PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis Acceptance

This is to certify that the thesis prepared

By Xiaojun Lin

Entitled

Simplification of Network Dynamics in Large Systems

Complies with University regulations and meets the standards of the Graduate School for originality and quality

For the degree of Doctor of Philosophy

Final examining committee members

N. B. Shroff , Chair

E. J. Coyle

R. Mazumdar

S. Fahmy

Approved by Major Professor(s): N. B. Shroff

Approved by Head of Graduate Program: V. Balakrishnan

Date of Graduate Program Head's Approval: 7/13/05

SIMPLIFICATION OF NETWORK DYNAMICS IN LARGE SYSTEMS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Xiaojun Lin

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2005

To my parents Xianba Lin and Aiju Shao, and my wife Lining Li.

ACKNOWLEDGMENTS

I would like to first and foremost thank my advisor, Professor Ness B. Shroff, for his contant guidance, support and encouragement during my entire Ph.D. years. I am priviledged to have such a wonderful advisor, who is at all times enthusiastic, optimistic, patient, helpful and encouraging. He gave me countless advice and insights during the course of my work, without which completing this thesis would have been much more difficult. He also patiently taught me how to write good papers and give impressive presentations, which are skills I will forever indebted to him.

I learned a lot of the theory of stochastic processes from Professor Ravi R. Mazumdar, who also served on my doctoral committee. His mastery of advanced mathematical tools is exemplary, and it has always been so nice to have him to consult with when I encountered difficult analytical problems. I am also priviledged to have Professor Edward J. Coyle and Professor Sonia Fahmy in my doctoral committee. I benefited from them not only through classes that I took, but also through many stimulating discussions over the years. Thanks also to Professor Kihong Park for serving on my doctoral committee during the early stage of my Ph.D. I am also grateful to Professor Edwin K. P. Chong, who later left Purdue and went to Colorado State University, and Professor R. Srikant at University of Illinois Urbana-Champaign for their many help and advice.

I had a wonderful summer internship at IBM T.J. Watson Research Labs in Summer 2002. I got to know many outstanding researchers and interesting problems when I was there. I want to extend my deepest gratitude to Dr. Jakka Sairamesh and Dr. Rakesh Mohan, who have accommodated me and provided me with guidance and help during and after my internship.

I would like to thank my current and former office-mates and friends that I met at Purdue. I am particularly grateful to Do Young Eun for his enthusiastic discussions

during the many years when he was here and after he became a faculty at North Carolina State University. I also want to thank Han S. Kim, Xin Liu, Mingbo Xiao, Dongyu Qiu, Jang-won Lee, Longbi Lin, Sungoh Kwon, Sarah Sellke, Gaurav Sharma, Qingjian Tian, Yu Ying, Le Cai, Vivek Mhatre, Sunil Kulkarni, Jin Zhang, Peilu Ding, Krishnakamal Sayana, Junshan Zhang, Brad and Alisha Sickler, Tom and Lori Summer and family, and many more. I thank you all for supporting me over the year and for making my life at Purdue the most enjoyable experience.

I want to thank my parents who have always loved me and encouraged me in my study. Special thanks also go to my parents-in-law for their selfless love and support, especially during the time when my daughter was born. Finally, I want to thank my dear wife Lining. This thesis would not be possible without her endless love.

TABLE OF CONTENTS

Page

Page

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Lin, Xiaojun. Ph.D., Purdue University, August, 2005. Simplification of Network Dynamics in Large Systems. Major Professor: Ness B. Shroff.

Controlling today's communication networks is a challenging task due to the tremendous growth both in terms of the network capacity and in the number of elements (e.g., end-users, routers and hosts) that the network supports. As the size of the network grows, the network dynamics also become increasingly complex. Solutions that once work well for small networks may no longer be appropriate for large-scale and high-bandwidth networks.

In this dissertation, we have taken two orthogonal approaches to simplify the network dynamics in large communication systems. In the first approach, we seek simplicity through exploiting the largeness of the network. We first study the pricing-based control problem and the Quality-of-Service routing problem in large-capacity wire-line networks. We show that simple static control policies can approach the performance of the optimal (but complex) dynamic control policy when the capacity of the system is large. We develop simple and distributed control algorithms based on these static control policies. Our control solution can significantly reduce the computation complexity and communication overhead without sacrificing the network performance. We then turn to wireless networks and investigate the fundamental tradeoff between the capacity and the delay in large mobile wireless networks. By exploiting the largeness in the number of nodes in these networks, we obtain simple scaling laws that determine the optimal achievable capacity given delay constraints.

In the second approach, we seek simplicity by designing an appropriate control architecture such that complex interactions within the system can be structured into layers that are only weakly dependent on each other through a judiciously chosen set

of control parameters. In particular, we investigate the cross-layer congestion control and scheduling problem in multi-hop wireless networks. We develop a loose-coupling approach to this problem, where the cross-layer solution only requires a minimal amount of interaction between the layers, and is robust to imperfect decisions at each layer. This result allows us to use imperfect, but simpler and potentially distributed, algorithms for cross-layer control of large wireless networks. We have successfully developed such a fully distributed cross-layer control solution for certain interference model.

1. INTRODUCTION

Today's communication networks have seen tremendous growth both in terms of the network capacity and in the number of elements (e.g., end-users, routers and hosts) that the network supports. Fiber-optic links in the Internet backbone can now operate at multi-Gigabits-per-second speeds, and can support thousands of users simultaneously. For the wireless domain, it is expected that multihop wireless networks will consist of hundreds of nodes (e.g., sensor networks or mesh networks). As the size of the networks continues to grow, it is becoming increasingly difficult to efficiently control the network resources and to satisfy the diverse service requirements of the various users. Solutions that once worked well for small networks may no longer be appropriate for large-scale and high-bandwidth networks. For example, while the optimal control problem for a small system can be formulated and solved via Dynamic Programming techniques for Markov Decision Processes (MDP) [1], it is common wisdom that the complexity of such an approach will grow exponentially as the size of the problem increases. Due to this "state explosion" problem, MDP approaches are generally viewed as computationally infeasible for large systems.

Given the sheer size of these communication networks and the infeasibility of traditional control solutions, there is a pressing need to carefully understand how to design and control these large-scale systems effectively. In this dissertation, our main objective is to understand how to model these large networks and to develop simple, efficient and scalable control solutions. Towards this end, we adopt two orthogonal approaches to simplify the network dynamics in large systems. In the first approach, we show that while largeness is a source of *complexity*, it can also be exploited to *simplify* network control. In the second approach, we show that simple and efficient control solutions can also be obtained by carefully designing the control architecture,

such that complex interactions within the system can be structured into layers that are only weakly dependent on each other through a judiciously chosen set of control parameters. Next, we will give an overview of both of these two approaches.

1.1 Exploiting the Largeness of the System to Simplify Control

The idea that largeness can also become a source of simplicity is not new. In fact, we use this idea frequently in our daily lives. Imagine that one needs to pack several pieces of luggage into the trunk of a car before driving for a long trip. A careful choice of the position and direction of each bag is often required to fill the car trunk with the maximum number of bags. Note that the number of bags that the car trunk can hold is usually not very large. Now imagine that the size of the car trunk grows very large, while the size of each piece of luggage remains the same. Then, compared with the huge car trunk, each piece of luggage looks like a sand particle compared with a jar. The packing problem suddenly becomes equivalent to filling a jar with sand. For the latter problem, we could simply pour the sand into the jar, without worrying about how to optimally adjust the position and direction of each sand particle. Our intuition tells us that even if an optimal positioning of each sand particle exists, the additional gain would be minimal. As we have seen, the largeness of the system has allowed us to use simple methods to easily achieve acceptable performance.

To a certain extent, results such as the Law of Large Numbers and the Central Limit Theorem, which all deal with aspects of largeness, have been the corner-stones of probability theory and engineering science. In the past, largeness has also been exploited extensively in network analysis. For example, in analyzing the buffer occupancy distribution of a single queue, researchers have used Large Deviation theory to obtain simple approximations when either the buffer length goes to infinity or when the capacity of the queue, the demand, and the buffer length are all increased proportionally to infinity (see [2–5] and the reference therein). A different approach

is to invoke the Central Limit Theorem and to approximate the aggregate traffic arrivals of a large number of connections as Gaussian. Simple and accurate approximations for the buffer overflow probability have been derived [6–9]. Although these approximations are obtained under the assumption that the number of connections is large, in practice they have been shown to be accurate even for moderate systems. Further, when one attempts to analyze the end-to-end Quality of Service for a network of queues, it has been shown that the traffic of a particular flow remains essentially unchanged when it traverses a node in which a large number of traffic flows are multiplexed [10, 11]. Hence, when the capacity of certain nodes is large, as long as their utilization is less than one, they may be ignored from the analysis and their "deletion" does not appreciably affect the overall accuracy of the analysis. Thus a network of queues can be decomposed and analyzed in isolation, a much simpler task than analyzing the entire network.

In this dissertation, we will focus on the network control problem, and we will show in Chapters 2-5 how largeness can be exploited to simplify the control mechanism in large communication networks. Note that the control problem is different from the analysis problem we just mentioned due to its closed-loop nature. Along this line, we will investigate a number of networking problems in both wire-line and wireless systems, including pricing-based network control, Quality-of-Service (QoS) routing, and the capacity-delay tradeoff in mobile wireless networks.

1.1.1 Pricing-Based Network Control

We first focus on wire-line networks where the capacity of the network is typically very large. To illustrate the type of simplicity that arises in large-capacity networks, we model the network control problem as a pricing problem. The price affects the users' interest to join the network (e.g., the arrival rate of the user is a function of the price). The network's objective is to maximize some overall revenue or utility by appropriately choosing the price. Such a framework has received significant interest

in the literature (e.g., see [12–16] and the references therein) wherein price provides a good control signal because it carries monetary incentives. The network can then use the current price of a resource as a feedback signal to coerce the users into modifying their actions (e.g., changing the rate or route).

In order to maximize the performance of the system, a network provider can choose between a *dynamic pricing scheme* and a *static pricing scheme*. A *dynamic pricing scheme* is one where the network provider can charge different prices to the user according to varying levels of congestion in the network. For example, the network provider can charge a higher price when the network is more congested, and charge a lower price (to attract new customers) when the network is less congested. A dynamic pricing scheme can be made more efficient (and complex) by taking into account the past history and the parameters of the individual user. For example, the network can use the past history to *predict* future congestion levels up to a certain time window, and use the prediction to make pricing decisions. Similarly, if the network provider has prior information about the parameters of individual users (for example, the duration that each user is going to stay in the network), it can charge an appropriate price, favoring one type of user over the other.

Clearly, the more information a dynamic pricing scheme can depend on, the higher the performance it can potentially achieve. However, this performance improvement comes at a cost. The optimal dynamic pricing scheme is usually very difficult, if not impossible, to obtain. Under Markovian assumption, one may use Dynamic Programming [1] to find the optimal pricing scheme. The complexity of Dynamic Programming will grow exponentially as the size of the problem increases (commonly referred to as the "curse of dimensionality"). Without Markovian assumptions, there are few techniques to solve the optimal pricing scheme for large-size problems.

A dynamic pricing scheme is also difficult to implement in practice. The dynamic scheme requires information about each instantaneous state. This not only incurs significant communication overhead (in order to collect each state information con-

tinuously), but also becomes infeasible due to the presense of network delay. When the dynamics of the system evolve faster than network delay, the information that a dynamic scheme could be based on is usually outdated. Hence the optimality of a dynamic pricing scheme could be compromised.

Interestingly[1], it turns out that when the capacity of the network is large, *simple static pricing schemes*, where the price is constant or changes relatively slowly over time, can asymptotically achieve the same revenue as the *optimal dynamic pricing scheme*. Further, the near-optimal static price can be determined by solving a *simple* non-linear programming problem that only depends on the average statistics of the network. In Chapter 2, we will establish these types of results under general network settings, first for a non-Markovian network with fixed topology and routing, then in the case when the network supports dynamic routing, and also in the case where the topology of the network becomes increasingly complex as its capacity grows. To illustrate how well simple static pricing schemes can typically perform, consider the following special case. Let the capacity of the network and the demand be scaled proportionally by a factor of c, and let $J^{*,c}$, J_0^c be the performance of the optimal dynamic scheme and the appropriate static scheme, respectively, in the c-scaled system. We will show in Chapter 2 that, as $c \to \infty$,

$$\lim_{c\to\infty} \frac{J^{*,c} - J_0^c}{J^{*,c}} = 0. \tag{1.1}$$

The above result suggests that we can use simple static schemes to control large communication networks [18–20]. Note that these static schemes can be much easier to compute and implement than dynamic schemes. These static schemes compute prices based on the average levels of congestion in the network (and hence do not require information about the current instantaneous levels of congestion). They have lower computation and communication overhead, and are insensitive to network delay. Hence, the above result indicates that significant simplicity in control can indeed be achieved in large networks.

[1]First pointed out by Paschalidis and Tsitsiklis [17]

1.1.2 Distributed Algorithms and Quality-of-Service Routing

For large networks, it is also imperative that the control algorithms that are developed can be implemented on-line and in a distributive fashion. Although the result we just stated indicates that some form of simple control suffices for large networks, it still requires solving a global optimization problem in order to obtain the control parameters (such as the static price). In Chapters 3 and 4, we will investigate how to develop distributed control algorithms for practical networking problems based on simple static controls. In particular, we will develop such distributed algorithms for the Quality-of-Service (QoS) routing problem in high-capacity networks. We will first show that simple proportional routing schemes (that are similar in spirit to the static pricing schemes in Chapter 2) are asymptotically optimal when the capacity of the network is large. We will then develop a fully-distributed algorithm for computing the parameters of the proportional routing scheme, rigorously establish the convergence of the algorithm to the optimal control parameters, and provide guidelines on how to choose the parameters of the algorithm to ensure efficient control. As a result, our proposed algorithm can not only achieve near-optimal routing performance, but also substantially alleviate the computation and communication overhead of QoS routing without sacrificing the performance [21]. Finally, this class of algorithms can not only be used for QoS routing, but can also be applied to a number of other networking problems, including network pricing and multi-path flow control [22].

1.1.3 Capacity-Delay Tradeoff in Large Mobile Wireless Networks

In the first two parts of the dissertation (i.e., Chapters 2-4), we have focused on largeness in terms of the network capacity. This type of largeness is typical in wireline networks. In the third part of the dissertation, we turn to simplicity in large wireless networks. However, the capacity of a wireless network is typically not very large. Hence, the approaches in Chapters 2 and 3 do not directly apply. Instead, we

can exploit other dimensions of largeness. Note, that in wireless ad hoc and sensor networks, the number of nodes can be quite large. In Chapter 5, we will study how this type of largeness can also lead to simple and critical insights in control. In particular, we will exploit this type of largeness to obtain simple relationships that characterize the fundamental tradeoff between the capacity and the delay in large mobile wireless networks. Note that although it is well known that mobility can improve network capacity [23], the questions that we attempt to answer in Chapter 5 are: What is the maximum capacity under a given delay constraint, and how can we design control solutions to achieve this maximum capacity. Finding the exact relationship for this tradeoff is difficult due to the complex interactions within the system. However, much simpler asymptotic relationships can be obtained in the asymptotic limit when the number of nodes is large. Although these asymptotic relationships are in the form of scaling laws (e.g., $\Theta(1/\sqrt{n})$) and they are only accurate up to the order, they still provide critical insights regarding the inherent performance limits of the system, and how to achieve the optimal performance. In Chapter 5, we will develop a systematic methodology both for finding the optimal capacity-delay tradeoff and for designing the capacity-achieving scheme. Our methodology can be applied to a number of mobility models, such as the *i.i.d.* mobility model, the random way-point mobility model, and the Brownian motion mobility model [24–27]. In each case, we have identified the limitations of existing works, obtained sharper results under more general settings, and provided new insights on the fundamental capacity-delay tradeoffs. In particular, under the *i.i.d.* mobility model, our study allows us to develop a scheme that can exploit mobility and achieve a provably larger per-node capacity than that of the static networks *even with delay that does not grow with the number of nodes*. This is the first such result of its kind in the literature. Our methodology can also be extended to incorporate additional scheduling constraints in the system, and be used to find optimal scheduling schemes in a variety of network settings.

1.2 Designing Appropriate System Architecture to Simplify Control

In Chapters 2-5, we show how one can obtain simple and efficient control solutions for large networks by exploiting the largeness of the system. The accuracy of such an approach varies according to the particular problem. For example, for wire-line networks where the capacity is large, we can obtain simple control solutions that are asymptotically exact in the sense that the gap between the simple solution and the optimal solution goes to zero as the scale of the system increases to infinity (see (1.1)). However, for wireless networks where the number of nodes is large, our approach that exploits largeness only allows us to obtain results that are order-accurate (e.g., $\Theta(1/n)$ versus $\Theta(1/n^2)$, see Chapter 5). Due to the scarcity of network resources in wireless systems, the constants before the order terms are often important in many scenarios (e.g., $1/n$ versus $10/n$). Thus, other approaches are required. In the fourth part of the dissertation, we will demonstrate that simplification of network control can also be obtained through appropriately designing the control architecture.

Traditionally, communication networks have been engineered according to the layered architecture [28]. The functionality of the network is divided into layers. For example, the OSI reference model has seven layers: the physical layer, the data link layer, the network layer, the transport layer, the session layer, the presentation layer, and the application layer. The layering concept essentially treats each layer as a *black box*: the higher layer only needs to know the interface to the lower layer, but not the details of how the interface is implemented. Clearly, the layered architecture provides *modularity*, which contributes to *simplicity* and *scalability* of the entire system. Thus, the layered architecture has been a key contributing factor to the success of many network systems, including the Internet.

While such a layered approach has been very successful for wire-line networks, it has turned out to be increasingly inadequate for wireless networks. In wireless networks, there exists a natural coupling between different layers. For example, the choice of the transmission scheme at the physical and MAC layer will affect the

capacity of each link, which in turn will affect the congestion control decision at the transport layer. Due to such coupling, it has become increasingly clear that merely optimizing within layers is insufficient to obtain the orders-of-magnitude performance gains necessary to fuel major growth in next-generation wireless services. To achieve these performance gains, it is imperative that network protocols and designs are engineered by optimizing across the layers (cross-layer design). The idea is that by jointly optimizing the control over two or more layers, cross-layer solutions can yield significantly improved performance by exploiting the tight coupling between the layers.

However, the Achilles heel of cross-layer design is its potential to destroy *modularity*. In cross-layer solutions, although the performance of the system can be optimized, the tight interaction between the layers could make the overall system highly sensitive to changes in each layer. Thus, there is a fundamental tradeoff between efficiency and modularity that needs to be carefully taken into account in any cross-layer solution.

