from the set of considered bit-rates $\omega^+(4)$ (which is given in Table 9.1) to a set

of all seven bit-rates (i.e., $\omega^+(5)$). Figure 9.5 gives the probability distribution of $\left(T^{\tau}_{\sigma,\omega^+(5)}/T^{\tau}_{\sigma,\omega^+(4)},\, C^{\tau}_{\sigma,\omega^+(5)}/C^{\tau}_{\sigma,\omega^+(4)}\right)$, where the distribution is over one, two, and a continuum of transmission powers and all 150 topologies.

Figure 9.5 indicates that in the typical case, considering the dense set of bit-rates provides little improvement in the throughput as compared to considering only the four sparsely distributed bit-rates. Furthermore, over the 150 topologies and the three transmission power schemes examined, there is only a 5.6% probability that increasing from four to all bit-rates will increase the throughput by more than 4%. Since bit-rates separated by a factor of 1.5 and 1.3 do not result in significantly higher throughput than bit-rates separated by a factor of 2, we conjecture that bit-rates separated finer than those supported by 802.11 a/g will not significantly increase the throughput.

In Section 9.2, (9.1) and (9.3) model the data rate across a link to be a binary function, taking the value 0 or R_{x_m}, which is the nominal data rate over link x with the mth modulation/coding scheme. In practice, ARQ can be used to achieve a continuum of data rates. For example, suppose that in the face of moderate interference, $PSP = 0.5$. With ARQ, the effective data rate is $R_{x_m} PSP$. This behavior is supported by the framework developed in Section 9.2 by employing an alternative "modulation/coding scheme," say m', where $PSP(SINR, m') := 1 - (1 - PSP(SINR, m))^2$ and $R_{x_{m'}} = R_{x_m}/2$. A similar approach can be used to support a wide range of bit-rates between R_{x_m} and 0. However, Figure 9.5 shows that a finer gradation between bit-rates is not needed, and hence there is no need to employ ARQ to achieve intermediate data rates.

9.5.4 The Impact of the Number of Considered Transmission Powers

Figures 9.3 indicates that considering two transmission powers provides a significant advantage over a single transmission power and that considering a continuum of transmission powers provides little improvement in the computed throughput

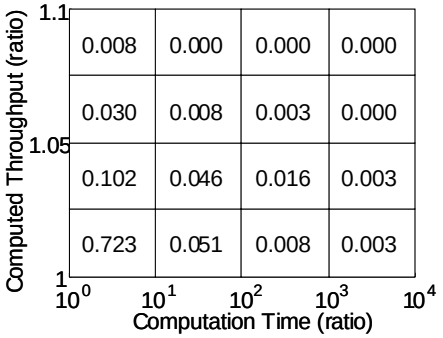


Figure 9.6: The probabilities of improvements in throughput and increases in computation time when switching from considering two transmission powers to a continuum of transmission powers.

and greatly increases the computation time. To examine this second observation in more detail, Figure 9.6 shows the probability of different values of $\left(T^{\tau}_{\infty,\omega}/T^{\tau}_{2,\omega}, C^{\tau}_{\infty,\omega}/C^{\tau}_{2,\omega}\right)$. Note that each topology, τ, and each set of considered bit-rates, ω, results in a different value of $\left(T^{\tau}_{\infty,\omega}/T^{\tau}_{2,\omega}, C^{\tau}_{\infty,\omega}/C^{\tau}_{2,\omega}\right)$. Figure 9.6 shows the probability distribution of $\left(T^{\tau}_{\infty,\omega^*(i)}/T^{\tau}_{2,\omega^*(i)}, C^{\tau}_{\infty,\omega^*(i)}/C^{\tau}_{2,\omega^*(i)}\right)$ over 150 topologies, and over $i = 1, 2, 3, 4,$ and 5 for topologies with 18-54 wireless routers, and for $i = 1, 2, 3,$ and 4 for topologies with 72 and 90 wireless routers. In all, the probability distribution of $\left(T^{\tau}_{\infty,\omega*(i)}/T^{\tau}_{2,\omega^*(i)}, C^{\tau}_{\infty,\omega^*(i)}/C^{\tau}_{2,\omega^*(i)}\right)$ was computed over 690 samples. Thus, Figure 9.6 shows the impact of switching from considering two transmission powers to considering a continuum of transmission powers.

As can be observed, considering a continuum of transmission powers increases the throughput by more than 5% for only 5% of the cases examined. In general, when switching from considering two transmission powers to considering any transmission power, the mean increase in the computed throughput is 1.1% and the median increase in the computed throughput is 0.6%. On the other hand, the mean increase in the computation time is 2100% and the median increase in the computation time is 295%. Since a continuum of transmission powers provides only a small improvement

in the throughput over two transmission powers, considering three or more discrete transmission powers will not significantly increase the throughput. Note that since the computed throughput is optimal for the particular set of considered bit-rates and transmission powers, the conclusion that there is little utility to consider more than two transmission powers holds even if more efficient computation schemes are developed.