In Chapter 6, we propose a *loose-coupling* approach to cross-layer design in wireless networks. By *loose-coupling*, we mean that the cross-layer solution should only require a minimal amount of interaction across layers and should be robust to imperfect information or imperfect actions taken at various layers. In other words, the complex cross-layer interactions in the system will then be structured into layers that are only *weakly* dependent on each other through a judiciously chosen set of control parameters. Clearly, such a loose-coupling approach allows us to have both *efficiency* and *modularity*. In Chapter 6, we will demonstrate how such a *loose-coupling* solution can be obtained for the cross-layer congestion control and scheduling problem in multihop wireless networks. We will formulate in Chapter 6 a cross-layer control problem that jointly allocates the end-to-end data rate to each user (i.e., the congestion-control problem) and computes the schedule and power/rate assignment for each link (which we will referred to as the scheduling problem). We will show that, in the optimal solution, the congestion-control component and the scheduling

component can be decomposed: they both act independently on the queue lengths of the system. The two components are then coupled by the update of the queue length at each link. Further, if one replaces the scheduling component by an imperfect scheduling policy that only computes suboptimal schedules at each time, we can still quantify the impact on the overall system performance fairly easily for a large class of imperfect scheduling policies. Our cross-layer solution results in provably better performance than a layered approach, even when imperfect scheduling is used.

These results provide us with a framework to design simple and efficient cross-layer control solutions for multihop wireless networks [29,30]. One may even be able to develop fully distributed cross-layer solutions that can be implemented in large networks. In fact, we have successfully developed such a fully distributed cross-layer congestion control and scheduling algorithm under a particular interference model.

The rest of the dissertation is organized as follows: In Chapter 2, we study the pricing problems and derive asymptotic optimality of static pricing schemes when the capacity of the system is large. In Chapter 3, we turn our attention to the problems of Quality of Service routing, and study how static scheme can reduce the computation and communication costs inherent in many QoS routing problems. We will present here a distributed algorithm that can be used to derive the parameters of the static scheme in an adaptive fashion based on on-line measurements. We study the convergence properties of this algorithm in Chapter 4. In Chapter 5, we study a different type of largeness, i.e., when the number of nodes in the system is large, and we establish the fundamental capacity-delay tradeoff of mobile wireless networks. In Chapter 6, we turn to simplicity that can be obtained from appropriately designing the control architecture, and we propose a loose-coupling approach to cross-layer design in multihop wireless networks. We then conclude in Chapter 7.

2. SIMPLIFICATION OF PRICING-BASED CONTROL IN LARGE NETWORKS

2.1 Introduction

In this chapter, we study the simplification of pricing-based control in large-capacity networks. We use pricing as the network control mechanism for achieving certain performance objectives, and the performance objectives can be modeled by some revenue- or utility-functions. Such a framework has received significant interest in the literature (e.g., see [12–16] and the references therein). Prices are good control signals because they are simple and effective. Prices carry clear monetary incentives that users can easily appreciate. The network can use the current price of a resource as a feedback signal to coerce the users into modifying their actions (e.g., changing the rate or route). Through utility functions, we can model the cost-sensitivity of a variety of users, some of which are more cost-conscious than others. Prices also provide clear incentive for the network (i.e., service provider) to optimize its efficiency. Finally, in a real system "prices" do not have to represent real money. A generic control signal where all users agree upon can also take the place of the "price." Hence, our results in this chapter can be generalized to those scenarios as well.

In [17], Paschalidis and Tsitsiklis have shown that when the number of users and the network capacity become very large, the performance (in terms of expected revenue or welfare) of an appropriately chosen *static pricing scheme* approaches the performance of the optimal *dynamic pricing scheme*. This elegant result is an example of the type of *simplicity* that one can obtain when the system becomes large. Note that a *dynamic pricing scheme* is one where the network provider can charge

different prices to the user according to varying levels of congestion in the network, while a *static pricing scheme* is one where the price only depends on the average levels of congestion in the network (and is hence invariant to the instantaneous levels of congestion). Static schemes are usually much easier to compute and implement than dynamic schemes. The result in [17] is obtained under the assumption of Poisson flow arrivals, exponential flow holding times, and a single resource (single link). In this chapter, we will show that simple static network control can also approach the optimal dynamic network control under more general assumptions and a variety of other network problems.

2.1.1 Summary of Contributions

For simplicity of exposition, we structure this chapter as follows. In Section 2.2, we extend the result of [17] from the single-link case to a general loss network with arbitrary holding time distributions[1]. Our technical contributions are three-fold:

1) We generalize the results of [17] and [31] to *non-exponential* holding time distributions. Note that while the assumption of Poisson arrivals for flows in the network is usually considered reasonable, the assumption of exponential holding time distribution is not. For example, much of the traffic generated on the Internet is expected to occur from large file transfers which do not conform to exponential modeling. By weakening the exponential holding time assumption we can extend our results to more realistic systems. We show that a static pricing scheme is still asymptotically optimal, and that the correct static price depends on the holding time distribution only through its mean. A nice observation that stems from this result is that under certain conditions, the static price depends only on the price-elasticity of the user, and not on the specific route or distance. This indicates, for example,

[1]Independently of our work that was first reported in [18], Paschalidis and Liu have also extended the work in [17] from a single-link case to a network case [31]. However, their result still uses the exponential holding-time assumption.

that the flat pricing scheme used in the domestic long distance telephone service in the U.S. may be an economically good pricing mechanism.

2) We also investigate whether more sophisticated schemes can improve network performance (e.g., schemes that have prior knowledge of the duration of individual flows, schemes that predict future congestion levels, etc.). We find that the performance gains using such schemes become increasingly marginal as the system size grows.

3) We study two types of scaling to model large networks. We study the original scaling in [17] and [31], which requires that the number of users between each source-destination pair is scaled proportionally with the capacity of the network. We also study a more general type of scaling that is suitable even when the capacity of the network is large but when the number of users between each source-destination pair is small. Our new scaling is more appropriate for modeling certain Internet scenarios where the topology is complex and the routing is diverse. We show that appropriate static schemes are asymptotically optimal under both scaling models.

We then weaken the assumption on fixed routing and study a dynamic routing model in Section 2.3 where flows can choose among several alternative routes based on the current network congestion level. We show that our *invariance* type of result still holds in this more general model, i.e., when the system is large, there still exists a static pricing scheme whose performance can approach that of the optimal dynamic scheme. We can also weaken the assumption on fixed bandwidth requirement and allow flows to change their rate based on the current price. We investigate this problem in Section 2.4.

In networks of today and in the future, the capacity will be very large, and the network will be able to support a large number of users. The work reported in this chapter demonstrates under general assumptions and different network problem settings that, when a network is large, significant simplicity can be exploited for pricing based network control. Our result also shows the importance of the *average network condition* when the system is large, since the parameters of the static schemes are

determined by average conditions rather than instantaneous conditions. These results will help us develop more efficient and realistic algorithms for controlling large networks.

2.1.2 Related Works

Our work is also related to the work in [32, 33], and the references therein. However, in their work, the price is set *a priori*, and the focus is on how to admit and route each flow. Our work (as well as [17, 31]) explicitly models the users' price-elasticity, and considers the problem of how to set the price. The impact of large systems has also been studied for flow control problems in [34–36], however, these works focus on the single-link case with a fixed number of flows.

2.1.3 Additional Comments

Before we proceed with the details of our results, we would like to comment on what "average network conditions" and "static schemes" mean in real networks. In practice, over a long enough period of time, the network condition is usually non-stationary, and even the average network statistics could vary. Hence, in real networks, when we say a "static scheme", it does not necessarily mean that the price is fixed over the entire time. Rather, the "static scheme" should be interpreted as *prices being static over the time period for which the network statistics do not change* (which is typically still fairly long in real networks) and the *average network condition* should be interpreted as the average *over such a time period*. Over longer time scales, the prices should still adapt to the changes in the dominant network condition, although relatively slowly.

2.2 Pricing in a General Multi-class Loss Network

2.2.1 Model

The basic model that we consider in this section is that of a multi-class loss network with Poisson arrivals and arbitrary service time distributions. There are L links in the network. Each link $l \in \{1, ..., L\}$ has capacity R^l. There are I classes of users. We assume that flows generated by users from each class have a fixed route through the network. The routes are characterized by a matrix $\{C_i^l, i = 1, ..., I, l = 1, ..., L\}$, where $C_i^l = 1$ if the route of class i traverses link l, and $C_i^l = 0$ otherwise. Let $\vec{n} = \{n_1, n_2, ..., n_I\}$ denote the state of the system, where n_i is the number of flows of class i currently in the network. We assume that each flow of class i requires a fixed amount of bandwidth r_i. *The fixed routing and fixed bandwidth assumption will be weakened in Sections 2.3 and 2.4, respectively.*

Flows of class i arrive to the network according to a Poisson process with rate $\lambda_i(u_i)$. The rate $\lambda_i(u_i)$ is a function of the price u_i charged to users of class i. Here u_i is defined as the price per unit time of connection. Therefore $\lambda_i(u_i)$ can be viewed as the "demand function" and it represents the *price-elasticity* of class i. We assume that for each class i, there is a "maximal price" $u_{\max,i}$ such that $\lambda_i(u_i) = 0$ when $u_i \geq u_{\max,i}$. Therefore by setting a high enough price u_i the network can prevent users of class i from entering the network. We also assume that $\lambda_i(u_i)$ is a continuous and strictly decreasing function for $u_i \in (0, u_{\max,i})$. Once admitted, a flow of class i will hold r_i amount of resource in the network and pay a cost of u_i per unit time, until it completes service, where u_i is the price set by the network at the time of the flow arrival. The service times[2] are *i.i.d.* with mean $1/\mu_i$. The service time distribution is general[3].

[2]We use the notions of *service time* and *holding time* interchangeably.

[3]By assuming that the demand function $\lambda_i(u_i)$, the mean holding time $1/\mu_i$ and the capacity R^l are fixed, we have assumed that the network condition is stationary. We will comment in Section 2.2.7 how the results in this chapter should be interpreted when this assumption does not hold.

The bandwidth requirement determines the set of feasible states $\mathbf{\Omega} = \{\vec{n} : \sum_{i=1}^{I} n_i r_i C_i^l \leq R^l \quad \forall l\}$. A flow will be blocked if the system becomes infeasible after accommodating it. Other than this feasibility constraint, the network provider can charge a different price to each flow, and by doing so, the network provider strives to maximize the revenue collected from the users. The way in which price is determined can range from the simplest *static pricing schemes* to more complicated *dynamic pricing schemes*. In a *dynamic pricing scheme*, the price at time t can depend on many factors at the moment t, such as the current congestion level of the network, etc. On the other hand, in *a static pricing scheme*, the price is fixed over all time t, and does not depend on these factors. Intuitively, the more factors a pricing scheme can be based on, the more information it can exploit, and hence the higher the performance (i.e., revenue) it can achieve.

The dynamic pricing scheme that we study in this section is more sophisticated than the one in [17]. Firstly, we allow the network provider to exploit the knowledge of the immediate past history of states up to length d. Note that when the exponential holding time assumption is removed, the system is no longer Markovian. There will typically be correlations between the past and the future given the current state. In order to achieve a higher revenue, the network provider can potentially exploit this correlation, i.e., it can use the past to predict the future, and use such prediction to determine price.

Secondly, we allow the network provider to exploit prior knowledge of the parameters of the incoming flows. In particular, the network provider knows the holding time of the incoming flows, and can charge a different price accordingly. In order to achieve a higher revenue, the network provider can thus use pricing to control the composition of flows entering the network, for example, short flows may be favored under certain network conditions, while long flows are favored under others. We assume that the price-elasticity of flows is independent of their holding times.

For convenience of exposition, we restrict ourselves to the case when the range of the service time can be partitioned into a collection of disjoint segments, and the

price is the same for flows that are from the same class and whose service times fall into the same segment. Specifically, let $\{a_k, k = 1, 2, ...\}$ be an increasing sequence of positive numbers that approach $+\infty$ as $k \to +\infty$. Let $a_0 = 0$. The service time of a flow is a non-negative real number and hence must fall into one of the segments $[a_{k-1}, a_k), k = 1, 2, ...$ We assume that at any time t, for all flows of class i whose service times T_i fall into segment $[a_{k-1}, a_k)$, we charge the same price $u_{ik}(t)$, i.e., we do not care about the exact value of T_i as long as $T_i \in [a_{k-1}, a_k)$.

The *dynamic pricing scheme* can thus be written as

$$\begin{aligned} u_i(t, T_i) &= u_{ik}(t) = g_{ik}(\vec{n}(s), s \in [t-d, t]), \\ &\text{for } T_i \in [a_{k-1}, a_k), \end{aligned} \tag{2.1}$$

where $\vec{n}(s), s \in [t-d, t]$ reflects the immediate past history of length d, T_i is the holding time of the incoming flow of class i, and g_{ik} are functions from $\mathbf{\Omega}^{[-d,0]}$ to the set of real numbers $\mathbf{R}$. *By incorporating the past history in the functions g_{ik}, we can study the effect of prediction on the performance of the dynamic pricing scheme without specifying the details of how to predict.* Let $\vec{g} = \{g_{ik}, i = 1, ..., I, k = 1, 2, ...\}$.

The system under such a dynamic pricing scheme can be shown to be stationary and ergodic under very general conditions. One such condition is stated in the following proposition. Readers can refer to Appendix A.1 for the proof.

Proposition 2.2.1 *Assume that for all classes i, the arrival rates $\lambda_i(u)$ are bounded from above by some constant λ_0. Further, for each class i, the holding times are* i.i.d. *with finite mean and independent of the holding times of other classes. If the price is only dependent on the current state of the system, and/or a finite amount of past history (i.e., prediction based on past history), and/or the holding times of the incoming flows, then the stochastic process $\vec{n}(t)$ (i.e. the system state) is asymptotically stationary as $t \to \infty$ and the stationary version is ergodic.*

We are now ready to define the performance objective function. For each class i, let $\tilde{T_{ik}}$ be the mean service time for flows of class i whose service time T_i falls into

segment $[a_{k-1}, a_k)$, i.e., $\tilde{T_{ik}} = \mathbf{E}\{T_i | T_i \in [a_{k-1}, a_k)\}$. The expectation is taken with respect to the service time distribution of class i. Let $p_{ik} = \mathbf{P}\{T_i \in [a_{k-1}, a_k)\}$ be the probability that the service time T_i of an incoming flow of class i falls into segment $[a_{k-1}, a_k)$. We can decompose the original arrivals of each class into a spectrum of substreams. Substream k of class i has service time in $[a_{k-1}, a_k)$. Its arrival is thus Poisson with rate $\lambda_i(u)p_{ik}$, since we assume that the price-elasticity of flows is independent of T_i.

For any dynamic pricing scheme $\vec{g}$, the expected revenue achieved per unit time is given by

$$\begin{aligned} & \lim_{\zeta\to\infty} \sum_{i=1}^{I} \frac{1}{\zeta} \mathbf{E}\left[\int_0^{\zeta} \sum_{k=1}^{\infty} \lambda_i(u_{ik}(t)) u_{ik}(t) \tilde{T_{ik}} p_{ik}\, dt\right] \\ = & \sum_{i=1}^{I}\sum_{k=1}^{\infty} \mathbf{E}\left[\lambda_i(u_{ik}(t)) u_{ik}(t) \tilde{T_{ik}} p_{ik}\right], \end{aligned}$$

where the expectation is taken with respect to the steady state distribution. The limit on the left hand side as the time $\zeta \to \infty$ exists and equals to the right hand side due to stationarity and ergodicity. Note that the right hand side is independent of t (from stationarity). The performance of the *optimal* dynamic policy is thus given by:

$$J^* \triangleq \max_{\vec{g}} \sum_{i=1}^{I}\sum_{k=1}^{\infty} \mathbf{E}\left[\lambda_i(u_{ik}(t)) u_{ik}(t) \tilde{T_{ik}} p_{ik}\right].$$

Finally, we construct the performance objective for *static pricing schemes*. In a static pricing scheme, the price for each class is fixed, i.e., it does not depend on the current state of the network, nor does it depend on the individual holding time of the flow. Let u_i be the static price for class i. Let $\vec{u} = [u_1, ..., u_I]$. Under this static pricing scheme $\vec{u}$, the expected revenue per unit time is:

$$J_0 = \sum_{i=1}^{I} \lambda_i(u_i) u_i \frac{1}{\mu_i}(1 - \mathbf{P}_{loss,i}[\vec{u}]),$$

where $\mathbf{P}_{loss,i}[\vec{u}]$ is the blocking probability for class i. Therefore the performance of the *optimal* static policy is

$$J_s \triangleq \max_{\vec{u}} \sum_{i=1}^{I} \lambda_i(u_i) u_i \frac{1}{\mu_i}(1 - \mathbf{P}_{loss,i}[\vec{u}]).$$

When the exponential holding time assumption is removed, we can no longer use the Dynamic Programming approach as in [17] to find the optimal dynamic pricing scheme. We will instead study the behavior of the optimal dynamic pricing scheme by bounding its performance. These bounds will then help us establish the main result that, when the network is large, an appropriately chosen static pricing scheme can achieve almost the same performance as that of the optimal dynamic scheme. By definition, the performance of any static pricing scheme becomes a lower bound for the performance of the optimal dynamic pricing scheme. An upper bound is presented next.

2.2.2 An Upper Bound

The upper bound is of a similar form to that in [17]. Let $\lambda_{\max,i} = \lambda_i(0)$ be the maximal value of λ_i. For convenience, we write u_i as a function of λ_i. Let $F_i(\lambda_i) = \lambda_i u_i(\lambda_i)$, $\lambda_i \in [0, \lambda_{\max,i}]$. Further, let J_{ub} be the optimal value of the following nonlinear programming problem:

$$\begin{aligned} \max_{\lambda_i, i=1,\ldots,I} \quad & \sum_{i=1}^{I} F_i(\lambda_i) \frac{1}{\mu_i} \\ \text{subject to} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i C_i^l \le R^l \quad \text{for all } l, \end{aligned} \tag{2.2}$$

where $1/\mu_i$, r_i are the mean holding time and the bandwidth requirement, respectively, for flows from class i, C_i^l is the routing matrix and R^l is the capacity of link l.

Proposition 2.2.2 *If the function F_i is concave in $(0, \lambda_{\max,i})$ for all i, then $J^* \le J_{ub}$.*

Proof Consider an optimal dynamic pricing policy. Let $n_{ik}(t)$ be the number of flows of substream k of class i in the system at time t, hence $n_i(t) = \sum_{k=1}^{\infty} n_{ik}(t)$. Let $\lambda_{ik}(t) = \lambda_i(u_{ik}(t))$. From Little's Law, we have

$$\mathbf{E}[n_{ik}(t)] = \mathbf{E}[\lambda_{ik}(t)p_{ik}]\tilde{T_{ik}}.$$

The expectation is taken with respect to the steady state distribution.