9.5.5 The Number of Bit Rates and Transmission Powers Used

Allowing the schedule to use a particular set of considered bit-rates and transmission powers does not imply that each link uses all considered bit-rates and transmission powers. For example, perhaps links with high channel gain will exclusively use low transmission power and links with low channel gain will exclusively use high transmission power. If this was the case, then high computed throughput could be efficiently computed by employing the single "correct" bit-rate and transmission power for each link. This section investigates this possibility.

As discussed in Section 9.2, an optimal schedule is defined by a set of weights $\{\alpha_{\mathbf{v}} | \mathbf{v} \in \mathcal{V}\}$, where $\alpha_{\mathbf{v}}$ is the fraction of time the schedule allocates to assignment $\mathbf{v}$. An assignment specifies which links transmit and at which bit-rate and power. Specifically, $v_{x_{m,s}} = 1$ implies that during assignment $\mathbf{v}$, link x transmits with the mth modulation scheme and with transmission power $p_{x_{m,s}}$, the sth transmission power. Thus, the relative fraction of time that link x uses bit-rate and transmission power (m, s) is

$$F(x, m, s) := \sum_{\mathbf{v} \in \left\{\mathbf{v} | v_{x_{m,s}} = 1\right\}} \alpha_{\mathbf{v}}.$$

For link x, the *most-used bit-rate and transmission power* (MUBRTP) is the $(m^+(x), s^+(x))$ such that $F(x, m^+(x), s^+(x)) \geq F(x, n, t)$ for all n and t. Figure 9.7 (a) shows the average values of $F(x, m^+(x), s^+(x))$ for different sets of considered bit-rates and transmission power. Each point in Figure 9.7 is averaged over all topologies

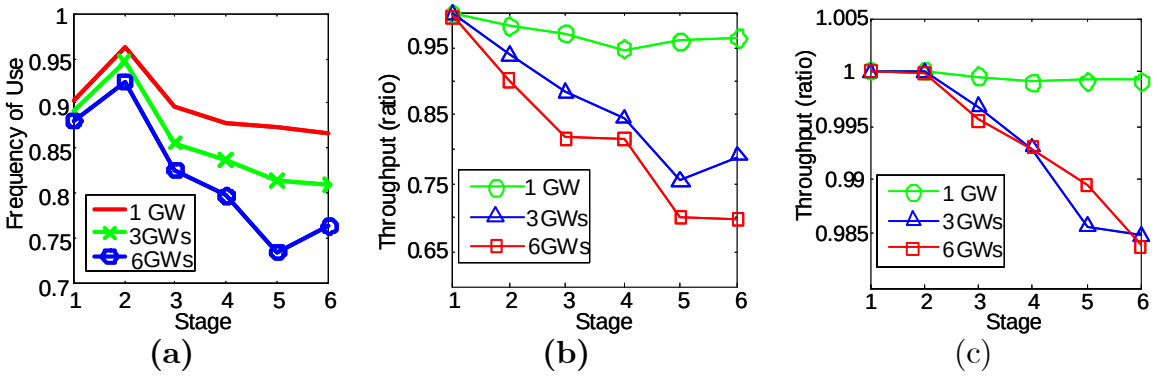


Figure 9.7: (a) The average value of $F(x, m, s)$, the fraction of time that the most used combination of bit-rate and transmission power is used. (b) The ratio of the computed throughput from a 1-quantization and $C^{\tau}_{1,\omega(1)}$. (c) The ratio of the computed throughput from a 2-quantization and $C^{\tau}_{1,\omega(1)}$.

with the indicated number of gateways. As can be observed, the fraction of time that the MUBRP is used is typically quite large. For example, in the case of 1 gateway and Stage 6, 14 different combinations of bit-rates and transmission powers are considered. And yet, on average, each link uses one combination of bit-rate and transmission power 86% of the time.

Now suppose that the MUBRP is known for each link. In this case, a new bit-rate and transmission power scheme can be considered where each link only transmits at its MUBRP. In this way, a set of considered bit-rates and transmission powers induce a particular bit-rate and transmission power scheme. The resulting scheme is referred to as a 1-quantization of the stage, since it results in a single combination of bit-rate and transmission power. A k-quantization uses the k most-used combinations of bit-rates and transmission powers.