Now let

$$\lambda_i^* = \frac{\sum_{k=1}^{\infty} \mathbf{E}[\lambda_{ik}(t)]p_{ik}\tilde{T_{ik}}}{\sum_{k=1}^{\infty} p_{ik}\tilde{T_{ik}}}.$$

Note that $\sum_{k=1}^{\infty} p_{ik}\tilde{T_{ik}} = 1/\mu_i$, therefore

$$\frac{\lambda_i^*}{\mu_i} = \sum_{k=1}^{\infty} \mathbf{E}[\lambda_{ik}(t)]p_{ik}\tilde{T_{ik}} = \sum_{k=1}^{\infty} \mathbf{E}[n_{ik}(t)] = \mathbf{E}[n_i(t)].$$

At any time t, $\sum_{i=1}^{I} n_i(t) r_i C_i^l \leq R^l$ for all l. Therefore

$$\sum_{i=1}^{I} \frac{\lambda_i^*}{\mu_i} r_i C_i^l \leq R^l \text{ for all } l \ .$$

Since the functions F_i are concave, using Jensen's inequality, we have,

$$\begin{aligned} J_{ub} &\geq \sum_{i=1}^{I} F_i\left(\lambda_i^*\right) \frac{1}{\mu_i} \\ &\geq \sum_{i=1}^{I} \sum_{k=1}^{\infty} F_i\Big(\mathbf{E}[\lambda_{ik}(t)]\Big) p_{ik}\tilde{T_{ik}} \\ &\geq \sum_{i=1}^{I} \sum_{k=1}^{\infty} \mathbf{E}\Big[F_i(\lambda_{ik}(t))\Big] p_{ik}\tilde{T_{ik}} = J^*. \end{aligned}$$

■

The upper bound (2.2) has a simple and intuitive form. Its objective function can be viewed as an approximation of the average revenue without taking into account blocking, while the constraints simply keep the load at all links to be no greater than 1

(where the load at a link l is defined by $\frac{1}{R^l}\sum_{i=1}^{I}\frac{\lambda_i}{\mu_i}r_i C_i^l$). The maximizer $[\lambda_1, ..., \lambda_I]$ of the upper bound (2.2) also induces a set of optimal prices $u_i^{ub} = u_i(\lambda_i)$. It is interesting to note that although the dynamic pricing scheme can use prediction and exploit prior knowledge of the parameters of the incoming flows, the upper bound (2.2) and its induced optimal prices are indifferent to these additional mechanisms.

Remark: The concavity assumption on F_i is essential in the proof of the upper bound and the result that follows. A linear $\lambda_i(u_i)$, as used in many applications, guarantees that F_i is concave. If F_i is not concave, some sort of "convexification" procedure needs to be invoked. Readers can refer to [17] for discussions on the implications of non-concave F_i.

2.2.3 Asymptotic Optimality of the Static Scheme

In this chapter, we are interested in studying *large systems*, i.e., when the capacity and the number of users in the network are large. We will show that, as the network grows large, the relative difference between the revenue of the optimal dynamic scheme, the revenue of the optimal static scheme, and the upper bound, will approach zero. We first study the following scaling **(S1)**, which is also used in [17], for modeling large systems. A different scaling will be studied in Section 2.2.6.

(S1) *Let $c \geq 1$ be a scaling factor. We consider a series of systems scaled by c. All systems have the same topology. The scaled system has capacity $R^{l,c} = cR^l$ at each link l, and the arrival rate of each class i is $\lambda_i^c(u) = c\lambda_i(u)$. Let $J^{*,c}$, J_s^c and J_{ub}^c be the optimal dynamic revenue, optimal static revenue, and the upper bound, respectively, for the c-scaled system.*

We are interested in the performance difference of the dynamic pricing schemes and the static pricing schemes when $c \uparrow \infty$. We first note that, under the scaling **(S1)**, *the normalized upper bound J_{ub}^c/c is a constant independent of c*, because J_{ub}^c is obtained by maximizing $\sum_{i=1}^{I} c\lambda_i u_i(\lambda_i)/\mu_i$, subject to the constraints

$\sum_{i=1}^{I} c\lambda_i r_i C_i^l/\mu_i \leq cR^l$, for all l. Therefore the optimal price induced by the upper bound is also independent of c.

Fix a set of static prices $\vec{u}$. Let $\mathbf{P}^c_{loss,i}[\vec{u}]$ denote the blocking probability of class i in the c-scaled system. The following lemma illustrates the behavior of $\mathbf{P}^c_{loss,i}[\vec{u}]$ as $c \to \infty$ under the scaling **(S1)**. Recall that the load at a link l is defined by $\frac{1}{R^l}\sum_{i=1}^{I}\frac{\lambda_i}{\mu_i} r_i C_i^l$.

Lemma 2.2.1 *Let λ_i be the arrival rate of flows from class i at a given set of static prices $\vec{u}$. Under the assumptions of Poisson arrivals and general holding time distributions, if the load at each resource is less than or equal to 1, i.e.,*

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i C_i^l \leq R^l \text{ for all } l,$$

*then under scaling **(S1)**, as $c \to \infty$, the blocking probability $\mathbf{P}^c_{loss,i}[\vec{u}]$ of each class i goes to 0, and the speed of convergence is at least $1/\sqrt{c}$, i.e., $\mathbf{P}^c_{loss,i}[\vec{u}] = O(\frac{1}{\sqrt{c}})$. Further, when the load at all links that class i traverses is strictly less than 1, the speed of convergence is exponential, i.e.,*

$$\limsup_{c\to\infty} \frac{1}{c} \log \mathbf{P}^c_{loss,i}[\vec{u}] \leq \max_{l:C_i^l=1} \inf_{w>0} \Lambda_l(w) < 0,$$

where

$$\Lambda_l(w) = \sum_{j=1}^{I} \frac{\lambda_j}{\mu_j}(e^{r_j w} - 1)C_j^l - wR^l. \tag{2.3}$$

The proof of this lemma can be found in Appendix A.2. We now use this lemma to show the following main result:

Proposition 2.2.3 *If the function F_i is concave in $(0, \lambda_{\max,i})$ for all i, then under the scaling **(S1)**,*

$$\lim_{c\to\infty} \frac{1}{c} J_s^c = \lim_{c\to\infty} \frac{1}{c} J^{*,c} = \lim_{c\to\infty} \frac{1}{c} J_{ub}^c = J_{ub}.$$

Instead of proving Proposition 2.2.3, we will prove the following stronger result that the static prices induced by the upper bound are in fact asymptotically optimal.

Proposition 2.2.4 *Let $\vec{u}^{ub} = [u_1^{ub}, ..., u_I^{ub}]$ denote the set of static prices induced by the upper bound, i.e., let $u_i^{ub} = u_i(\lambda_i)$ for all i, where $[\lambda_i, i = 1, ..., I]$ is the maximizer of the upper bound (2.2). Let $\tilde{J}_s^c$ be the revenue for the c-scaled system under this static price. If the function F_i is concave in $(0, \lambda_{\max,i})$ for all i, then under the scaling* ***(S1)****,*

$$\lim_{c\to\infty} \frac{1}{c}\tilde{J}_s^c = \lim_{c\to\infty} \frac{1}{c} J_{ub}^c = J_{ub}.$$

Proof Since $\tilde{J}_s^c \le J_s^c \le J^{*,c} \le J_{ub}^c = cJ_{ub}$, in order to prove the above two propositions, we only need to show that $\lim_{c\to\infty} \tilde{J}_s^c/c = J_{ub}$.

For every static price $\vec{u} = [u_1, ...u_I]$ falling into the constraint of J_{ub}, i.e.,

$$\sum_{i=1}^{I} \frac{c\lambda_i(u_i) r_i C_i^l}{\mu_i} \le cR^l \quad \text{for all } l, \tag{2.4}$$

let J_0^c denote the revenue under this static price. Since (2.4) guarantees that the condition of Lemma 2.2.1 is met, we have $\mathbf{P}_{loss,i}^c[\vec{u}] \to 0$, as $c \to \infty$. Therefore

$$\begin{aligned} \lim_{c\to\infty} \frac{J_0^c}{c} &= \lim_{c\to\infty} \sum_{i=1}^{I} \lambda_i(u_i) u_i \frac{1}{\mu_i}(1 - \mathbf{P}_{loss,i}^c[\vec{u}]) \\ &= \sum_{i=1}^{I} \lambda_i(u_i) u_i \frac{1}{\mu_i}. \end{aligned} \tag{2.5}$$

If we take the static price u_i^{ub} induced by the upper bound as our static price, then inequality (2.4) is satisfied. By definition, $J_0^c = \tilde{J}_s^c$, and the right hand side of (2.5) is exactly the upper bound. Therefore,

$$\lim_{c\to\infty} \frac{\tilde{J}_s^c}{c} = J_{ub},$$

and Propositions 2.2.3 and 2.2.4 then follow. ■

Propositions 2.2.2 and 2.2.3 are parallel to Theorems 6 and 7, respectively, in [17]. In [17], they are shown under the assumption of a single link and exponential holding time distribution. Propositions 2.2.2 and 2.2.3 tell us that extending the results of [17] from a single link to a network of links and from exponential holding time

distributions to arbitrary holding time distributions does not change the invariance result. In other words, there still exist static pricing schemes whose performance can approach that of the optimal dynamic pricing scheme when the system is large. Further, even though the dynamic pricing scheme can use prediction and exploit prior knowledge of the parameters of the incoming flows, the upper bound (2.2) turns out to be indifferent to these additional mechanisms. Therefore, these extra mechanisms only have a minimal effect on the long term revenue when the system is large.

To see how fast the gap between the performance of the optimal dynamic scheme and that of the static schemes diminishes to zero as $c \to \infty$, note that

$$\begin{aligned} \frac{J^{*,c} - J_s^c}{J^{*,c}} &\leq \frac{J_{ub} - \frac{1}{c} J_0^c(\vec{u}^{ub})}{J_{ub}} \\ &= \frac{\sum_{i=1}^{I} \lambda_i(u_i^{ub}) u_i^{ub} \frac{1}{\mu_i} \mathbf{P}_{loss,i}^c[\vec{u}^{ub}]}{\sum_{i=1}^{I} \lambda_i(u_i^{ub}) u_i^{ub} \frac{1}{\mu_i}} \leq \max_i \mathbf{P}_{loss,i}^c[\vec{u}^{ub}]. \end{aligned}$$

Therefore, the speed of convergence of Propositions 2.2.3 and 2.2.4 can be determined by the $\mathbf{P}_{loss,i}^c[\vec{u}^{ub}]$ that has the slowest speed of convergence to zero. If at the price u_i^{ub} induced by the upper bound, the load of all links is strictly less than 1, then the convergence of Propositions 2.2.3 and 2.2.4 is exponential:

$$\limsup_{c\to\infty} \frac{1}{c} \log \frac{J^{*,c} - J_s^c}{J^{*,c}} \leq \max_l \inf_{w>0} \Lambda_l(w) < 0.$$

On the other hand, if at the price u_i^{ub} the load of some links is equal to 1, then the convergence is $1/\sqrt{c}$, i.e.,

$$\frac{J^{*,c} - J_s^c}{J^{*,c}} = O(\frac{1}{\sqrt{c}}).$$

Our result is different from that of [17] and [31] in the following aspects: Firstly, we remove the assumption on the exponential holding time distributions. Secondly, we characterize two different regions for the speed with which the performance of the optimal static scheme approaches that of the optimal dynamic scheme. When the load on some links is equal to 1, the convergence is $1/\sqrt{c}$. When the load of all

links is strictly less than 1, the convergence is exponential. In [17] and [31], only the latter region is characterized.

Thirdly, and more importantly, we show in Proposition 2.2.4 that the static price induced by the upper bound is by itself asymptotically optimal. Hence, this result permits us to use the price induced by the upper bound directly in the static schemes. In [17] and [31], although the authors show the asymptotic optimality of the *optimal static scheme*, it is not clear whether the price induced by the upper bound can serve as a viable alternative, because with this price the load at some links could be equal to 1, and the convergence in this region is not characterized in their work.

2.2.4 Distance Neutral Pricing

When prices are congestion-dependent, two classes that traverse different routes will typically be charged different prices, even if they have the same price-elasticity function. A class that traverses a longer distance, or one that traverses a more congested route, will likely be charged a higher price.

However, if the following condition is satisfied, this distance-dependence can be eliminated when we adopt an asymptotically optimal static pricing scheme: We say a network has no *significant constraint* of resources if the *unconstrained* maximizer of $\sum_{i=1}^{I} F_i(\lambda_i)$ satisfies the constraint in (2.2). If there is no *significant constraint* of resources, there exists an asymptotically optimal static scheme whose prices depend only on the price-elasticity of each class, and are independent of their routes. To see this, we go back to the formulation of the upper bound (2.2). If the *unconstrained* maximizer of $\sum_{i=1}^{I} F_i(\lambda_i)$ satisfies the constraint, then it is also the maximizer of the *constrained* problem. In this case, if we use the prices induced by the upper bound as the asymptotically optimal static prices, the static prices will depend only on the function F_i, which represents the price-elasticity of the users.

This result suggests that the use of flat pricing, as in inter-state long distance telephone service in the United States, can also be economically near-optimal under

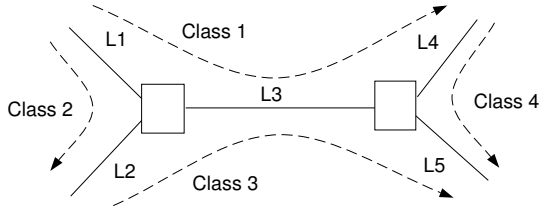


Fig. 2.1. The network topology

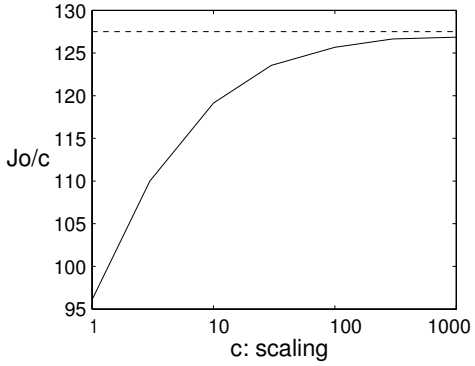


Fig. 2.2. The static pricing policy compared with the upper bound: when the capacity of link 3 is 5 bandwidth units. The dotted line is the upper bound.

appropriate conditions. Assuming that there is no significant constraint of resources in the domestic telephone network in the U.S., and all consumers have the same price-elasticity, then our result indicates that a flat (i.e., independent of both time and distance) pricing scheme will suffice, given that the capacity of the network is very large.

Table 2.1
Traffic and price parameters of 4 classes

	Class 1	Class 2	Class 3	Class 4
$\lambda_{\max,i}$	0.01	0.01	0.02	0.01
$u_{\max,i}$	10	10	20	20
Service Rate μ_i	0.002	0.001	0.002	0.001
Bandwidth r_i	2	1	1	2

2.2.5 Numerical Results

We report a few numerical results here. Consider the network in Fig. 2.1. There are 4 classes of flows. Their routes are shown in the figure. Their arrivals are Poisson. The function $\lambda_i(u)$ for each class i is of the form

$$\lambda_i(u) = \left[\lambda_{\max,i}\left(1 - \frac{u}{u_{\max,i}}\right)\right]^+,$$

i.e., $\lambda_i(0) = \lambda_{\max,i}$ and $\lambda_i(u_{\max,i}) = 0$ for some constants $\lambda_{\max,i}$ and $u_{\max,i}$. The price-elasticity is then

$$-\frac{\lambda_i'(u)}{\lambda_i(u)} = \frac{1/u_{\max,i}}{1 - u/u_{\max,i}} \text{ , for } 0 < u < u_{\max,i}.$$

The function F_i is thus

$$F_i(\lambda_i) = \lambda_i(1 - \frac{\lambda_i}{\lambda_{\max,i}})u_{\max,i},$$

which is concave in $(0, \lambda_{\max,i})$. The holding time is exponential with mean $1/\mu_i$. The parameters $\lambda_{\max,i}$, $u_{\max,i}$, service rate μ_i, and bandwidth requirement r_i for each class are given in Table 2.1.

We first consider a base system where the capacity of the five links are 10, 10, 5, 15, and 15 bandwidth units, respectively. The solution of the upper bound (2.2) is shown in Table 2.2. The upper bound is $J_{ub} = 127.5$. We then use simulations to verify how tight this upper bound is and how close the performance of the static

Table 2.2
Solution of the upper bound (2.2) when the capacity of Link 3 is 5 bandwidth units. The upper bound is $J_{ub} = 127.5$

	Class 1	Class 2	Class 3	Class 4
u_i	9.00	5.00	12.00	10.00
$\lambda_i(u_i)$	0.00100	0.00500	0.00800	0.00500
$\lambda_i(u_i)/\mu_i$	0.500	5.00	4.00	5.00

Table 2.3
Solution of the upper bound when the capacity of Link 3 is 15 bandwidth units. The upper bound is $J_{ub} = 137.5$

	Class 1	Class 2	Class 3	Class 4
u_i	5.00	5.00	10.00	10.00
$\lambda_i(u_i)$	0.00500	0.00500	0.0100	0.00500
$\lambda_i(u_i)/\mu_i$	2.50	5.00	5.00	5.00

pricing policy can approach this upper bound when the system is large. We use the price induced by the upper bound calculated above as our static price. We first simulate the case when the holding time distributions are exponential. We simulate c-scaled versions of the base network where c ranges from 1 to 1000. For each scaled system, we simulate the static pricing scheme, and report the revenue generated. In Fig. 2.2 we show the normalized revenue J_0/c as a function of c. As we can see, when the system grows large, the difference in performance between the static pricing scheme and the upper bound decreases. Although we do not know what the optimal dynamic scheme is, its normalized revenue J^*/c must lie somewhere between that of the static scheme and the upper bound. Therefore the difference in performance between the static pricing scheme and the optimal dynamic scheme is further reduced. For example, when $c = 10$, which corresponds to the case when the link capacity can accommodate around 100 flows, the performance gap between the static policy and the upper bound is less than 7%. The gap decreases as $1/\sqrt{c}$.

We next change the capacity of link 3 from 5 bandwidth units to 15 bandwidth units. The solution of the upper bound is shown in Table 2.3. The upper bound is $J_{ub} = 137.5$. Simulation using the static prices induced by the upper bound shows similar curves as in Fig. 2.2. This latter example also demonstrates distance-neutral pricing. For example, classes 1 and 2, and classes 3 and 4 have different routes but have the same price (and price-elasticity). Readers can verify that, in this example,

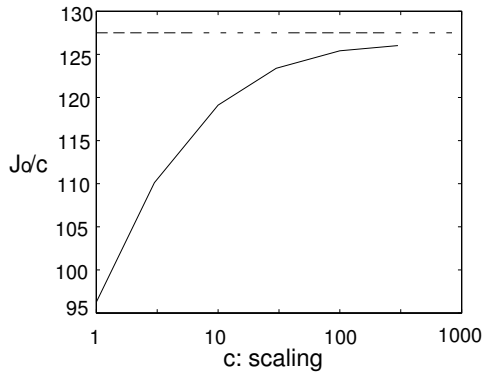


Fig. 2.3. The static pricing policy compared with the upper bound: when the service time distribution is Pareto and the capacity of link 3 is 5 bandwidth units.

if we lift the constraints in (2.2), and solve the upper bound again, we will get the same maximizer. Thus there is no *significant constraint* of resources, and hence the optimal price will only depend on the price-elasticity of each class and not on the specific route. Since class 1 has the same price-elasticity as class 2, its price is also the same as that of class 2, even though it traverses a longer route through the network.

We also simulate the case when the holding time distribution is deterministic. The result is the same as that of the exponential holding time distribution. The simulation result with heavy-tail holding time distribution also shows the same trend except that the sample path convergence (i.e., convergence in time) becomes very slow, especially when the system is large. For example, Fig. 2.3 is obtained when the holding time distribution is Pareto, i.e., the cumulative distribution function is $1 - 1/x^a$, with $a = 1.5$. We use the same set of parameters as in Fig. 2.2, and let the Pareto distribution have the same mean as that of the exponential distribution. Note that this distribution has finite mean but infinite variance. This demonstrates that our result is indeed invariant of the holding time distribution.

2.2.6 Scaling with Topological Changes

In Section 2.2.3, we have studied a large network under the scaling **(S1)**, where the network topology is fixed and we increase both the demand function $\lambda_i(u_i)$ and the capacity R^l proportionally. Hence, when the network is large, the number of users of each class i (i.e., between each source-destination pair) is also assumed to be large. This scaling is suitable for a *large capacity* network with a *simple* topology, e.g., network backbones. In this section, we will consider large networks where the number of users over each source-destination pair is *much smaller than the capacity of the network*. Large networks of this type arise naturally when the network topology becomes increasingly complex, for instance, when we include the access links into the topology. To illustrate a real world example, note that even though the links between Purdue University, Indiana and Columbia University, New York could have large capacities, the number of users communicating *between these two institutions* at any time is usually quite low. The high-capacity access link that connects Purdue University to the Internet is to accommodate the large aggregate traffic between Purdue University and *all* destinations on the Internet, while the amount of traffic to *any single* destination is much smaller.

The question we attempt to answer in this section is: will similar simplicity results as in Section 2.2.3 hold in this type of large networks? We will first present a different scaling model that allows the topology of the network to become more complex as we scale its capacity. We will then show that, under some reasonable assumptions, the performance of appropriately chosen static schemes will still approach that of the optimal dynamic schemes, as long as *the capacity of the network is large.*

We consider a series of networks indexed by the scaling factor c. We use $L(c)$, $I(c)$, and $C(c)$ to denote the number of links, the number of classes, and the routing matrix, respectively, in the c-th network. Note that in the scaling **(S1)** in Section 2.2.3, these quantities are assumed to be fixed for all c. In this section, in order to accommodate topological changes, we allow them to vary with c. We still use

$\lambda_i^c(u)$ and $R^{l,c}$ to denote the demand function for class i and the capacity of link l, respectively, in the c-th system. However, unlike the case for the scaling **(S1)**, $\lambda_i^c(u)$ and $R^{l,c}$ do not need to follow the linear scaling rule.

The *scale* of the c-th network is defined as

$$S(c) = \min_{l=1,\ldots,L(c)} \frac{R^{l,c}}{\max\limits_{i:C_i^l=1} r_i^c}.$$

We will consider the scaling **(S2)** that satisfies the following set of assumptions:

(A) $S(c) \to \infty$, as $c \to \infty$.

(B) For all networks, the maximum number of links on any route is bounded from above by a number M.

The first assumption simply states that, with large c, the capacity of each link in the network will be large compared to the bandwidth requirement of each flow that goes through the link. The second assumption limits the maximum number of hops each flow can traverse. This is a reasonable assumption: for example, in TCP/IP, the TTL (time-to-live) field in the IP header occupies only 8 bits. This effectively put an upper bound of 255 on the number of hops a flow can traverse within a network. Real Internet topology exhibits the small world phenomenon [37, 38], where the average number of hops of a source-destination pair is typically small. Any series of networks under scaling **(S1)** trivially satisfy scaling **(S2)**.

Finally, let $\lambda_{\max,i}^c$ be the maximal value of $\lambda_i^c(u)$. Let $u_i^c(\lambda_i)$ be the inverse function of $\lambda_i^c(u_i)$. We assume that:

(C) The function $F_i^c(\lambda_i) = \lambda_i u_i^c(\lambda_i)$ is concave in $(0, \lambda_{\max,i}^c)$ for all c and i.

Let $J^{*,c}$, J_s^c, and J_{ub}^c be the optimal dynamic revenue, the optimal static revenue, and the upper bound, respectively, for the c-th network. By Proposition 2.2.2, under Assumption C, the optimal dynamic revenue $J^{*,c}$ is no greater than J_{ub}^c for all c. In general, J_{ub}^c does not follow the simple linear rule as in scaling **(S1)**. Hence, Proposition 2.2.3 will not hold under scaling **(S2)**. Instead, we can show the following:

Proposition 2.2.5 *Under Scaling **(S2)** and Assumption C, as $c \to \infty$, the relative difference among the optimal dynamic revenue $J^{*,c}$, the optimal static revenue J_s^c, and the upper bound J_{ub}^c will converges to zero. That is,*

$$\lim_{k\to\infty} \frac{J_{ub}^c - J_s^c}{J_{ub}^c} = \lim_{k\to\infty} \frac{J_{ub}^c - J^{*,c}}{J_{ub}^c} = 0.$$

The proof is provided in Appendix A.3. Proposition 2.2.5 tells us that no matter how complex the topology of the network is, as long as the capacity of the network is large, the performance of the optimal static scheme will be close to that of the optimal dynamic scheme. We defer relevant numerical results until Section 2.3. The definition of $S(c)$ allows us to apply this result to certain networks with heterogeneous capacity. The network could have both large-capacity links and small-capacity links, and both large-bandwidth flows and small-bandwidth flows. As long as the capacity of each link is large compared to the bandwidth requirement of the flows *that traverse the link*, static schemes will suffice. On the other hand, there may exist network scenarios where $S(c)$ is not large, which means that the capacity of some links can only accommodate a small number of flows that go through them. Proposition 2.2.5 will not hold in such scenarios.

It is easy to check that Proposition 2.2.3 and Proposition 2.2.5 are equivalent under scaling **(S1)**. There is yet a difference between the results we can obtain under the two different types of scaling. Under scaling **(S1)**, we can show that the price $\vec{u}^{ub}$ induced by the upper bound suffices to be the near-optimal static price (see Proposition 2.2.4). Such conclusion cannot be drawn under scaling **(S2)**. The difficulty is that, under scaling **(S2)**, we cannot show that the blocking probability $\mathbf{P}_{loss,i}^c[\vec{u}^{ub}]$ at the static price $\vec{u}^{ub}$ goes to zero as $c \to \infty$ *when the constraints in the upper bound (2.2) are satisfied with equality.* In spite of this technical difficulty, we expect that in most cases the price induced by the upper bound would still suffice under scaling **(S2)**. We can argue as follows using the familiar *independent blocking assumption* underlying the Erlang Fixed Point approximation [39], i.e., we assume that blocking occurs independently at each link. Under the independent blocking assumption, the blocking probability B^l at each link l can be calculated *as if the*

traffic offered to the link l comprises independent Poisson streams with arrival rates that are "thinned" by other links in the network. At the price induced by the upper bound, the offered load at any link l *after "thinning"* will be no greater than the offered load *before "thinning"*, i.e., $\sum_i \frac{\lambda_i^c}{\mu_i} r_i$, which is no greater than the capacity $R^{l,c}$ of the link l. Applying the techniques in the proof of Lemma 2.2.1 to a single link, we can show that, *if the independent blocking assumption holds*, the blocking probability at each link will go to zero as $S(c) \to \infty$, and the convergence is uniform over all links. Since the number of hops each route can traverse is upper bounded by M, we can then infer that the blocking probability of all classes will go to zero uniformly as $S(c) \to \infty$. Therefore, using an argument similar to that of Proposition 2.2.3, we can infer that the price induced by the upper bound will suffice. The validity of the above argument relies on the independent blocking assumptions. A rigorous characterization of this convergence is an interesting problem for future work.

2.2.7 Remarks on Non-stationary Scenarios

We have shown under two different types of scaling that the performance of an appropriately chosen static pricing scheme will approach that of the optimal dynamic pricing scheme when the capacity of the network is large. We conclude this section with some discussion on the assumption we have made. In our model, we assume that the demand function $\lambda_i(u_i)$, the mean service time $1/\mu_i$ and the capacity R^l are fixed. Hence, we have assumed that the network condition is stationary. However, in practice the network condition is usually non-stationary and even the average network statistics can change over time. If we assume that the changes of these average network conditions are at a much slower time scale than that of the flow arrivals and departures, we can still use the result in this section with the following interpretation: the static scheme should be interpreted as *prices being fixed over a time period for which the network statistics do not change* (which is typically a fairly long time in real networks) and the average network condition should be

interpreted as *the average of the dominant network condition over such a time period*. Similarly, the invariance result regarding prediction should also be interpreted under this context: if the dynamic pricing scheme uses a prediction window (i.e., d in Equation (2.1)) that is smaller than the time scale of the changes of the average network condition, then the dynamic scheme will not significantly outperform the static pricing scheme. Over the longer time scale, the prices should still adapt to the changes in the dominant network condition and prediction on the changes of the average network statistics will help.

2.3 Dynamic Routing

Our analyses in Section 2.2 have focused on systems where routing is fixed. We next consider systems with dynamic routing. Many results in the Quality-of-Service (QoS) routing literature focus on finding the "best" route for each individual flow based on the instantaneous network conditions. When these QoS routing algorithms are used for dynamic routing, the network is typically required to first collect link states (such as available bandwidth, delay, etc.) on a regular basis. Then, when a request for a new flow arrives, the QoS routing algorithms are invoked to find a route that can accommodate the flow. When there are multiple routes that can satisfy the request, certain heuristics are used to pick one of the routes. However, such "greedy" routing policies may be sub-optimal system wide, because a greedy selection may result in an unfavorable configuration such that more future flows are blocked. Further, an obstacle to the implementation of this type of routing policies is that it consumes a significant amount of resources to propagate link states throughout the network. Propagation delay and stale information will also degrade the quality of the routing decision.

In this section, we will formulate a dynamic routing problem that directly optimizes the total system revenue. Although our model is simplified, it reveals important insight on the performance tradeoff among different types of routing policies.

We will first establish an upper bound on the performance of all *dynamic schemes*, which are schemes that can compute both the prices and the routing decisions based on the instantaneous congestion level of the network. We will then show that the performance of an appropriate chosen *static scheme*, which uses static prices and selects routes based on some pre-determined probabilities, can approach the performance of *the optimal dynamic scheme* when the system is large. The static scheme only requires some average parameters. It consumes less communication and computation resources, and is insensitive to network delay. Thus the static scheme is an attractive alternative for control of routing in large networks.

The network model is similar to that of Section 2.2, except that now a user of class i has $\theta(i)$ alternative routes that are represented by the matrix $\{H_{ij}^l\}$ such that $H_{ij}^l = 1$, if route j of class i uses resource l and $H_{ij}^l = 0$, otherwise. The dynamic schemes we consider have the following *idealized* properties: The routes of existing flows can be changed during their connection; and the traffic of a given flow can be transmitted on multiple routes at the same time. Thus our model captures the packet-level dynamic routing capability in the current Internet. These idealized capabilities allow dynamic schemes to "pack" more flows into the system. Yet, we will show that an appropriately chosen static scheme will have comparable performance to the optimal dynamic scheme.

Let n_i be the number of flows of class i currently in the network. Consider the k-th flow of class i, $k = 1, ..., n_i$. Let P_{ij}^k denote the proportion of traffic of flow k assigned to route j, $j = 1, ...\theta(i)$. Then, state $\vec{n} = \{n_1, ..., n_I\}$ is feasible if and only if

$$\begin{aligned} &\text{There exists } P_{ij}^k \text{ such that } \sum_{j=1}^{\theta(i)} P_{ij}^k = 1, \forall i, k, \\ &\text{and } \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} r_i H_{ij}^l \sum_{k=1}^{n_i} P_{ij}^k \leq R^l \quad \text{for all } l. \end{aligned} \tag{2.6}$$

The set of feasible states is $\mathbf{\Omega} = \{\vec{n} \text{ such that (2.6) is satisfied}\}$.

A dynamic scheme can charge prices based on the current state of the network, or a finite amount of past history, i.e., prediction based on past history. (For simplicity

we consider pricing schemes that are insensitive to the individual holding times.) An incoming flow will be admitted if the resulting state is in $\mathbf{\Omega}$. Once the flow is admitted, its route (i.e., P_{ij}^k) is assigned based on (2.6), involving (in an idealized dynamic scheme) possible rearrangement of routes of all existing flows. We assume that such rearrangement can be carried out instantaneously. Thus a dynamic scheme can be modeled by $u_i(t) = g_i(\vec{n}(s), s \in [t-d, t])$, where g_i is a function from $\mathbf{\Omega}^{[-d,0]}$ to $\mathbf{R}$. Let $\vec{g} = \{g_1, ..., g_I\}$.

The performance objective is again the expected revenue per unit time generated by the incoming flows admitted into the system. The performance of the *optimal dynamic scheme* is given by:

$$J^* \triangleq \quad \max_{\vec{g}} \mathbf{E}\{\sum_{i=1}^{I} \lambda_i(u_i(t))u_i(t)\frac{1}{\mu_i}\} \qquad (2.7)$$
$$\text{subject to (2.6).}$$

The expectation is taken with respect to the steady state distribution. Note that (2.7) is independent of t because of stationarity and ergodicity.

The set of dynamic schemes we have described may require complex capabilities (e.g., rearrangements of routes and transmitting traffic of a single flow over multiple routes) and hence may not be suitable for actual implementation. We make clear here that we do not advocate implementing such schemes but instead advocate implementing static schemes. In fact, we will show that, as the system scales, our static scheme will approach the performance of the optimal (and idealized) dynamic scheme. The static schemes do not require the afore-mentioned complex capabilities and could be an attractive alternative for network routing.

Let $u_i = u_i(\lambda_i)$ and $F_i(\lambda_i) = u_i(\lambda_i)\lambda_i$. Analogous to Proposition 2.2.2, we can derive the following upper bound on the optimal revenue in (2.7). The proof is a natural extension of that of Proposition 2.2.2.

Proposition 2.3.1 *If the function F_i is concave in $(0, \lambda_{\max,i})$ for all i, then $J^* \leq J_{ub}$, where J_{ub} is defined as the solution for the following optimization problem:*

$$J_{ub} \triangleq \max_{\lambda_{ij}} \quad \sum_{i=1}^{I} F_i(\sum_{j=1}^{\theta(i)} \lambda_{ij}) \frac{1}{\mu_i} \tag{2.8}$$

$$\textit{subject to} \quad \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\mu_i} H_{ij}^l r_i \leq R^l \quad \forall l.$$

Proof Assuming that we have already obtained an optimal dynamic policy $\vec{g}(.)$, let $u_i^*(t)$ and $P_{ij}^{k*}(t)$ be the price and routing proportions under such an optimal policy, let $\lambda_i(t) = \lambda_i(u_i^*(t))$ be the corresponding arrival rates, and let $n_i^*(t)$ be the evolution of the number of calls in the system. Let $P_{ij}^*(t) = \sum_{k=1}^{n_i^*(t)} P_{ij}^{k*}(t)/n_i^*(t)$. We can treat these as random variables. Let n_i^*, P_{ij}^*, and λ_i^* represent random variables with their corresponding stationary distribution, and let $\lambda_{ij} = \mathbf{E}\{n_i^* P_{ij}^*\}\mu_i$. Since $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} n_i^*(t) P_{ij}^*(t) H_{ij}^l r_i \leq R^l$, for all l, we have

$$\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\mu_i} H_{ij}^l r_i \leq R^l, \text{ for all } l.$$

Therefore λ_{ij} satisfies (2.9).

From Little's Law, we have

$$\mathbf{E}\{\lambda_i^*\}/\mu_i = \mathbf{E}\{n_i^*\} = \sum_{j=1}^{\theta(i)} \mathbf{E}\{n_i^* P_{ij}^*\} = \sum_{j=1}^{\theta(i)} \lambda_{ij}/\mu_i.$$

Now if the functions F_i are concave, we have

$$\begin{aligned} J^* &= \mathbf{E}\{\sum_{i=1}^{I} F_i(\lambda_i^*) \frac{1}{\mu_i}\} \leq \sum_{i=1}^{I} F_i(\mathbf{E}\{\lambda_i^*\}) \frac{1}{\mu_i} \\ &= \sum_{i=1}^{I} F_i(\sum_{j=1}^{\theta(i)} \lambda_{ij}) \frac{1}{\mu_i} \leq J_{ub}, \end{aligned}$$

by Jensen's Inequality. ■

We next construct our static scheme *using probablistic routing* as follows: The network charges a static price to all incoming flows, and the incoming flows are directed to alternative routes based on pre-determined probabilities. *Note that the*

static scheme does not have the idealized capabilities prescribed for the dynamic schemes, i.e., all traffic of a flow has to follow the same path, and rearrangement of routes of existing flows is not allowed. Let $\{u_i^s, P_{ij}^s\}$ denote such a static scheme, where u_i^s is the price for class i, and P_{ij}^s is the bifurcation probability that an incoming flow from class i is directed to route j. The performance of the *optimal static scheme* is then given by:

$$J_s \triangleq \max_{u_i^s, P_{ij}^s, \sum_{j=1}^{\theta(i)} P_{ij}^s = 1} \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \lambda_i(u_i^s) u_i^s P_{ij}^s \frac{1}{\mu_i} [1 - \mathbf{P}_{Loss,ij}], \tag{2.9}$$

where $\mathbf{P}_{Loss,ij}$ is the blocking probability experienced by users of class i routed to j.