Figure 9.7 (b) shows the ratio of the computed throughput with a 1-quantization and the throughput that was achieved by the stage. Figure 9.7 (c) is the same as Figure 9.7 (b), but with a 2-quantization. As can be observed, the throughput under

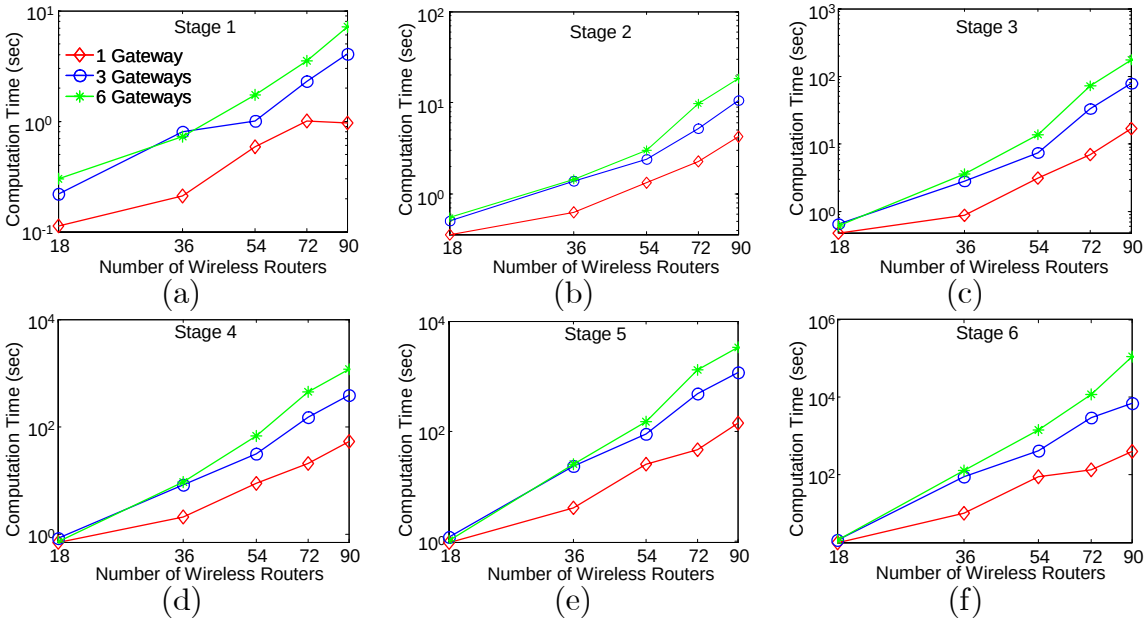


Figure 9.8: Computation time for different stages, where the stages are defined by Table 9.2.

the 1-quantization is quite poor. In fact, in many cases the throughput under the 1-quantization is below the throughput achieved with a single bit-rate and transmission power. On the other hand, the throughput of the 2-quantization is quite good, with throughput within a few percent of what the unquantized stage achieves.

The performance of the 2-quantization motivates a search of two pairs of bit-rates and transmission powers that achieve near optimal throughput. However, despite our efforts, we have been unable to discover an a priori technique to determine these two pairs. Thus, currently, we are only able to find these special pairs after the full computation is complete.

9.5.6 Computation Time Scaling

While the analysis above shows that using multiple powers and multiple bit-rates can improve the computed throughput at the expense of the computation time,

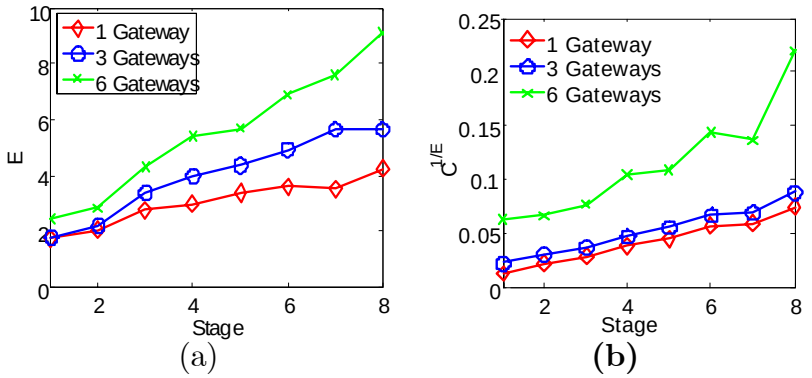


Figure 9.9: The time to compute the optimal schedule modeled at $C \times$ (Number of Wireless Node)^E where the E is shown in (a) and the $C^{1/E}$ is shown in (b). The stages are given in Table 9.2

this section shows the rate that the computation time grows with the size of the network also increases as more transmission powers and bit-rates are considered.

Figure 9.8 shows the average computation time as a function of the number of wireless routers for different numbers of gateways and for the first six stages given by Table 9.2. Note that the relationship is approximately linear on the log-log plot, indicating a polynomial relationship. The quality of the linear approximation is not as good when the computation time is small, e.g., for the first stage and when there are 18 nodes. The reason for this behavior is that since these computation times are so small, they are dominated by such things as loading files and the memory management. Considering networks with 36-90 wireless routers, by taking the log of both sides, least squares can be used to solve for C and E in

$$\text{Computation time} = C \times \text{Number of wireless routers}^E.$$

Figure 9.9 (a) and (b) shows the value of E and C as a function of the stage. As can be observed, the exponent, E, increases as more bit-rates and more transmission powers are considered. This increase is the most significant for networks with a dense set of gateways.