We consider a special static scheme derived from the solution of the upper bound in Proposition 2.3.1. If λ_{ij}^{ub} is the maximal solution to the upper bound, we let $u_i^s = u_i(\sum_{j=1}^{\theta(i)} \lambda_{ij}^{ub})$, and $P_{ij}^s = \frac{\lambda_{ij}^{ub}}{\sum_{j=1}^{\theta(i)} \lambda_{ij}^{ub}}$. The revenue with this static scheme differs from the upper bound only by the term $(1 - \mathbf{P}_{Loss,ij})$, and this revenue will be less than J_s. However, under scaling **(S1)**, we can show that, as $c \to \infty$, $\mathbf{P}_{loss,ij} \to 0$. Therefore, we have our invariance result (stated next).

Proposition 2.3.2 *In the dynamic routing model, if the function F_i is concave in $(0, \lambda_{\max,i})$ for all i, then under the scaling **(S1)**,*

$$\lim_{c\to\infty} J_s^c/c = \lim_{c\to\infty} J^{*,c}/c = \lim_{c\to\infty} J_{ub}^c/c = J_{ub}.$$

Proof First we notice again that the normalized upper bound J_{ub}^c/c is fixed over all c. Therefore the optimal price induced by the upper bound is also independent of c. Since $J_s^c \le J^{*,c} \le J_{ub}^c = cJ_{ub}$, we only need to show that $\lim_{c\to\infty} \frac{J_s^c}{c} \ge J_{ub}$.

Now consider J_s^c. When we use the static price and routing probabilities induced by the upper bound, i.e., $u_i^s = u_i(\sum_{j=1}^{\theta(i)} \lambda_{ij}^{ub})$ and $P_{ij}^s = \frac{\lambda_{ij}^{ub}}{\sum_{j=1}^{\theta(i)} \lambda_{ij}^{ub}}$, then $\lambda_{ij}^{ub} = \lambda_i(u_i^s) P_{ij}^s$ is exactly the arrival rate to path j from flows of class i. Hence, the constraint of J_{ub} will be satisfied, i.e.,

$$\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}^{ub}}{\mu_i} H_{ij}^l r_i \le R^l \quad \forall l. \tag{2.10}$$

Let J_0^c denote the revenue under this static price. Since (2.10) guarantees that the condition of Lemma 2.2.1 is met, we have $\mathbf{P}_{loss,ij} \to 0$, as $c \to \infty$. Therefore

$$\begin{aligned} \lim_{c\to\infty} \frac{J_0^c}{c} &= \lim_{c\to\infty} \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \lambda_i(u_i^s) u_i^s \frac{1}{\mu_i} P_{ij}^s (1 - \mathbf{P}_{loss,ij}) \\ &= \sum_{i=1}^{I} \lambda_i(u_i^s) u_i^s \frac{1}{\mu_i} = J_{ub}. \end{aligned} \tag{2.11}$$

Therefore,

$$\lim_{c\to\infty} \frac{J_s^c}{c} \geq \lim_{c\to\infty} \frac{J_0^c}{c} \geq J_{ub}$$

and the result follows. ■

A similar result under scaling **(S2)** can be stated analogous to Proposition 2.2.5.

When the routing is fixed, by replacing λ_{ij} with λ_i, and H_{ij}^l with C_i^l, we recover Propositions 2.2.2 and 2.2.3 from the results in this section. When there are multiple alternate routes, the upper bound in Proposition 2.3.1 is typically larger than that of Proposition 2.2.2. Therefore one can indeed improve revenue by employing dynamic routing. However, Proposition 2.3.2 shows that, when the system is large, most of the performance gain can also be obtained by simpler static schemes that routes incoming flows based on pre-determined probabilities. Further, what we learn is that for large systems the capability to rearrange routes and to transmit traffic of a single flow on multiple routes does not lead to significant performance gains.

Not only can the static schemes be asymptotically optimal, they also have a very simple structure. Their parameters are determined by average conditions rather than instantaneous conditions. Collecting average information introduces less communication and processing overhead, and it is also insensitive to network delay. Hence the static schemes are much easier to implement in practice.

The asymptotically optimal static scheme also reveals the macroscopic structure of the optimal dynamic scheme. For example, the static price u_i^s shows the preference of some classes than the others, and the static bifurcation probability P_{ij}^s reveals the preference on certain routes than the other. While a "greedy" routing scheme tries

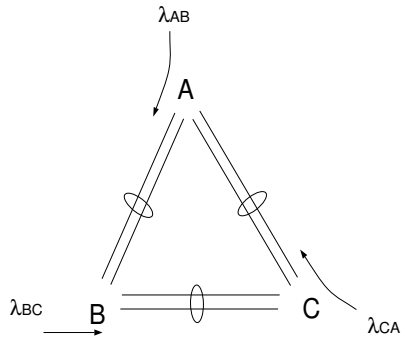


Fig. 2.4. The dynamic routing problem: There are 3 classes of flows, AB, BC, CA. For each class, there are two alternate routes. For example, for class AB, the direct one-link path is $A \to B$, while the indirect two-link path is $A \to C \to B$.

to accommodate each individual flow, the optimal static scheme may reveal that one should indeed prevent some flows from entering the network, or prevent some routes from being used.

We use the following examples to illustrate the results in this section. We first consider a triangular network (Fig. 2.4). There are three classes of flows, AB, BC, CA. There are two possible routes for each class of calls, i.e., a direct one-link path (route 1), and an indirect two-link path (route 2). Each call consumes one bandwidth unit along the link(s) and holds the link(s) for a mean time of 1 unit. Let the capacity of all links be R.

Let $\vec{\lambda} = \{\lambda_{AB,1}, \lambda_{AB,2}, \lambda_{BC,1}, \lambda_{BC,2}, \lambda_{CA,1}, \lambda_{CA,2}\}$. By Proposition 2.3.1, we can formulate the upper bound as (based on (2.8)):

$$\begin{aligned} J_{ub} &= \max_{\vec{\lambda}} \sum_{i=AB,BC,CA} \lambda_i u_i(\lambda_i) \\ \lambda_i &= \lambda_{i,1} + \lambda_{i,2}, \quad i = AB, BC, CA, \end{aligned} \tag{2.12}$$

subject to the following resource constraints:

$$\lambda_{AB,1} + \lambda_{BC,2} + \lambda_{CA,2} \leq R$$

$$\lambda_{BC,1} + \lambda_{AB,2} + \lambda_{CA,2} \leq R$$

$$\lambda_{CA,1} + \lambda_{AB,2} + \lambda_{BC,2} \leq R$$
$$\lambda_{i,j} \geq 0, \quad i = AB, BC, CA, j = 1, 2.$$

Once the upper bound is solved, we can find the near-optimal static scheme using the one induced by the upper bound, i.e., the price charged to class i, $i = AB, BC, CA$, is

$$u_i^s = u_i(\lambda_{i,1} + \lambda_{i,2}),$$

and the bifurcation probabilities are

$$\begin{aligned} P_{i,1}^s &= \frac{\lambda_{i,1}}{\lambda_{i,1} + \lambda_{i,2}} \\ P_{i,2}^s &= \frac{\lambda_{i,2}}{\lambda_{i,1} + \lambda_{i,2}}. \end{aligned}$$

Let $R = 100$. We consider the following examples:

1) When the price-elasticity function of all classes are $\lambda_{AB}(u) = \lambda_{BC}(u) = \lambda_{CA}(u) = 100(1-u)$, the solution of (2.12) gives $\lambda_{AB} = \lambda_{BC} = \lambda_{CA} = 50$, and the prices for each class are $u_{AB}^s = u_{BC}^s = u_{CA}^s = 0.5$. It also coincides with the solution of the unconstrained version of (2.12). This corresponds to the case of light traffic load. The price is only determined by the price-elasticity of each class. There are multiple solutions for the bifurcation. One example is $\lambda_{i,1} = 50$, and $\lambda_{i,2} = 0, i = AB, BC, CA$, i.e., all calls use the direct link.

2) When we change the price-elasticity of class AB to $\lambda_{AB}(u) = 500(1-u)$, the solution of (2.12) is shown in Table 2.4. This corresponds to the case of heavy traffic load. The price are raised from that of the unconstrained problem in order to limit incoming traffic. All constraints are binding. Note that here in order to maximize the revenue, class AB has a higher arrival rate λ_i than that of class BC and CA, and the network should allow flows from class AB to use indirect two-link path, while flows from classes BC and CA should not be allowed to use the indirect routes.

We next use a larger network example to demonstrate the optimality of static schemes with probablistic routing. We use the BRITE topology generation tool [40] and the Barabasi-Albert model [38] to generate a random network with 100 nodes

Table 2.4
Solution of the upper bound in the dynamic routing problem: when the price-elasticity of class AB is $\lambda_{AB}(u) = 500(1-u)$

	Class AB	Class BC	Class CA
$\lambda_{i,1}$	100	40.91	40.91
$\lambda_{i,2}$	59.09	0	0
λ_i	159.09	40.91	40.91
Price u_i^s	0.682	0.591	0.591
$P_{i,1}^s$	62.86%	100%	100%
$P_{i,2}^s$	37.14%	0	0

and 197 links. The Barabasi-Albert topology model is able to capture the power-law of node-connectivity in real Internet topologies. There are a total of 9900 source-destination (s-d) pairs. For each s-d pair, we use the set of minimum-hop paths as the alternate paths. The demand function for each source-destination pair is the same. We formulate the upper bound for the randomly-generated network, and use standard convex optimization methods to solve the static prices and the bifurcation probabilities. We then use simulation to obtain the static revenue under the prices and the bifurcation probabilities induced by the upper bound.

We present the following result from a typical simulation. The bandwidth for each link is 1000 units. The bandwidth requirement of each flow is 1 unit and the mean holding time is 1 unit. The demand function for each s-d pair is $\lambda(u) = 10(1-u)$. At this level of demand, around 22% of the links experience congestion, i.e., their respective constraints in (2.8) are binding. The upper bound is found to be 2.45×10^4, while the static revenue obtained from the simulation is 2.40×10^4. The relative difference is just 2%. This validates our result that the performance of static schemes is close to that of the optimal dynamic scheme and the upper bound. Among the 9900 s-d pairs, around 49% have multiple alternate routes. However, among those with multiple routes, only 30% actually use multiple routes. Hence, a large number of s-d pairs does not benefit from multiple alternative paths. The average number of routes between a s-d pair is 2.15. Finally, note that in this example, the end-to-end demand of each source-destination pair is at most $10(1-0)/1 = 10$, which is much smaller than the capacity of the links (1000 units). Hence, this simulation serves to validate our result under scaling **(S2)** where the number of users of each source-destination pair is much smaller than the capacity of the network. In all of our simulations, the prices induced by the upper bound turn out to be near-optimal, even although we have not been able to establish this optimality rigorously under scaling **(S2)**. We have also run simulations using other topology models and find similar results.

2.4 Elastic Flows

In previous sections we have restricted ourselves to the case when the bandwidth requirements of flows are fixed. In this section we will study an alternate model where users can change their bandwidth requirements according to the current congestion level of the network. For ease of exposition we assume that there is only one route for each class i. The routes are again represented by the matrix $\{C_i^l\}$ as in Section 2.2. Flows of class i enter the network according to a Poisson process with rate λ_i. The service times of flows of class i are i.i.d. with mean $1/\mu_i$. The service time distribution is general. Let $U_i(x_i)$ be the utility function for each class i, where x_i is the amount of resource assigned to a class i flow along its route. We assume that U_i is a continuous differentiable and strictly concave function of x_i, and $U_i(0) = 0$. This model is appropriate for real-time streaming applications that can change the transmission rate according to the network congestion level. For example, the utility function $U_i(x_i)$ can be taken as the index of reception quality when the real-time stream is transmitting at rate x_i.

The network tries to allocate resources to the flows so that the total utility of all flows supported by the network is maximized. For each flow, the resource allocation may vary over time. In this section, we will first establish the optimal dynamic scheme. We will then show, as before, that there exists a static scheme whose performance will approach that of the optimal dynamic scheme when the system is large. Surprisingly, this near-optimal solution is in a "fixed-bandwidth" and "loss-network" form as in Section 2.2.

2.4.1 The Optimal Dynamic Scheme

Let $n_i(t)$ be the number of flows from class i that are in the network at time t. Let $\vec{n}(t) = \{n_1(t), n_2(t), ..., n_I(t)\}$. The optimal resource assignment is then given by the solution to the following problem:

$$\begin{aligned} J^*(\vec{n}(t)) \triangleq \max_{x_1,...,x_I} \quad & \sum_{i=1}^{I} n_i(t) U_i(x_i) \\ \text{subject to} \quad & \sum_{i=1}^{I} n_i(t) x_i C_i^l \leq R^l, \end{aligned} \tag{2.13}$$

where $J^*(\vec{n}(t))$ can be interpreted as the maximal total utility achieved by the system at time t. For each t we can solve (2.13) and obtain the optimal assignment $x_i(t)$. Over time, this policy will optimize the total utility.

Remark: In the optimal assignment (2.13), each flow of class i will consume the same amount of resource x_i. This is a consequence of the concavity of U_i.

In the past (e.g., [14–16, 41]) this model has been used to study the behavior of TCP congestion control *when the number of flows in the system is fixed*. It has been shown that there exist distributed algorithms that can drive the flows to the optimal resource assignment. The notion of "price" arises naturally as Lagrange multipliers for the constraints. Some examples of such distributed algorithms resemble the control of TCP in the Internet. Therefore, TCP congestion control can be seen to maximize the total utility of a group of users with concave utility functions. Our model is different from theirs because we consider the *dynamics caused by the arrivals and departures of flows*. We are interested in finding alternative forms of resource assignment schemes that can also achieve near optimal total utility when the system is large. These schemes can then be used in cases when TCP does not work as well.

2.4.2 An Upper Bound

Let $\mathbf{E}[n_i]$ be the stationary mean of $n_i(t)$, i.e., $\mathbf{E}[n_i] = \lambda_i/\mu_i$. We formulate another optimization problem:

$$J_{ub} \triangleq \max_{x_1,...,x_I} \sum_{i=1}^{I} \mathbf{E}[n_i] U_i(x_i) \qquad (2.14)$$
$$\text{subject to } \sum_{i=1}^{I} \mathbf{E}[n_i] x_i C_i^l \le R^l.$$

Proposition 2.4.1 *The expected total utility of the optimal dynamic scheme is upper bounded by J_{ub}, i.e. $\mathbf{E}[J^*] \le J_{ub}$, where the expectation is taken with respect to the steady state distribution of $n_i(t)$.*

Proof Note that J^* is a function of $\vec{n}(t) = \{n_i(t), i = 1, ...I\}$. Then $J_{ub} = J^*(\mathbf{E}[\vec{n}])$.

To show that $\mathbf{E}[J^*(\vec{n})] \le J^*(\mathbf{E}[\vec{n}]) = J_{ub}$, it is sufficient to show that $J^*(\vec{n})$ is a concave function of $\vec{n}$,i.e., for any $\vec{n}^1 = [n_1^1, n_2^1, ..., n_I^1]$, $\vec{n}^2 = [n_1^2, n_2^2, ..., n_I^2]$ and $0 \le a \le 1$, let $n_i = an_i^1 + (1-a)n_i^2$, $\vec{n} = [n_i]$, we need

$$J^*(\vec{n}) \ge aJ^*(\vec{n}^1) + (1-a)J^*(\vec{n}^2).$$

In order to show this, let x_i^1, x_i^2 be the optimal assignment leading to $J^*(\vec{n}^1)$ and $J^*(\vec{n}^2)$ respectively. Let

$$x_i = \frac{an_i^1 x_i^1 + (1-a)n_i^2 x_i^2}{an_i^1 + (1-a)n_i^2},$$

then

$$\sum_{i=1}^{I} \left(an_i^1 + (1-a)n_i^2\right) x_i C_i^l \le R^l.$$

Since U_i is concave, we have

$$U_i(x_i) \ge \frac{an_i^1 U_i(x_i^1) + (1-a)n_i^2 U_i(x_i^2)}{an_i^1 + (1-a)n_i^2}.$$

Hence,

$$\begin{aligned} J^*(\vec{n}) &\ge \sum_{i=1}^{I} (an_i^1 + (1-a)n_i^2) U_i(x_i) \\ &\ge \sum_{i=1}^{I} \left(an_i^1 U_i(x_i^1) + (1-a)n_i^2 U_i(x_i^2)\right) \\ &= aJ^*(\vec{n}^1) + (1-a)J^*(\vec{n}^2), \end{aligned}$$

and the result follows. ■

2.4.3 Static Policy

Let $x^0 = \{x_1^0, x_2^0, ..., x_I^0\}$ be the maximizer of (2.14). Now consider the following control algorithm with a static rate assignment: when a new flow from class i arrives to the network, it will be assigned a rate x_i^0 if there is enough capacity available along its route, otherwise it will either be blocked, or, equivalently, be assigned a rate 0. Therefore, the flow is still elastic except that the rate is chosen according to *the average condition* as in (2.14) rather than *the instantaneous condition* as in (2.13). The flow will hold the same amount of resource x_i^0 until it leaves the system.

In such a system, the expected total utility will be

$$J_s \triangleq \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(x_i^0)(1 - \mathbf{P}_{loss,i}),$$

where $\mathbf{P}_{loss,i}$ is the blocking probability of class i. Under scaling **(S)**, we have the following proposition.

Proposition 2.4.2 *In the elastic flow model,*

$$\lim_{c\to\infty} \frac{1}{c} J_s^c = \lim_{c\to\infty} \frac{1}{c} \mathbf{E}[J^{*,c}] = \lim_{c\to\infty} \frac{1}{c} J_{ub}^c = J_{ub}.$$

Proof Since $\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} x_i^0 C_i^l = \sum_{i=1}^{I} \mathbf{E}[n_i] x_i^0 C_i^l \le R^l$, as $c \to \infty$, we have $\mathbf{P}_{loss,i} \to 0$. Therefore

$$\begin{aligned} J_s^c/c &= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(x_i^0)(1 - P_{loss,i}) \\ &\to \sum_{i=1}^{I} \mathbf{E}[n_i] U_i(x_i^0) = J_{ub}^c/c. \end{aligned}$$

Now $J_s^c/c \le \mathbf{E}[J^{*,c}]/c \le J_{ub}^c/c$, then the result follows. ■

An application of this result is on the rate control of real-time flows (e.g. audio and video streaming) on the Internet. A central question in congestion control of

streaming traffic is its fairness with respect to TCP. When real-time flows and TCP flows coexist in the same network, they should consume comparable bandwidth, and neither flows should be starved by the other. Among the existing congestion control schemes for real-time flows, some use the same AIMD (Additive Increase Multiplicative Decrease) idea as TCP [42]. They are usually fair with TCP if timeouts occur infrequently. However, these schemes typically produce a TCP-like saw-tooth type of trajectory, which leads to rapid changes in reception quality. Such rapid changes in quality are disconcerting for the viewer of multimedia flows [43]. Equation-based congestion control does not use AIMD and produces smoother rates at small time-scales. However, simulation results show that at time-scales around 10 seconds, the fluctuation is still quite significant [44]. There are yet other schemes, such as some binomial algorithms [45], which change the rate slower than TCP. However they are also slower in adapting to changing network conditions.

Note that fairness objectives are very closely related to the utility maximization objectives. For example, proportional fairness is equivalent to maximizing the total utility of a group of users with log-utility functions. If we adopt utility maximization as a substitute for the fairness requirement, we can use the result in this section to obtain a new class of congestion-control algorithms for real-time traffic. For example, consider the special case when α portion of the flows are real-time flows, and the rest are TCP flows. To be precise, let $n_i^{\mathrm{RT}}(t)$ and $n_i^{\mathrm{TCP}}(t)$ denote the number of real-time flows and TCP flows, respectively, at time t. Then their stationary means are $\mathbf{E}[n_i^{\mathrm{RT}}] = \alpha \mathbf{E}[n_i]$ and $\mathbf{E}[n_i^{\mathrm{TCP}}] = (1-\alpha)\mathbf{E}[n_i]$. Let us assign to real-time flows of class i the fixed bandwidth x_i^0 that is the maximizer of (2.14), and allow the real-time flows to use the same amount of bandwidth throughout the connection. Such fixed bandwidth allocation is beneficial to streaming applications because it ensures a stable reception quality for the viewer. Therefore the expected total utility achieved by real-time flows is given by

$$J^{\mathrm{RT}} = \mathbf{E}[n_i^{\mathrm{RT}}] U_i(x_i^0)(1 - \mathbf{P}_{loss,i}^{\mathrm{RT}}) = \alpha J_{ub}(1 - \mathbf{P}_{loss,i}^{\mathrm{RT}}),$$

where $\mathbf{P}_{loss,i}^{\mathrm{RT}}$ is the blocking probability experienced by the real-time flows. The total utility achieved by TCP flows at time t is given by the following optimization problem:

$$J^{\mathrm{TCP}} \triangleq \max_{x_1,\ldots,x_I} \quad \sum_{i=1}^{I} n_i^{\mathrm{TCP}}(t) U_i(x_i), \tag{2.15}$$

$$\text{subject to} \quad \sum_{i=1}^{I} n_i^{\mathrm{TCP}}(t) x_i C_i^l \le R^l - \sum_{i=1}^{I} n_i^{\mathrm{RT}}(t) x_i C_i^l.$$

The expected total utility achieved by both the real-time flows and the TCP flows, $J^{\mathrm{RT}} + \mathbf{E}[J^{\mathrm{TCP}}]$, is bounded from above by J_{ub} and bounded from below by J_s. Therefore, by Proposition 2.4.2,

$$\lim_{c\to\infty} \frac{J^{\mathrm{RT},c} + \mathbf{E}[J^{\mathrm{TCP},c}]}{c} = \lim_{c\to\infty} \frac{J_{ub}^c}{c} = J_{ub},$$

where $J^{\mathrm{RT},c}$, $J^{\mathrm{TCP},c}$ and J_{ub}^c are the respective utility when the system is scaled by c. Now by Lemma 2.2.1,

$$\lim_{c\to\infty} \mathbf{P}_{loss,i}^{\mathrm{RT}} = 0.$$

Therefore

$$\lim_{c\to\infty} \frac{J^{\mathrm{RT},c}}{c} = \alpha J_{ub},$$

and we conclude that

$$\lim_{c\to\infty} \frac{\mathbf{E}[J^{\mathrm{TCP},c}]}{c} = (1-\alpha) J_{ub}.$$

Note that by Proposition 2.4.2, $(1-\alpha)J_{ub}$ is also the limit of the normalized expected total utility achieved by the TCP flows as $c \to \infty$, *when the remaining portion α of the flows are also TCP flows*. This shows that when the same utility functions are used for both the real-time flows and TCP flows, assigning the fixed bandwidth x_i^0 to real-time flows does not degrade the performance of the TCP flows when the system is large.

It is interesting to compare existing congestion-control schemes with our scheme above. In existing schemes, flows start from an arbitrary initial condition, and congestion control is exercised *during* the connection. In our scheme, congestion control

is exercised *at the beginning* of the connection. The congestion controller reacts to changing network condition by choosing the correct initial bandwidth assignment for incoming flows. Although our scheme does not modify the bandwidth assignment for on-going flows, the difference between the total utility of our scheme and the optimal utility is minimal (when the system is large). Therefore, in the long run, the real-time flows and TCP flows will receive fair share of the bandwidth. In future work we plan to investigate the problem of efficiently distributing our congestion controller over the network.

2.5 Conclusion, Discussion, and Future Work

In this chapter, we have studied the simplification of pricing-based network control in large-capacity communication networks. We have shown under general settings that the performance of an appropriately chosen *static scheme* can approach that of the *optimal dynamic scheme* when the capacity of the network is large. These results have important implications for the design and control of large-capacity networks. Compared with the optimal dynamic scheme, the static scheme has several desirable features. The static schemes are much easier to obtain because of their simple structures. They are also much easier to execute since they do not require the collection of instantaneous load information. Instead, they only depend on some average parameters, such as the average load. Hence, they introduce less computation and communication overhead, and they are less sensitive to feedback delay. These advantages make the static scheme an attractive alternative for controlling large networks.

However, one should keep in mind that static schemes also have their disadvantage, namely, their lack of adaptivity. Static schemes could be more sensitive to modeling errors than dynamic schemes [46, 47]. If the parameters of the model are estimated incorrectly, the resulting static scheme may lead to bad performance. Further, as we discussed at the end of Section 2.2, the network condition may be

non-stationary and even the average network parameters may change over time. The static prices that are good for one time may not be good for the next moment. Therefore, we are not advocating that in practice *purely* static schemes (i.e., prices being fixed for all time) be used.

Nonetheless, we believe that the results in this chapter can be exploited to develop practical network control algorithms that are both *simple* and *adaptive*. One direction is to develop efficient algorithms that can compute the static prices based on the current dominant network condition and allow the prices to *adaptively* track the changes in the average network parameters. Here we briefly discuss one possible approach. Note that although the static prices are calculated by solving a *global* optimization problem, i.e., the upper bound (2.2) or (2.8), it is possible to develop a distributed solution. Indeed, we can associate a non-negative Lagrange multiplier p^l for the constraint at each resource l. The Lagrange multiplier p^l can be viewed as the implicit cost that summarizes the congestion information at link l. Given p^l, in order to determine the price for class i, one only needs to know the price-elasticity of class i (i.e., the function F_i) and the sum of the implicit costs along the path that flows of class i traverse. *Therefore we can decompose the global optimization problem into several subproblems for each class.* We can have the core routers update these implicit costs based on the congestion level at each link and have the ingress router serve as "brokers" to probe these implicit costs and determine the price offered to users of each class i. The idea of this decomposition has been used in [15] and [16] to develop distributed algorithms for optimization flow control, and it is also mentioned in [17] for computing the static prices in the single-link case.

The distributed algorithm described above can achieve adaptivity in several ways. Firstly, the edge router can use the *online measurement* of flow arrivals at different price levels to update its estimate of the demand function. Secondly, the core router can use the *online measurement* of the congestion level at each link to update the implicit costs.

It is instructive to compare such a *quasi-static* distributed algorithm with typical dynamic and static schemes. Note that since the distributed algorithm updates the implicit costs based on online measurements of the congestion level at the link, it can also be viewed as a *dynamic scheme*. However, the distributed algorithm is based on the asymptotic optimality of the static schemes. It attempts to solve for the static prices according to the current dominant network condition. Hence, we refer to the distributed algorithm as being *quasi-static*. On one hand, the distributed algorithm exploits the simplicity of the static scheme, thus has a simple form and is easier to implement than the optimal dynamic scheme. When the network condition is stationary, the prices computed by the distributed algorithm will converge to that of the near-optimal static scheme. On the other hand, the distributed algorithm *is* by definition also dynamic in that, when the network condition is non-stationary, the prices computed by the distributed algorithm will *track* the long term changes. Hence, the distributed algorithm is more robust than *purely* static schemes.

In Chapter 3, we will turn to the simplification of Quality-of-Service routing in high-bandwidth networks. Although there is no notion of "price" in the QoS routing problem in Chapter 3, simplicity results similar to those in this chapter still hold. Further, we will develop in Chapter 3 a *quasi-static* adaptive scheme similar to the one we just described. We will show that such a scheme can significantly reduce the computation and communication overhead of QoS routing without sacrificing the routing performance.

As a final remark of this chapter, we note that there are also possibilities of extremal changes in network conditions, such as failures of network components. When such situations occur, an appropriate *immediate* response is usually more important than an optimal but *slower* one. For such situations, other levels of network control, such as failure detection and fault recovery, are more appropriate than the pricing-based control studied in this chapter.

3. SIMPLIFICATION OF QUALITY-OF-SERVICE ROUTING IN HIGH-BANDWIDTH NETWORKS

3.1 Introduction

In Chapter 2, we have modeled the network control problem as a pricing problem, and we have shown that significant simplicity can arise in pricing-based network control when the capacity of the system is large. In particular, we have shown that simple static pricing schemes can approach the performance of the optimal dynamic scheme in large-capacity networks. Static schemes have a number of attractive features. The near-optimal static prices can easily be derived from the solution of a simple non-linear programming problem. The static schemes are also easy to implement because they do not require the collection of each *instantaneous* state information of the network. Rather, they only depend on some average statistics such as the average offered load. Therefore, static schemes introduce less communication and computation overhead and they are insensitive to feedback delays. These features make static schemes attractive alternatives for the control of large networks.

Having said that, we note that there are still two factors that may prevent us from using static schemes as practical control mechanisms for real networks. Firstly, to obtain the parameters of the static scheme, one typically needs to solve a global optimization problem. A centralized approach for solving this optimization problem is usually impractical in real systems due to scalability and reliability concerns. Secondly, in real systems the average network condition may also change over time. Purely static schemes lack the adaptivity that is required to track the non-stationary behavior in real networks. Therefore, in order to exploit the simplicity results in

Chapter 2, it is imperative that we are able to develop *distributed* and *adaptive* solutions based on simple static control schemes.

In this chapter, we will study how to develop such distributed and adaptive solutions. In order to highlight the practical significance of these types of solutions, we will derive our result in the context of the Quality-of-Service (QoS) routing problem in high-bandwidth networks. Nonetheless, the algorithms that we developed will also apply to other control problems, including the pricing problem in Chapter 2. In this chapter, we will focus on demonstrating the benefits of the proposed solution, and we will show how the largeness of the system can be exploited to develop an *optimization-based approach* to QoS routing, which significantly reduces the computation and communication overhead of QoS routing without sacrificing the routing performance. We will defer to Chapter 4 the rigorous study of the convergence properties of the proposed distributed algorithms.

3.1.1 Quality-of-Service Routing

Future telecommunication networks are expected to support applications with diverse Quality-of-Service requirements. Quality-of-Service (QoS) routing is an important component of such networks and has received considerable attention over the past decade (for a good survey, see [48] and the reference therein). The objective of QoS routing is two-fold: to find a feasible path for each incoming connection; and to optimize the usage of the network by balancing the load.

In this chapter, as in the majority of studies on QoS routing, we assume a source routing model where routing decisions are made at the point where connection requests originate. In most of these studies, researchers take the following view of the QoS routing problem: The links are "dumb" and they advertise their status. The intelligence lies in the end-systems (sources or edge routers) to compute paths based on the current knowledge of the link states.

The above paradigm would have worked well if the link states were stable. However, not all link state metrics are stable. In particular, the available bandwidth metric of a link is inherently dynamic and changes frequently as connections enter and leave the network. Therefore, the link state advertisement and the QoS routing algorithm have to be executed frequently in order to keep up with the changes in the link states. This leads to a significant amount of computation and communication overhead. To reduce the computation and communication burden, the frequency of the computation and the link state updates then need to be contained. This could, however, result in staleness of the link state information and inaccuracy in the routing decisions. Hence, there is a fundamental tradeoff between the amount of computation and communication resources consumed and the quality of the routing decisions. This tradeoff is usually difficult to analyze and researchers have had to resort to simulation studies [49–52]. These studies reveal that the performance of existing QoS routing schemes degrades when computation and link state updates become infrequent. However, the extent to which the performance degrades depends not only on how infrequently the computation and link state updates are made, but also on a large number of other factors that include: the specifics of the path computation algorithm, the topology and the demand pattern of the network, the cost metrics assigned for each link, the link state update strategy, and the strategy to handle routing failures, etc. In general, the exact level of performance degradation is hard to predict.

3.1.2 Summary of Contributions

In this chapter, we take a different view of the QoS routing problem. We view the network (including the end-systems *and* the links) that employs QoS routing as an integral entity that jointly optimizes some global utility function. Once the solution to this optimization problem is found, the network will be driven to an efficient operating point, and the routing performance will be close to optimal. No

further computation and communication are needed as long as the prevailing network condition remains essentially unchanged.[1]

We refer to our proposed scheme as the *optimization based approach* for QoS routing. In high-bandwidth networks, such an optimization based approach is advantageous due to a known simplicity result, which is similar in spirit to the results that we have developed in Chapter 2 for simplification of network pricing in large-capacity networks. In particular, one can show that *simple proportional routing schemes* can approach the performance of the *optimal dynamic routing schemes* when the capacity of the network is large [20, 32, 33]. In a *proportional routing scheme*, calls are routed to alternate paths based on pre-determined probabilities. The right routing probabilities can be derived from the solution of a simple optimization problem that depends only on the *average* demand and capacity of the network.

We will develop an online, distributed algorithm that can efficiently solve the optimization problem and compute the right routing probabilities. Fig. 3.1 provides a high-level view of the optimization based approach. Each link in the network is associated with an implicit cost. The implicit cost summarizes the congestion level at the link and can be updated by the observed demand and capacity at the link. Thus, we equip the link with only a minimal amount of intelligence (i.e., to update the implicit cost). It turns out that the implicit cost is the only information that the end-system needs to solve the optimization problem. The end-system has three components: a path-finding component that maintains a set of alternate paths; an optimization component that solves for the optimal routing probabilities; and a randomized routing component that routes each incoming connection based on the precomputed routing probabilities.

Compared with existing QoS routing schemes, our optimization based approach has the following advantages:

[1]In practice, some computation and communication will still be required to track changes in the network condition. However, a nice feature of our work is that computation and communication intensive operations can be done at very long time-scales, with a negligible impact on performance.

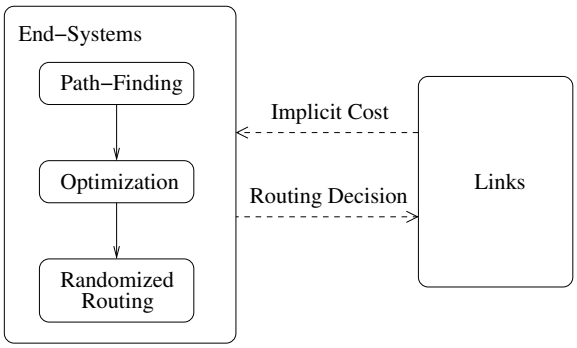


Fig. 3.1. Our optimization based approach

(1) The computation and communication overhead can be greatly reduced without sacrificing performance. Once the optimal operating point is found, the same routing parameters can be used by a large number of future arrivals, as long as the *average* network condition remains unchanged. Infrequent computation and link state updates will only affect the speed of convergence of the distributed algorithm, but not the end-result that the algorithm converges to.

In practical networks, the *average* network condition can also change gradually over time (non-stationary behavior), e.g., during the course of a day. Our distributed algorithm will track the changes in the average network condition and adjust the operating point accordingly. Note that in a control system, there has always been the issue of the right *time-scale of control*. A nice feature of our proposed solution is that, the control that needs to be done at a fast time scale, i.e., the randomized routing, is very simple; while the control that requires a large amount of computation, i.e., the optimization of routing probabilities and the search for new alternate paths, can be carried out over a much slower time scale. Using the right separation of control time scales, our optimization based approach ensures near optimal performance even when the computation and communication become infrequent.

(2) The operating characteristics of the network can be analytically studied. Given the network model, we can easily predict the operating point by solving the optimization problem. In contrast, due to the complexity of the system, the analysis of existing QoS routing schemes appears to be intractable, especially under inaccurate link state information and infrequent computation.

(3) The desired operating point can be tuned by appropriately choosing the utility functions. The optimization based approach allow us not only to *predict* the operating point of the network, but also to *control* it. By choosing different utility functions for different classes and source-destination pairs, we can achieve the desired balance among the service levels offered to different groups of users. For example, when the network becomes congested, connections with a larger number of hops could suffer significantly more blocking than shorter connections. In our

optimization based approach, this can be avoided by assigning longer connections a utility function that has a higher marginal utility.

3.1.3 Related Work

The optimal control of loss networks has been studied extensively in the past. Both off-line [53–55] and simulation based schemes [46] have been proposed. Our contribution is to propose an *online* solution for QoS routing. Our online scheme exploits the fact that simplicities arise in high-bandwidth networks, which we have seen in Chapter 2 and we will discuss in more detail in Section 3.2. These results lead to a much simpler and easily decomposable optimization problem.

Our proposed solution employs a proportional routing scheme. The asymptotic optimality of the proportional routing scheme in large systems has been known for some time [32, 33]. However, a major criticism of proportional routing schemes has been the following: if the demand is incorrectly estimated, the computed routing probabilities could lead to poor performance [46, 47]. We solve this problem by using an *adaptive* algorithm that does not rely on any prior knowledge of the demand. The Adaptive Proportional Routing scheme proposed in [56, 57] is also related to our work. In their scheme, each class measures the amount of blocking along each alternate paths, and uses the inverse Erlang formula to estimate a "virtual capacity" grabbed by the class along each path. Then each class locally optimizes the routing probabilities based on the demand and these virtual capacities. Compared with the Adaptive Proportional Routing scheme, the advantage of our optimization based approach is that the optimality of the resulting operating point and the convergence of the algorithm can be rigorously shown. Further, the implicit costs provide additional information for discovering new alternate paths.

The mathematical structure of the optimization problem studied in this chapter is closely related to those found in *multi-path* flow control problems [14, 58–60]. In [14], two classes of solutions to flow control problems are categorized, i.e., primal solutions

and dual solutions. For the *single-path* flow control problem, both the primal and the dual solutions have been studied extensively (see [61] for a good survey). On the other hand, the *multi-path* flow control problem has received less attention. Our implicit cost based solution can be viewed as a dual solution to this problem. A similar algorithm was proposed in [58]. In [58], the authors claim that their algorithm is one of the Arrow-Hurwicz algorithms [62]. However, the convergence of the Arrow-Hurwicz algorithm was established in [62] only for the case when the objective function is strictly concave, which is not true for the problem at hand. In this chapter, we report a new result that characterizes the convergence correctly. Primal solutions to the *multi-path* flow control problem were developed in [59, 60].