9.6 Conclusion

This work studied the impact of using power control and different bit-rates to increase the computed throughput of a multihop wireless mesh network. As expected, optimizing over more bit-rates and transmission powers increases the computed throughput at the expense of increasing the computation time. A broad conclusion of this work is that complex power control schemes will not greatly impact the throughput and come at the expense of dramatic increase in computation time. More specifically, this work found that it is sufficient to consider two transmission powers and 2 or 3 bit-rates for each link.

Chapter 10

CONCLUSIONS AND FUTURE WORK

10.1 Conclusion

This thesis presented practical techniques for computing optimal schedules in multihop wireless networks even when co-channel interference arises. The critical challenge facing optimal scheduling is the exponential optimization space which makes the Linear or Nonlinear programming problem computationally complex and even intractable. An iterative method to construct a small set of assignments was proposed to reduce the optimization space but achieve the same optimality. As a cost, a set of MWIS problems has to be solved, which has the worst case complexity of NP hard. However, the MWIS problem is solvable in many practical wireless mesh networks. Specifically, by examining over 10000 randomly generated topologies, it was found that the time to compute the MWIS grows polynomially with the number of nodes and linearly with the mean degree of the conflict graph. Moreover, the mean time to solve the MWIS problem for networks with 2048 nodes was approximately one second. The ability to quickly solve MWIS problems allows optimal schedules to be quickly found.

This thesis also explored communication models used in computing optimal throughput. It is found that the traditional protocol models, such as Node Exclusive, 2-hop Node Exclusive and Sensing protocol models have the drawback that they do not accurately model interference. Therefore, the actual throughput provided by these traditional protocol models is poor no matter how good the theoretical

throughput offered. A general SINR protocol model is proposed to more accurately represent the interference. Even if multi-conflicts are ignored, the SINR protocol model exhibits good throughput when applied to a physical model. If we employ techniques to correct multi-conflicts, then the final scheduling is feasible and the actual throughput is no worse than the theoretical one.

The performance improvement provided by the optimal scheduling is significant if there are a large number of gateways. For example, as compared to 802.11's CSMA/CA, optimal scheduling improves performance by a factor between 3 and 11, with the improvement increasing as the density of gateways increases.

As it is possible to compute optimal schedules quickly, this thesis explores joint optimal routing and scheduling. An iterative algorithm for optimal routing is developed along with an approximation of the optimal algorithm. Lagrange multipliers are used as the link cost to find the optimal path. In the networks examined, the approximation yields the same throughput as the optimal algorithm. It is found that in realistic topologies, the proposed algorithm improves throughput by 60% over least hop routing and 20-35% over routing based on max-flow.

Finally, this thesis studied the impact of using power control and different bit-rates to increase the computed throughput of a multihop wireless mesh network. As expected, optimizing over more bit-rates and transmission powers increases the computed throughput at the expense of increasing the computation time. A broad conclusion of this work is that complex power control schemes will not greatly impact the throughput and come at the expense of dramatic increase in computation time. More specifically, this thesis found that it is sufficient to consider two transmission powers and 2 or 3 bit-rates for each link.

10.2 Future Work

10.2.1 Time-Varying Traffic Demands

The traffic demands can be represented with the weights w_ϕ. Thus, when traffic demands change, the w_ϕ change, requiring a new schedule to be generated. Determining a new schedule is a computationally complex task that may generate considerable overhead. Note that there are two optimization problems. One problem finds the optimal schedule for a given set of assignments. A second problem generates new "good" assignments. We consider the second problem first.

10.2.1.1 Robustness of the Set of Considered Assignments

While further work remains, we have found that the set of optimal assignments for one set of w_ϕ provides good performance for a range of w_ϕ. The intuitive reason as to why this is the case is that the good assignments provide high link data rates over a large set of links. Assignments that give high link rates are useful for a wide range of w_ϕ. Furthermore, while Theorem 1 implies that only L assignments are required to be able to compute the optimal schedule, it is, of course, possible to consider a larger set of assignments. In particular, a moderately large set of good assignments can be precomputed such that for a very wide range of w_ϕ, the optimal schedule is within the convex hull of the set of assignments. In this case, when w_ϕ change, there is no need to recompute a new set of assignments. On the other hand, if the number of considered assignments is too large, then it may take considerable computational effort to solve (3.2). Therefore, we will examine the trade-off between computational complexity and the ability to support a wide range of w_ϕ.

Note that since the w_ϕ are related to the traffic demands, it is also important that the set of considered assignments be selected so that they can support the w_ϕ that are reasonably likely to arise. Since mesh networks are long-term deployments, it is reasonable to estimate the traffic demands from network monitors. In fact, this approach is often followed for traffic engineering in wired networks.