The rest of the chapter is organized as follows: In Section 3.2, we present the asymptotic optimality of the proportional routing scheme. In Section 3.3, we derive the distributed algorithm for computing the optimal routing probabilities and obtain the proposed QoS algorithm. We discuss implementation issues in Section 3.4, present simulation results in Section 3.5, and then conclude.

3.2 Simplification of QoS Routing in Large Networks

3.2.1 The Model

We adopt a multi-class loss network model. There are L links in the network. Each link $l \in \{1, ..., L\}$ has capacity R^l. There are I classes of users. Each class is associated with one source-destination pair, and some given QoS requirements. Flows of class i arrive to the network according to a Poisson process with rate λ_i. Once admitted, a flow of class i will hold r_i amount of bandwidth. (For the moment we assume that bandwidth is the only QoS metric. The extension to multiple QoS metrics will be addressed in Section 3.3.4.) The service times within a class are *i.i.d.* and independent of the arrival process. The service time distribution is general with mean $1/\mu_i$. Each admitted flow of class i generates v_i amount of revenue per

unit time. The objective of the network is to maximize the revenue from all flows admitted into the network.

Such a network model could represent the backbone of an ISP serving applications with different QoS requirements. The revenue v_i could either be actual money, or simply an assigned weight that represents the network's preference for each class. The bandwidth requirement r_i could be some form of *effective bandwidth* for flows of class i. There could be multiple classes associated with each source-destination pair, differing in their bandwidth requirement r_i and revenue v_i.

In this section, we assume that each class i has set up $\theta(i)$ alternate paths using, for example, MPLS [63] (we will address how these alternate paths can be found in Section 3.3.3). The alternate paths are represented by a matrix $[H_{ij}^l]$ such that $H_{ij}^l = 1$ if path j of class i uses link l, and $H_{ij}^l = 0$ otherwise. We denote the state of the system by a vector $\vec{n} = [n_{ij}, i = 1, ...I, j = 1, ..., \theta(i)]$, where n_{ij} is the number of flows of class i currently using path j. The bandwidth requirements and the capacity constraints then determine the set of feasible states $\mathbf{\Omega_n} = \{\vec{n} : \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} n_{ij} r_i H_{ij}^l \leq R^l \text{ for all } l\}$.

We denote the routing decision (which can be time varying) for class i by a vector

$$\vec{p_i} \quad = \quad [p_{i1}, p_{i2}, ..., p_{i,\theta(i)}],$$

where

$$p_{ij} = \mathbf{Pr}\{\text{an incoming flow of class } i \text{ is routed to path } j\}.$$

Thus

$$\vec{p_i} \in \Omega_i \triangleq \{p_{ij} \geq 0, \sum_{j=1}^{\theta(i)} p_{ij} \leq 1, \text{ for all } j\}.$$

An incoming flow of class i will be admitted with probability $\sum_{j=1}^{\theta(i)} p_{ij}$, and, if admitted, it will be routed to path j with probability $p_{ij} / \sum_{k=1}^{\theta(i)} p_{ik}$. Let $\vec{p} = [\vec{p}_1, ..., \vec{p}_I]$.

A *dynamic routing scheme* is one where routing decisions can adapt to the changing utilization level of the network. For example, $\vec{p}(t)$ can be a function of the current

state of the network, i.e., $\vec{p}(t) = g(\vec{n}(t))$. Note that this model can characterize virtually any QoS routing proposals that select paths based on the current snapshot of the network. Alternatively, $\vec{p}(t)$ can be a function of some past history of network states $\vec{n}(s), s \in [t-d, t]$, where d is the length of the history information. The network can use the past history to predict the future, and use prediction to improve the routing decision. $\vec{p}(t)$ can also depend on the service time T of the incoming connection, if this information is available. The routing policy can then be written, in a most general form, as

$$\vec{p}(t) \quad = \quad g(\vec{n}(s), s \in [t-d, t]; T). \tag{3.1}$$

As in Proposition 2.2.1 of Chapter 2, it can be shown that the system under any policy g will always converge to a stationary version, and the stationary version is ergodic.

Each admitted flow of class i will generate v_i amount of revenue per unit time. The dynamic routing scheme that maximizes the long term average revenue is then

$$J^* \triangleq \max_g \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \mathbf{E}_g \left[n_{ij}(t) \right] v_i,$$

where $\mathbf{E}_g$ denotes the expectation taken with respect to the stationary distribution under policy g.

Finally, in a *static scheme*, the routing policy is represented by a time-invariant vector $\vec{p}$. This corresponds to a proportional routing scheme. The performance of the static scheme is:

$$J_0 \triangleq \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij} v_i [1 - \mathbf{P}_{Loss,ij}],$$

where $\mathbf{P}_{Loss,ij}$ is the blocking probability experienced by flows of class i routed to path j.

3.2.2 Asymptotic Optimality of Static Schemes

The drawback of dynamic schemes is that the optimal schemes are difficult to find, and the implementation of these dynamic schemes will consume a large amount

of computation and communication resources. It turns out that when the capacity of the system is large, *simple static schemes can approach the performance of the optimal dynamic scheme.* This has been the central theme of our results in Chapter 2. Here, we rephrase the main result under the context of QoS routing. We scale the capacity and the demand proportionally by $c > 1$, i.e., in the c-scaled network, the capacity at each link l is $R^{l,c} = cR^l$, and the arrival rate of each class i is $\lambda_i^c = c\lambda_i$. The following result shows that when c is large[2], a simple static scheme will suffice. The static scheme is constructed as follows:

Step 1: Solve the following optimization problem:

$$\begin{aligned} J_{ub} = \max_{\vec{p}\in\Omega} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} \sum_{j=1}^{\theta(i)} p_{ij} v_i \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l \le R^l \quad \text{for all } l, \end{aligned} \tag{3.2}$$

where $\Omega = \bigotimes_{i=1}^{I} \Omega_i$.

Step 2: Use the optimal point $\vec{p}$ in (3.2) as the static policy. Let J_s be its performance.

The following proposition shows that the normalized revenue of the static scheme constructed above will approach that of the optimal dynamic scheme when $c \to \infty$.

Proposition 3.2.1 *Let $J^{*,c}$ and J_s^c be the revenue of the optimal dynamic scheme and the revenue of the static scheme constructed above, respectively, in the c-scaled system, then*

$$\lim_{c\to\infty} J_s^c/c = \lim_{c\to\infty} J^{*,c}/c = J_{ub}.$$

Proposition 3.2.1 can be shown as in Proposition 2.2.3 of Chapter 2. Precisely, we can first show that cJ_{ub} is an upper bound of $J^{*,c}$ *under any dynamic routing policy* g [32, 33]. Note that the static revenue J_s^c differs from the upper bound cJ_{ub} only by the term $(1-\mathbf{P}_{Loss,ij})$. Now since $\vec{p}$ satisfies the constraint of (3.2), the traffic load at each link is no greater than 1. Lemma 2.2.1 in Chapter 2 then ensures that

[2]Note that here largeness does not imply over-provisioning.

the blocking probability goes to zero as $c \to \infty$. Finally, because $J_s^c \leq J^{*,c} \leq cJ_{ub}$, Proposition 3.2.1 then follows.

3.3 The Optimization Based Approach to QoS Routing

There is a continuing trend to deploy routers with larger and larger link capacities in the Internet. Therefore, the results in the last section offer important insights on the QoS routing problem in the high-bandwidth networks of today and the future. Firstly, by solving a simple upper bound, we can obtain a simple *time-invariant* scheme that is *close to optimal*. Once we *precompute* the routing probabilities according to (3.2), the result can be used for a large number of future arrivals. Thus, *the computation overhead can be greatly reduced.* Secondly, the upper bound (3.2) replaces the instantaneous capacity constraint $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} n_{ij} r_i H_{ij}^l \leq R^l$ by an average load constraint $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l \leq R^l$. Hence, the precomputation only needs to react to the *average congestion level* in the network rather than the *instantaneous congestion level*. The staleness of the link state information is no longer a major issue!

Therefore, if we are able to solve the upper bound (3.2) efficiently, we can obtain a QoS routing algorithm that is close to optimal in large networks and that can tolerate infrequent computation and infrequent link state updates. However, we still need to consider the following issues.

- The upper bound is a global optimization problem. A distributed solution is desired.
- Some parameters, such as λ_i and μ_i, could be unknown a priori and changing gradually over time. A solution is needed that can automatically adapt to these changes.

We next present an adaptive, distributed algorithm for solving the upper bound. Before we proceed, we note that in many scenarios, it is also desirable to modify

the upper bound to improve fairness. We can view the upper bound (3.2) as a constrained optimization problem that maximizes some aggregate utility functions:

$$\begin{aligned} \max_{\vec{p}\in\Omega} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l \le R^l \quad \text{for all } l, \end{aligned} \tag{3.3}$$

where the utility function U_i is linear: $U_i(p) = p$. A linear utility function, however, does not possess good fairness properties: for example, connections with a larger number of hops could be completely blocked to give way to connections with fewer hops. To improve fairness, we can use a strictly concave utility function U_i, as in flow control problems [14–16, 41]. The derivative $U_i'(\sum_{j=1}^{\theta(i)} p_{ij})$ represents the amount of marginal utility lost if the overall admission probability for class i is further reduced. The desired balance among different classes can be achieved by tuning the revenue v_i and the utility function U_i. Proposition 3.2.1 can be generalized to the case with concave utility functions as in Chapter 2 (see Appendix B.1). In this chapter, we will use utility functions that satisfy $U_i'(1) = 1$. This choice of the utility function ensures that the revenue v_i is correctly reflected by the marginal utility when all flows of class i can be admitted, i.e., $v_i U_i'(\sum_{j=1}^{\theta(i)} p_{ij}) = v_i$ when $\sum_{j=1}^{\theta(i)} p_{ij} = 1$. As long as the utility function follows this rule, our simulation results indicate that the revenue is usually not affected much by changing the utility functions.

3.3.1 A Distributed Algorithm

Let $\vec{p^*}$ be the maximizer of the modified upper bound (3.3). Because the objective function is concave and the constraint set is convex and compact, a maximizer always exists. However, it is generally not unique, since the objective function is not strictly concave. (Note that even if U_i is strictly concave, the overall problem is not, because of the linear operation $\sum_{j=1}^{\theta(i)} p_{ij}$.)

The form of the upper bound motivates us to study its dual. However, when the objective function of the primal problem is not strictly concave, the dual problem may not be differentiable. To circumvent this difficulty, we use ideas from Proximal Optimization Algorithms [64, Chapter 3.4.3]. The idea is to add a quadratic term to the objective function. We introduce an auxiliary variable y_{ij} for each p_{ij}. Let $\vec{y}_i = [y_{ij}, j = 1, ..., \theta(i)]$ and $\vec{y} = [\vec{y}_1, .., \vec{y}_I]$. The optimization becomes:

$$\begin{aligned} \max_{\vec{p}\in\Omega, \vec{y}\in\Omega} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i \\ & - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \qquad (3.4) \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l \leq R^l \quad \text{for all } l, \end{aligned}$$

where ν_i is some positive number chosen for each class i. For a fixed $\vec{y}$, the objective function in (3.4) is strictly concave. It is easy to show that the optimal value of (3.4) coincides with that of (3.3). In fact, if $\vec{p} = \vec{p^*}$ is the maximizer of (3.3), then $\vec{p} = \vec{p^*}, \vec{y} = \vec{p^*}$ is the maximizer of (3.4).

The standard Proximal Optimization Algorithm then proceeds as follows:

Algorithm $\mathcal{P}$:

At the t-th iteration,

P1) Fix $\vec{y} = \vec{y}(t)$ and maximize the augmented objective function with respect to $\vec{p}$. To be precise, this step solves:

$$\begin{aligned} \max_{\vec{p}\in\Omega} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i \\ & - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \qquad (3.5) \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l \leq R^l \quad \text{for all } l. \end{aligned}$$

Since the objective function in (3.5) is now strictly concave, the maximizer exists and is unique. Let $\vec{p}(t)$ be the solution to this optimization.

P2) Set $\vec{y}(t+1) = \vec{p}(t)$.

Step P1) can now be solved through its dual. Let $q^l, l = 1, ..., L$ be the Lagrange Multiplier for the constraints in (3.5). Let $\vec{q} = [q^1, ..., q^L]$. Define the Lagrangian as:

$$\begin{aligned} L(\vec{p}, \vec{q}, \vec{y}) &= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i \\ &\quad - \sum_{l=1}^{L} q^l (\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l - R^l) \\ &\quad - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \\ &= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} \left\{ U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} \sum_{l=1}^{L} H_{ij}^l q^l \right. \\ &\quad \left. - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \right\} + \sum_{l=1}^{L} q^l R^l. \end{aligned} \tag{3.6}$$

Let $q_{ij} = \sum_{l=1}^{L} H_{ij}^l q^l$, $\vec{q_i} = [q_{ij}, j = 1, ..., \theta(i)]$. The objective function of the dual problem is then:

$$D(\vec{q}, \vec{y}) = \max_{\vec{p} \in \Omega} L(\vec{p}, \vec{q}, \vec{y}) = \sum_{i=1}^{I} B_i(\vec{q_i}, \vec{y_i}) \frac{\lambda_i}{\mu_i} + \sum_{l=1}^{L} q^l R^l, \tag{3.7}$$

where

$$\begin{aligned} B_i(\vec{q_i}, \vec{y_i}) &= \max_{\vec{p_i} \in \Omega_i} \left\{ U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} q_{ij} \right. \\ &\quad \left. - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \right\}. \end{aligned} \tag{3.8}$$

Note that in the definition of the dual objective function $D(\vec{q}, \vec{y})$ in (3.7), we have decomposed the original problem into I separate subproblems. Given $\vec{q}$, each class can solve the routing probabilities $\vec{p_i}$ via its local subproblem (3.8) independently. If we interpret q^l as the implicit cost per unit bandwidth at link l, then q_{ij} is the total cost per unit bandwidth for all links in the path j of class i. Thus the q_{ij}

captures all the information each subproblem needs about the path class i traverses. We note that an important feature of this decomposition is that the subproblem (3.8) is independent of the parameters λ_i and μ_i. This makes online implementation particularly easy.

The dual problem of (3.5), given $\vec{y}$, is:

$$\min_{\vec{q} \geq 0} D(\vec{q}, \vec{y}).$$

Since the objective function of the primal problem (3.5) is strictly concave, the dual is always differentiable. The gradient of D is

$$\frac{\partial D}{\partial q^l} = R^l - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij}^0 r_i H_{ij}^l, \tag{3.9}$$

where p_{ij}^0 solves the local subproblem (3.8). Then step P1) can be solved by using the gradient descent iteration on the dual variable, i.e.,

$$q^l(t+1) = \left[q^l(t) - \alpha^l (R^l - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij}^0 r_i H_{ij}^l) \right]^+, \tag{3.10}$$

where $[.]^+$ denotes the projection to $[0, +\infty)$.

The class of distributed algorithms we will use in this chapter can be summarized as follows:

Algorithm $\mathcal{A}$:

At the t-th iteration:

A1) Fix $\vec{y}(t)$ and use the gradient descent iteration (3.10) on the dual variable $\vec{q}$. Depending on the number of times the descent iteration is executed, we will obtain a dual variable $\vec{q}(t+1)$ that either exactly or approximately minimizes $D(\vec{q}, \vec{y}(t))$ (and, equivalently, solves (3.5)). Let K be the number of times the dual descent iteration is executed.

A2) Let $\vec{p}(t)$ be the primal variable that maximizes, over all $\vec{p} \in \Omega$, the Lagrangian $L(\vec{p}, \vec{q}(t+1), \vec{y}(t))$ corresponding to the new dual variable $\vec{q}(t+1)$. Set $\vec{y}(t+1) = \vec{p}(t)$.

From now on, we will refer to (3.10) as the *dual update*, and step A2) as the *primal update*.

A stationary point of algorithm $\mathcal{A}$ can be defined as a primal-dual pair $(\vec{y^*}, \vec{q^*})$ such that

$$\vec{y^*} = \operatorname*{argmax}_{\vec{p}\in\Omega} L(\vec{p}, \vec{q^*}, \vec{y^*}),$$
$$q^{l,*} \geq 0 \text{ and } \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} y_{ij}^* r_i H_{ij}^l \leq R^l \text{ for all } l, \text{ and}$$
$$q^{l,*}\left(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} y_{ij}^* r_i H_{ij}^l - R^l\right) = 0 \text{ for all } l.$$

By standard duality theory, any stationary point $(\vec{y^*}, \vec{q^*})$ of the algorithm $\mathcal{A}$ solves the augmented problem (3.4). Hence $\vec{p} = \vec{y^*}$ solves the upper bound (3.3).

An important question is how large K (in step A1) needs to be for algorithm $\mathcal{A}$ to converge to a stationary point. The standard proximal optimization theory [64, Chapter 3.4.3] requires $K = \infty$, i.e., at each iteration the optimization (3.5) has to be solved exactly. This requirement essentially corresponds to a time-scale separation between the time-scale of the primal updates and that of the dual updates. When $K < \infty$, at best an approximate solution to (3.5) is obtained at each iteration. If the accuracy of the approximate solution can be controlled appropriately (see [65]), one can still show the convergence of algorithm $\mathcal{A}$. However, in this case the number of dual updates K has to depend on the required accuracy and usually needs to be large.

For online implementation, one cannot carry out the dual update infinitely many times *for one iteration of algorithm* $\mathcal{A}$. It is also difficult to distributively control the accuracy of the approximate solution to (3.5). Hence, in this work we use a different approach. The following result is new and shows that, by appropriately choosing the stepsize α^l, the algorithm $\mathcal{A}$ converges *for any choice of* $K \geq 1$. No time-scale separation is needed! The proof is highly technical, and since this convergence result

has independent interest even for problems other than QoS routing, we will dedicate a separate chapter (Chapter 4) to the study of the convergence of Algorithm $\mathcal{A}$.

Proposition 3.3.1 *Fix $1 \leq K \leq \infty$. As long as the stepsize α^l is small enough, the algorithm $\mathcal{A}$ will converge to a stationary point $(\vec{y^*}, \vec{q^*})$ of the algorithm, and $\vec{p^*} = \vec{y^*}$ solves the upper bound (3.3). The sufficient condition for convergence is:*

$$\max_l \alpha^l < \begin{cases} \frac{2}{\mathcal{SL}} \min_i \frac{\mu_i \nu_i v_i}{\lambda_i r_i^2} & \text{if } K = \infty \\ \frac{1}{2\mathcal{SL}} \min_i \frac{\mu_i \nu_i v_i}{\lambda_i r_i^2} & \text{if } K = 1 \\ \frac{4}{5K(K+1)\mathcal{SL}} \min_i \frac{\mu_i \nu_i v_i}{\lambda_i r_i^2} & \text{if } K > 1 \end{cases},$$

where $\mathcal{L} = \max\{\sum_{l=1}^{L} H_{ij}^l, i = 1, ..., I, j = 1, ...\theta(i)\}$ is the maximum number of hops for any path, and $\mathcal{S} = \max\{\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} H_{ij}^l, l = 1, ..., L\}$ is the maximum number of paths going through any link.