10.2.1.2 System Framework for Traffic Adaptation

When the traffic demands change (i.e., the w_ϕ change), a new schedule must be computed. There are two classes of approaches to perform this computation. In one approach, each node only uses information from its neighbors. In this way, this class of approaches is similar to distance vector routing. The other approach is that each router is aware of the network state (specifically, R and w_ϕ) so each router generates the same optimal schedule. This class of approaches is similar to link state routing. The comparison to the link state/distance vector dichotomy is useful. While distance vector routing has the nice feature that only communication between neighbors is required, the convergence can be poor. In link state, overhead is required to distribute the state of links. However, this overhead is viewed as acceptable considering the good performance in terms of convergence time. For this reason, link state routing is typically used in wired networks. Algorithms that solve for schedules with only local information suffer from very slow convergence in comparison to approaches with global knowledge.

It might be practical to occasionally distribute the w_ϕ. Recall that the flow ϕ represents the aggregate of flows from a wired base station to an IN. Each w_ϕ may be a single byte, and can be piggy-backed on transmissions. Thus, periodic or triggered distribution of w_ϕ might result in minimal overhead. Once each router knows the w_ϕ for every flow, each router can determine the optimal schedule. Since all routers have the same information, they will all generate the same result. Note that the w_ϕ are similar to the link state information in link state routing. Thus, in the same way that routers distributed link state information, and then individually compute the paths, here the w_ϕ are distributed and each router computes the schedule. In both cases, the routers perform redundant computation, but the result is good performance in terms of convergence and acceptable overhead.

A key issue of this part of the investigation is how often w_ϕ must be distributed to result in good performance for time-varying traffic demand. On the one hand, the distribution of w_ϕ could be limited to only times when the network administrator is performing manual traffic engineering. On the other hand, one can consider w_ϕ being updated each time a new flow starts. Of course, there is a wide spectrum of schemes between these two ends of the spectrum that need to be investigated. While the overhead of distributing the weights must be considered, the computational load on each router must also be considered.

BIBLIOGRAPHY

[1] Http://www.muniwireless.com/article/articleview/5868/1/23/.

[2] Tropos, "Predictive wireless routing protocol," www.tropos.com/technology/scalability.html.

[3] V. Sridhara, J. Kim, and S. Bohacek, "Performance of urban mesh networks," in *Proc. Of the 8-Th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, 2005.

[4] "From applications to ROI: System architecture for wireless meshes," Farpoint Group, Tech. Rep. FPG 2007-127.1, April 2007.

[5] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of ACM MobiCom*, San Diego, CA, September 2003, pp. 66–80.

[6] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks," in *Proceedings of IEEE INFOCOM*, New York, NY, June 2002, pp. 976–985.

[7] G. Brar, D. M. Blough, and P. Santi, "Computationally efficient scheduling with the physical interference model for throughtput improvment in wireless mesh networks," in *Mobicom*, 2006, pp. 2–13.

[8] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *IEEE INFOCOM*, March 2003.

[9] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, Nov 1998.

[10] S. H. Low and D. E. Lapsley, "Optimization flow control.i: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[11] M. Chiang, "To layer or not to layer: Balancing transport and physical layers in wireless multihop networks," in *IEEE Infocom*, 2004.

[12] Y. Yi and S. Shakkottai, "Hop-by-hop congestion control over a wireless multi-hop network," in *IEEE Infocom*, 2004.

[13] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, March 1972, pp. 85–103.

[14] J. Edmonds, "Paths, trees, and flowers," *Canadian Journal of Mathematics*, vol. 17, pp. 449–467, 1965.

[15] ——, "Maximum matching and a polyhedron with 0,1-vertices," *Journal of Research of the National Bureau of Standards*, vol. 69B, pp. 125–130, 1965.

[16] Y. Cheng, M. Neely, and K. M. Chugg, "Iterative message passing algorithm for bipartite maximum weighted matching," in *2006 IEEE International Symposium on Information Theory*, 2006, pp. 1934–1938.

[17] M. Hanckowiak, M. Karonski, and A. Panconesi, "A faster distributed algorithm for computing maximal matching deterministically," in *Proceedings of PODC 99, the Eighteen Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, 1999, pp. 219–228.

[18] A. Czygrinow, M. Hanckowiak, and P. E. Szymanska, "Distributed algorithm for approximating the maximum matching," *Discrete Applied Mathematics*, vol. 143, pp. 62–71, 2004.

[19] A. Eryilmaz and R. Srikant, "Joint congestion control, routing, and MAC for stability and fairness in wireless networks," *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, vol. 24, no. 8, pp. 1514 – 1524, August 2006.

[20] A. Kashyap, S. Sengupta, R. Bhatia, and M. Kodialam, "Two-phase routing, scheduling and power control for wireless mesh networks with variable traffic," in *SIGMETRIC*, 2007, pp. 85 – 96.

[21] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," in *SIGMETRICS*, 2007, pp. 313 – 324.

[22] S. Sarkar and L. Tassiulas, "End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks," *IEEE Trans. on Automatic Control*, vol. 50, no. 9, pp. 1246–1259, Sep 2005.

[23] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint congestion control and distributed scheduling in multihop wireless networks with a node-exclusive interference model," in *IEEE Infocom*, 2006.

[24] ——, "Joint asynchronous congestion control and distributed scheduling," in *Infocom*, 2006.

[25] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *Information Theory, IEEE Transactions on*, vol. 34, no. 5, pp. 910–917, 1988.

[26] A. Brzenzinski, G. Zussman, and E. Modiano, "Enabling distributed throughput maximization in wireless mesh networks - a partitioning approach," in *Mobicom*, 2006.

[27] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, April 2006.

[28] 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*, vol. 37, no. 12, pp. 1936–1948, 1992.

[29] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput and fairness guarantees through maximal scheduling in wireless networks," in *Proccedings of the Allerton Conference on Communication, Control, and Computing*, 2005.

[30] D. J. Baker, J. Wieselthier, and A. Ephremides, "A distributed algorithm for scheduling the activation of links in a self-organizing mobile radio network," in *IEEE ICC*, 1982, pp. 2F.6.1–2F.6.5.

[31] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Infocom*, 2006.

[32] G. Sharma, R. Mazumdar, and N. Shroff, "On the complexity of scheduling in wireless networks," in *MobiCom*. ACM, 2006.

[33] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "Maximum weighted matching with interference constraints," in *IEEE International Workshop on Foundations and Algorithms For Wireless Networking*, Pisa, Italy, June 2006.

[34] H. Balakrishnan, C. L. Barrett, V. S. A. Kumar, M. V. Marathe, and S. Thite, "The distance-2 matching problem and its relationship to the MAC-layer capacity of ad hoc wireless networks," *IEEE JSAC*, vol. 22, no. 6, pp. 1069–1079, August 2004.

[35] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "Joint congestion control and distributed scheduling for throughput guarantees in wireless networks," in *Infocom*, 2007.

[36] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *MobiCom*. ACM, 2006.

[37] T. Moscibroda and R. Wattenhofer, "Coloring unstructured radio networks," in *Proceedings of the 17th Symposium on Parallel Algorithms and Architectures*, 2005, pp. 39–48.

[38] V. Kumar, M. Marathe, S. Parthasarathy, and A. Srinivasan, "End-to-end packet-scheduling in wireless ad-hoc networks," in *Proceedings of the 15th Symposium on Discrete Algorithms (SODA)*, 2004.

[39] S. Ramanathan and E. L. Lloyd, "Scheduling algorithms for multihop radio networks," *IEEE/ACM Transactions on Networking*, vol. 1, no. 2, pp. 166 – 177, 1993.

[40] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughout optimization in multi-radio wireless mesh networks," in *MobiCom*. ACM, 2005.

[41] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *Infocom*, 2006.

[42] R. Nelson and L. Kleinrock, "Spatial-TDMA: A collison-free multihop channel access protocol," *IEEE Trans. on Communication*, vol. 33, pp. 934–944, 1985.

[43] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 74–85, Jan 2004.

[44] L. Chen, S. H. Low, and J. C. Doyle, "Joint congestion control and media access control design for wireless ad hoc networks," in *Proceedings of IEEE INFOCOM*, March 2005, pp. 2212–2222.

[45] J. Gronkvist and A. Hansson, "Comparison between graph-based and interference-BAsed STDMA scheduling," in *MobiHoc*, 2001.

[46] M. Kodialam and T. Nandagopal, "Characterizing achievable rates in multi-hop wireless mesh networks with orthogonal channels," *IEEE/ACM TRANSACTIONS ON NETWORKING*, vol. 13, no. 4, pp. 868 – 880, August 2005.

[47] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE Journal of Selected Areas on Communications*, vol. 23, pp. 104–116, 2005.

[48] J. Yuan, Z. Li, W. Yu, and B. Li, "A cross-layer optimization framework for multihop multicast in wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2092–2103, Nov 2006.

[49] N. Jin, G. Venkitachalam, and S. Jordan, "Dynamic congestion-based pricing of bandwidth and buffer," *IEEE/ACM Transactions on Networking*, vol. 13, no. 6, pp. 1233 – 1246, Dec 2005.

[50] R. Madan and S. Lall, "Distributed algorithms for maximum lifetime routing in wireless sensor networks," in *IEEE GLOBECOM*, Nov 2004, pp. 748 – 753.

[51] D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, pp. 2608 – 2623, June 2006.

[52] X. Wang and K. Kar, "Cross-layer rate control for end-to-end proportional fairness in wireless networks with random access," in *MobiHoc*, May 2005.

[53] Y. Yi, G. de Veciana, and S. Shakkottai, "On optimal MAC scheduling with physical interference," in *IEEE Infocom*, 2007.

[54] P. Wang and S. Bohacek, "The practical performance of subgradient computational techniques for mesh network utility optimization," in *NET-COOP.*, 2007.

[55] G. Minty, "On maximal independent sets of vertices in claw-free graphs," *Journal of Combinatorial Theory B*, vol. 28, pp. 284–304, 1980.

[56] A. Kako, T. One, T. Hirata, and M. M. Halldorsson, "Approximation algorithms for the weighted independent set problem," available at: http://www.hi.is/ mmh/papers/WIS_WG.pdf.

[57] M. M. Halldórsson, "Approximations of weighted independent set and hereditary subset problems," *Journal of Graph Algorithms and Applications*, vol. 4, no. 1, pp. 1–16, 2000.

[58] L. Tassiulas and A. Ephremides, "Jointly optimal routing and scheduling in packet radio networks," *IEEE Trans. on Info. Theory*, vol. 38, no. 1, pp. 165–168, Jan 1992.

[59] X. Lin and S. Rasool, "A distributed and provably-efficient joint channel-assignment, scheduling and routing algorithm for multi-channel multi-radio wireless mesh networks," Purdue University, Tech. Rep., 2006.

[60] P. Kyasanur and N. H. Vaidya, "Routing and interface assignment in multi-channel multi-interface wireless networks," in *Wireless Communications and Networking Conference*, 2005.

[61] X. Lin and S. Rasool, "A distributed joint channel-assignment, scheduling and routing algorithm for multi-channel ad hoc wireless networks," in *IEEE Infocom*, 2007.

[62] B. Radunovic and J.-Y. L. Boudec, "When power adaptation is useless or harmful," EPFL, Tech. Rep. IC/2004/60, 2004.

[63] P.R.Kumar and P.Gupta, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, March 2000.

[64] L. Massoulie and J. Roberts., "Bandwidth sharing: Objectives and algorithms." *IEEE/ACM Transactions on Networking*, no. 3, June 2002.

[65] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," 2006.

[66] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, January 1997.

[67] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 525–536, August 2003.

[68] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[69] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2003.

[70] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2000.

[71] M. Grotschel, L. Lovasz, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*. Berlin: Springer-Verlag, 1993.

[72] T. Matsui, "Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs," in *JCDCG*, 1998, pp. 194–200.

[73] G. Minty, "On maximal independent sets of vertices in claw-free graphs," *J. Combinatorial Theory*, vol. B, no. 28, pp. 284–304, 1980.

[74] V. Alekseev, "A polynomial algorithm for finding the largest independent sets in fork-free graphs," *Discrete Applied Mathematics*, vol. 135, pp. 3–16, 2004.

[75] G. H. Chen, M. T. Kuo, and J. P. Sheu, "An optimal time algorithm for finding a maximum weight independent set in a tree," *BIT*, vol. 23, pp. 353–356, 1988.

[76] R. Karp and M. Sipser, "Maximum matchings in sparse random graphs," in *FOCS*, 1981.

[77] G. Valiente, *A New Simple Algorithm for the Maximum-Weight Independent Set Problem on Circle Graphs*. Springer, 2003, vol. 2906, pp. 129–137.

[78] M. Fürer and S. P. Kasiviswanathan, "Algorithms for counting 2-SAT solutions and colorings with applications," *Electronic Colloquium on Computational Complexity (ECCC)*, no. 033, 2005.

[79] F. V. Fomin, S. Gaspers, and S. Saurabh, "Branching and treewidth based exact algorithms," in *ISAAC*, 2006, pp. 16–25.

[80] L. Babel, "A fast algorithm for the maximum weight clique problem," *Computing*, vol. 52, pp. 31–38, 1994.

[81] E. Balas and J. Xue, "Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring," *Algorithmica*, vol. 15, no. 5, pp. 397–412, 1996.

[82] ——, "Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs," *SIAM J. Comput.*, vol. 20, no. 2, pp. 209–221, 1991.

[83] J. S. Warren and I. V. Hicks, "Combinatorial branch-and-bound for the maximum weight independent set problem," 2007.

[84] P. R. J. Östergård, "A fast algorithm for the maximum clique problem," *Discrete Applied Mathematics*, vol. 120, no. 1-3, pp. 197 – 207, August 2002.

[85] M. DEMANGE and V. PASCHOS, *Exact and Approximation Results on Maximum Independent Set and Minimum Vertex Coveringgraphs with Great Stability Number*. Universit Paris-Dauphine: Cahier du LAMSADE 128, 1995a.

[86] M. M. Halldórsson, *Approximation Algorithms for Combinatiorial Optimization*. Springer Berlin / Heidelberg, 2004, ch. Approximations of Independent Sets in Graphs, pp. 24–45.

[87] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithms for the maximum weighted independent set problem," *Discrete Applied Mathematics*, vol. 126, pp. 313–322, 2003.

[88] ILOG, "CPLEX," http://www.ilog.com/products/cplex/.

[89] S. Bohacek and P. Wang, "Toward tractable computation of the capacity multihop wireless networks. [implementation]," available at: http://udelmodels.eecis.udel.edu/lumnets1.php.

[90] S. Bohacek, V. Sridhara, and J. Kim, "UDel Models," available at: http://udelmodels.eecis.udel.edu/.

[91] V. Sridhara and S. Bohacek, "Realistic propagation simulation of urban mesh networks," *The International Journal of Computer and Telecommunications Networking Computer Networks and ISDN Systems (COMNET)*, 2007.

[92] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap.* Chapman and Hall/CRC, 1994.

[93] J. Zhu, X. Guo, L. Yang, and W. S. Conner, "Leveraging spatial reuse in 802.11 mesh networks with enhanced physical carrier sensing," in *Proc. Of IEEE ICC*, 2004.

[94] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum, 1972, pp. 85–103.

[95] J. Hastad, "Clique is hard to approximate within $n^{1-\epsilon}$." in *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, 1996, pp. 627–636.

[96] V. E. Alekseev and V. V. Lozin, "Augmenting graphs for independent sets," Rutgers Center of Operations Research, Tech. Rep. 59-2000, 2000.

[97] V. Klee and G. J. Minty, "How good is the simplex algorithm," in *Inequalities III*, O. Shisha, Ed. New York: Academic Press, 1972, pp. 159–175.

[98] Dash Optimization, "Xpress optimizer," 2007.

[99] V. Damerow, "Average and smoothed complexity of geometric structures," Ph.D. dissertation, Universitat at Paderborn, 2005.

[100] B. Bollobas, T. I. Fenner, and A. M. Frieze, "An algorithm for finding hamilton cycles in a random graph," in *Proceedings of the 17th Annual ACM Symposium on Theory of Computing.* ACM, 1985, pp. 430–439.

[101] L. Perkovic, "Edge coloring, polyhedra and probability," Ph.D. dissertation, Carnegie Mellon University, 1998.

[102] H. S. Wilf, *Algortihms and Complexity*. AK Peters, Ltd, 2002, no. available at: http://www.physics.it/lectures/AlgorithmComplexity.pdf.

[103] A. Goldberg, "On the complexity of the satisfiability problem," New York University, Tech. Rep. Courant Computer Science Report 16, 1979.

[104] A. Goldberg, P. W. Purdom, and C. A. Brown, "Average time analysis of simplified davis-putnam procedures," *Information Processing Letters*, vol. 15, pp. 72–75, 1982.

[105] J. N. Hooker, "Resolution vs. cutting plane solution of interference problems: Some computational experience," *Operations Research Letters*, vol. 7, pp. 1–7, 1988.

[106] A. P. Kamath, N. K. Karmarker, K. G. Ramakrishnan, and M. G. C. Resende, "Computational experience with an interior point algorithm on the satisfiability problem," in *Proceedings of the Conference on Integer Programming and Combinatorial Optimization*. Mathematical Programming Society, 1990, pp. 333–349.

[107] D. G. Mitchell, B. Selman, and H. J. Levesque, "Hard and easy distributions for SAT problems," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, P. Rosenbloom and P. Szolovits, Eds. Menlo Park, California: AAAI Press, 1992, pp. 459–465. [Online]. Available: citeseer.ist.psu.edu/mitchell92hard.html

[108] E. Arikan, "Some complexity results about packet radio networks," *IEEE Trans. on Info. Theory*, vol. 30, no. 4, pp. 681–685, Jul 1984.

[109] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Cross-layer optimization in TCP/IP networks," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 582–568, 2005.

[110] S. Bohacek and P. Wang, "Toward tractable computation of the capacity of multihop wireless networks," in *Infocom*, 2007.

[111] M. Haenggi and D. Puccinelli, "Routing in ad hoc networks: a case for long hops," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 93–101, Oct. 2005.

[112] S.-J. Lee and M. Gerla, "Aodv-br: backup routing in ad hoc networks," *Wireless Communications and Networking Conference, 2000. WCNC. 2000 IEEE*, vol. 3, pp. 1311–1316 vol.3, 2000.

[113] A. Nasipuri and S. Das, "On-demand multipath routing for mobile ad hoc networks," *Computer Communications and Networks, 1999. Proceedings. Eight International Conference on*, pp. 64–70, 1999.

[114] Y. Liu, X. Hu, M. Lee, and T. Saadawi, "A region-based routing protocol for wireless mobile ad hoc networks," *IEEE Network Magazine*, vol. 18, 2004.

[115] S. Bohacek, J. P. Hespanha, J. Lee, C. Lim, and K. Obraczka, "TCP-PR: TCP for persistent packet reordering," *IEEE/ACM Transactions on Networking*, 2006.

[116] K. C. Kapur, "On cutoff optimization methods in infinite-dimensional spaces and applications," *JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS*, vol. 12, no. 1, pp. 16–31, 1973.

[117] D. M. Topkis, "A note on cutting-plane methods without nested constraint sets," *Operations Research*, vol. 18, no. 6, pp. 1216–1220, 1970.

[118] T. Rappaport, *Wireless Communications - Principles and Practice*. Prentice Hall, 2002.