Remark: The sufficient condition for $K = 1$ differs from that of $K = \infty$ only by a constant factor. For $K > 1$, our result requires that the stepsizes decrease on the order of $O(1/k^2)$. This is probably not the tightest possible result, and we conjecture that stepsizes of order $O(1)$ would work for any K. However, we leave this for future work. Note also that ν_i appears on the right hand side of the condition. Hence, by making the objective function more concave, we also relax the requirement on the stepsize α^l. Finally, Proposition 3.3.1 does not require the routing matrix $[H_{ij}^l]$ to be of full rank.

3.3.2 Distributed Implementation

Algorithm $\mathcal{A}$ lends naturally to online distributed implementation. The ingress router for each class is responsible for determining the routing probabilities for this class. To do so, the ingress router only needs to solve the local subproblem (3.8) using the implicit costs q^l at all core routers that class i traverses. An efficient algorithm can solve (3.8) in at most $O[\theta(i) \log \theta(i)]$ steps (see Appendix B.2). The core routers

bear the responsibility to update the implicit costs q^l according to the simple dual update rule (3.10). After every K dual updates, the ingress router executes the primal update.

We have mentioned earlier that the solution of each local subproblem (3.8) does not require knowledge of the demand parameters λ_i and μ_i. Next, we show that the dual update can also be carried out using online measurement at each link, again without prior knowledge of the demand parameters of each class. We then obtain an *adaptive* algorithm that can track changes in the network conditions.

Note that in the dual gradient (3.9), $\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)}\frac{\lambda_i}{\mu_i}r_i p_{ij} H_{ij}^l$ is the average load per unit time at link l. This motivates us to estimate the gradient as follows: over a certain time window W, each link l collects the information of flow connection requests *from all classes* that arrive at the link. Let w be the total number of flow arrivals during W. Let $r_k, T_k, k = 1, ...w$ denote the bandwidth requirement and the service time, respectively, of the k-th arrival. (This information can be carried along with the connection requests.) Then we can use

$$G_t = R^l - \frac{\sum_{k=1}^{w} r_k T_k}{W} \tag{3.11}$$

to estimate the gradient. The interpretation is immediate: $\sum_{k=1}^{w} r_k T_k$ is the total amount of load brought to link l. One can verify that this estimate is unbiased, i.e., $\mathbf{E}[G_t] = \partial D/\partial q^l$. We can then update the implicit costs by

$$q^l(t+1) = \left[q^l(t) + \alpha^l\left(\frac{\sum_{k=1}^{w} r_k T_k}{W} - R^l\right)\right]^+. \tag{3.12}$$

When W is not large, the stepsize α^l has to be small to "average out" the noise in the estimate. This algorithm has the flavor of *stochastic approximation algorithms* [66] that have been used in many engineering problems. Our simulations with this algorithm demonstrate good convergence properties when a small fixed stepsize is used. That is, according to the simulations, the stochastic approximation algorithm converges to a small neighborhood of the solution to the upper bound. Further, when the stepsize α^l is away from zero, our algorithm can track the nonstationary behavior

of the network. As the demand (i.e., λ_i, μ_i) changes, it is reflected in the gradient estimate G_t. The network will then move towards the new optimal operating point.

3.3.3 How to Generate Alternate Paths

The set of alternate paths, denoted by the matrix $[H_{ij}^l]$, could potentially be the enumeration of all possible paths for each class. In practice, however, a much smaller set of alternate paths suffices. Maintaining this set of alternate paths is the role of the path-finding component in Fig. 3.1. There are several options to generate the candidate paths.

Option 1: Use paths that appear to be "heuristically good." For example, given a source-destination pair, we can use the set of minimum-hop paths, or, paths whose number of hops is no greater than h plus that of the minimum-hop path. Obviously, h should be small to avoid an explosion in the number of candidate paths.

Option 2: A better approach is to discover new paths online. The implicit costs q^l, which arise naturally as the Lagrangian Multipliers of the dual problem, give us guidelines on discovering potentially better alternate paths. Given a configuration of the alternate paths, we can easily verify the following properties that characterize any stationary point $(\vec{p^*}, \vec{q^*})$ of algorithm $\mathcal{A}$ (see Appendix B.3).

Properties of the Stationary Points:

1. When the utility functions are strictly concave, the admission probability $\sum_{j=1}^{\theta(i)} p_{ij}^*$ for each class i can be uniquely determined;

2. Only paths that have the minimum cost see positive routing probabilities. The cost of a path is the sum of the implicit costs for all links along the path. Hence, if we let $q_{i,0}$ denote the minimum cost among all alternate paths for class i, i.e., $q_{i,0} \triangleq \min_j \sum_{l=1}^{L} H_{ij}^l q^{l,*}$, then for all j,

$$p_{ij}^* > 0 \Rightarrow q_{ij}^* = q_{i,0}.$$

The above properties are consistent with the concept of the *minimum first derivative path* discussed in [64, p417]. Therefore, adding paths whose costs are larger than the minimum cost will not yield any gain. We can thus use the above properties to iteratively generate the candidate paths online. Starting from any initial set of candidate paths, we execute the distributed algorithm $\mathcal{A}$ to solve the upper bound. Then based on the implicit costs at the (possibly approximate) stationary point, we can run *any* minimal cost routing algorithm using the implicit costs as the cost metric for each link. If the minimal cost is smaller than the minimal cost among the current set of candidate paths by a certain threshold, we add this new path into the set, and continue. Otherwise, we can conclude that no further alternate paths need to be added.

Option 3: Use historical data. This can be viewed as a traffic engineering step. We first take measurements of typical traffic demands at different times of the day. For each demand pattern, we can use the above procedure in Option 2 *offline* to find the optimal alternate paths. The union of the alternate paths under all demand patterns can then be used as the set of candidate paths. The role of the distributed algorithm $\mathcal{A}$ is to shift the traffic load among these candidate paths automatically as the network condition changes.

3.3.4 Extensions to Multiple QoS Constraints

So far we have assumed that the bandwidth constraint is the only QoS constraint. We now address the extension to multiple QoS metrics and constraints. We can argue that link-state metrics other than the available bandwidth (e.g., delay and overflow probabilities, etc.) could be more stable in future high-bandwidth networks. When the link capacity of the network is large, the network can support a large number of connections at the same time. Due to the complexity in maintaining per-flow information, Quality of Service is likely to be provisioned on an aggregate basis. Each node in the network will provide a QoS guarantee on delay and/or packet loss

probabilities *for all flows* belonging to the same class, rather than *for each individual flow*. Such guarantees will stay unchanged as new flows arrive at or old flows depart from the network.

Let each class be given some QoS requirements on *both* the bandwidth constraint *and* some other constraints such as delay or packet loss probabilities. We now assume that each link will provision certain QoS guarantees on these other QoS metrics. Such guarantees are constant over time and can be advertised to the entire network. The alternate paths for each class must now be constrained to those that satisfy these other QoS requirements. Given a set of alternate paths, the distributed algorithm in Section 3.3.1 can be used unchanged to find the optimal routing probabilities. In order to generate the alternate paths, we can use the options in Section 3.3.3, except that now we have to consider other constraints too. For example, in Option 2, we can still use the implicit cost as the cost metric for each link and execute any *constrained minimal cost* QoS routing algorithm to search for new alternate paths.

It is important to note that the path-finding step does not deal with the available bandwidth constraint directly. Instead, it is based on the implicit cost, which is a more stable parameter that depends on the average congestion level of the network. Hence, the path-finding step can be carried out infrequently. Note that the computation of optimal paths under multiple QoS constraints is usually a NP-complete problem. Hence, for any practical implementation of QoS routing solutions, the computation overhead has always been a key issue. Our optimization based approach does not directly reduce the computational complexity. Rather, it reduces the *frequency* of the computation. We emphasize that *the optimal performance is still preserved even though computation becomes infrequent*. This, as mentioned in the Introduction, is again due to the separation of control time-scales: the set of candidate paths needs to change only when the *average* demand and capacity of the network changes significantly. Hence, the intensive computations only need to be carried out infrequently.

3.4 Implementational Issues

In this section we address some implementational issues.

3.4.1 Communicating the Implicit Costs

The distributed algorithm requires communicating the implicit costs back to the ingress routers. There are two alternatives. One is to use the connection request packets sent by the ingress router. Each link can insert its own implicit costs when processing the connection request packets. When the response is sent back to the ingress router, the implicit costs are piggy-backed for free. The other approach is to periodically advertise the implicit costs throughout the network. In the latter case, even when the implicit costs are updated infrequently, while the speed of convergence of the distributed algorithm will be affected, the optimal routing probabilities that the algorithm converges to will remain the same.

3.4.2 Gradient Estimates at the Link Algorithms

For the link algorithm, the gradient estimate in (3.11) requires the information from all flow arrivals, including those that could have been rejected by the upstream links. In some network systems, once an intermediate link along the path rejects a connection request, the request will not be passed on to downstream links. Let $\mathbf{P}_{i,j}^{B,l}$ be the probability that a connection request of class i routed to path j is rejected by links that are upstream to link l. The true connection arrival rate of class i at a link l will be $\lambda_i p_{ij}(1-\mathbf{P}_{i,j}^{B,l})$. In this case, the gradient estimate constructed in (3.11), by counting only actual arrivals, will be biased. However, when the system is large, this error will be small. This is due to two factors: Firstly, as long as the load at each link is less than or equal to 1, $\mathbf{P}_{i,j}^{B,l}$ will be close to zero (see Lemma 2.2.1 in Chapter 2); Secondly, if some links have load greater than 1, the implicit costs at these links will

be increased until the loads become less than or equal to 1. Therefore, in the end $\mathbf{P}_{i,j}^{B,l}$ will be close to zero and will have a minimal impact on the gradient estimate.

The gradient estimate in (3.11) needs knowledge of the service time T_k of an incoming flow. When this information is not known at the time of connection arrival, it can also be replaced by the time average of the service time of past flows. This time average can be calculated at the ingress router by measuring the service time of the flows that have completed service. The unbiasness of (3.11) is not affected by such changes.

3.4.3 Adaptive Stepsizes

The transient behavior of the distributed algorithm is sensitive to the choice of the stepsize α^l. A smaller stepsize will result in a smaller *misadjustment* (overshoot or undershoot) around the optimal solution, but takes a longer time to converge. A larger stepsize expedites the convergence at the cost of larger misadjustment. This tradeoff between misadjustment and speed of convergence is a fundamental one for stochastic approximation algorithms with constant stepsizes. A better approach is to use an adaptive stepsize scheme: a larger stepsize is used initially (or when sudden changes occur) to expedite convergence, followed by a smaller stepsize to reduce the misadjustment. This idea of stepsize adaptation has been used in many other applications, especially in adaptive filtering. Here we illustrate one such approach, borrowed from the idea in [67]:

Fix a link l. Let G_t be the estimate of the gradient at the t-th iteration. Let E_t be a weighted average of the past samples of G_t, i.e., upon a new sample G_t, let

$$E_{t+1} = \epsilon^l G_t + (1 - \epsilon^l) E_t,$$

where ϵ^l is a small positive constant. Let α_t^l denote the stepsizes at the t-th iteration. We can update the stepsize based on the correlation between E_t and G_t, i.e.,

$$\alpha_{t+1}^l = \min\{[\alpha_t^l + \beta^l E_t G_t]^+, \alpha_{\max}\}, \tag{3.13}$$

where β^l is a small positive constant, and $\alpha_{\max}$ is a maximum allowable stepsize chosen to ensure the stability of the system. We will demonstrate in the simulation results in Section 3.5 that the distributed algorithm with such an adaptive stepsize scheme can swiftly track the changes in the network condition, and it can also effectively reduce the misadjustment when the network condition is stable.

3.5 Simulation Results

In this section, we present simulation results that illustrate our optimization based approach for QoS routing. We implement the distributed algorithm following the online measurement based scheme in Section 3.3.2. The topologies we use are shown in Fig. 3.2. We first demonstrate the convergence of the distributed algorithm using the "triangle" network in Fig. 3.2. There are three classes of flows (AB, BC, CA). For each class of flows, there are two alternate paths, i.e., a direct one-link path, and an indirect two-link path. The arrival rates for classes AB, BC, CA are 1, 1 and 3 *flows per time unit*, respectively. Each flow consumes one *bandwidth unit* along the path(s) and holds the resources for a time that is exponentially distributed with mean of 100 units. Let the capacity of all links be 100 *bandwidth units*. For all classes the revenue v_i is 1 and the utility function is $U_i(p) = \ln p$. Both the revenue and the implicit cost are chosen to be unitless.

Fig. 3.3 demonstrates the evolution over time of the implicit costs at all links and the evolution of the routing probabilities of class CA. The x-axis corresponds to the total number of arrivals simulated. Readers can verify that all quantities of interest converge to a small neighborhood of the solution to the upper bound. The parameters we use for the distributed algorithm are: $\alpha^l = 0.0001$ *per bandwidth unit*, $\nu_i = 1$, $K = 1000$ and $W = 1$ *time unit*.

Fig. 3.4 demonstrates the convergence of the implicit costs when we use the adaptive stepsize scheme in Section 3.4. The parameters we use are: $\epsilon^l = 0.001$, $\alpha_{\max} = 0.1$ *per bandwidth unit*, $\beta^l = 0.0001$ *per cubic bandwidth unit* and $\alpha_0^l = 0$.

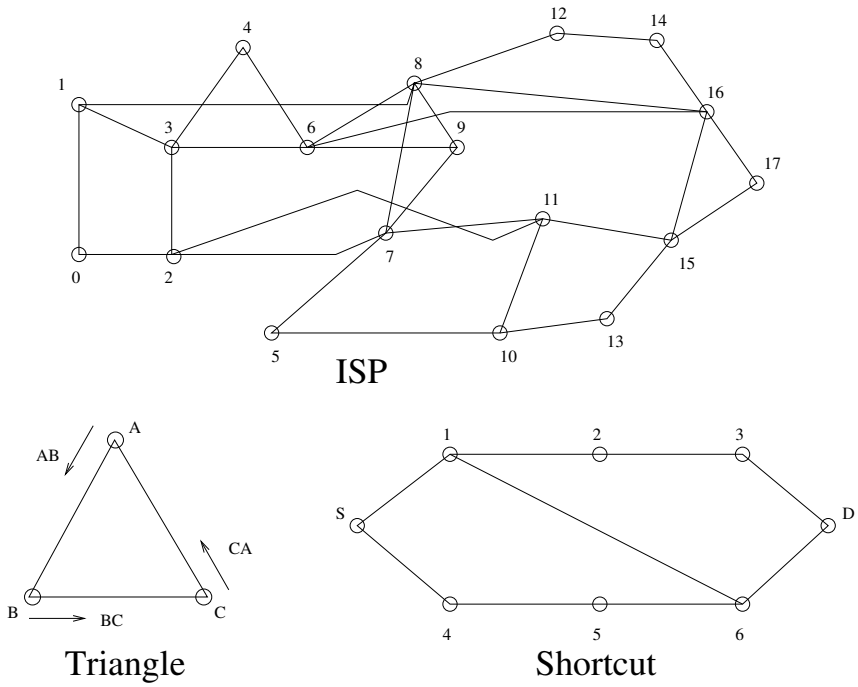


Fig. 3.2. Network topologies

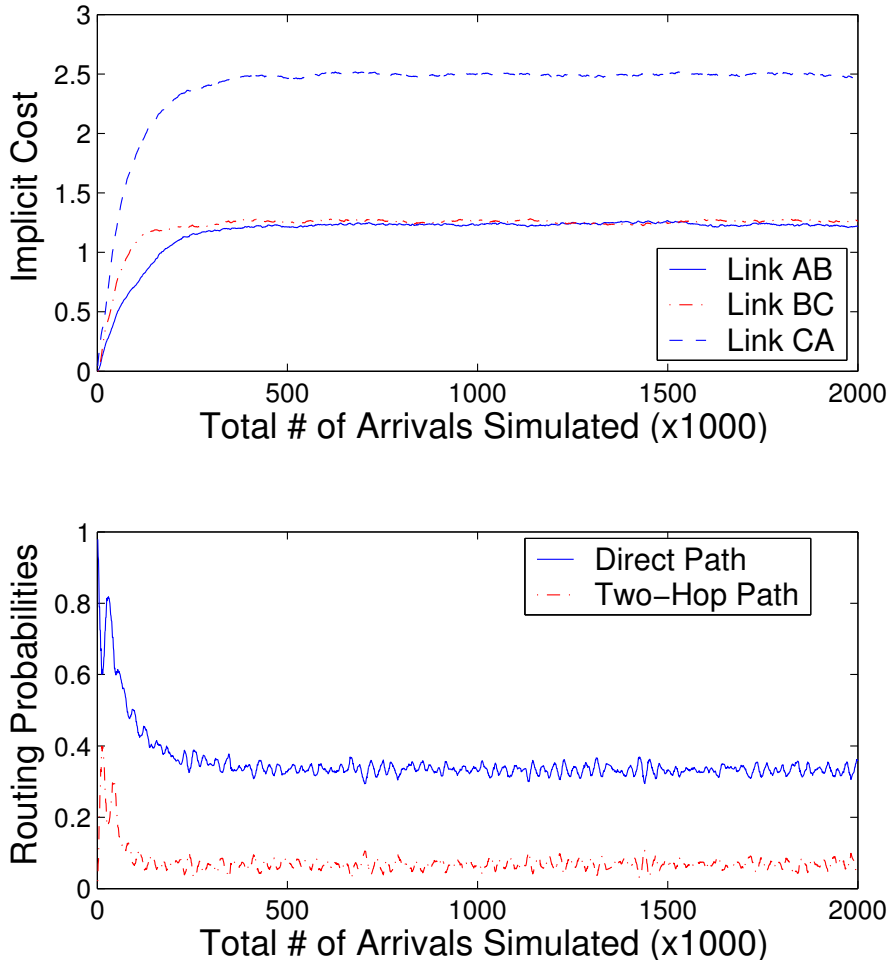


Fig. 3.3. Evolution of the implicit costs (top) and the routing probabilities of class CA (bottom) with respect to the number of arrivals simulated. The unit of x-axis is 1000 arrivals. The solution to the upper bound is the following: the implicit costs are 1.25, 1.25, and 2.5, respectively, for link AB, BC and CA. The routing probability for class CA are 0.33 for the direct path and 0.067 for the two-hop path.

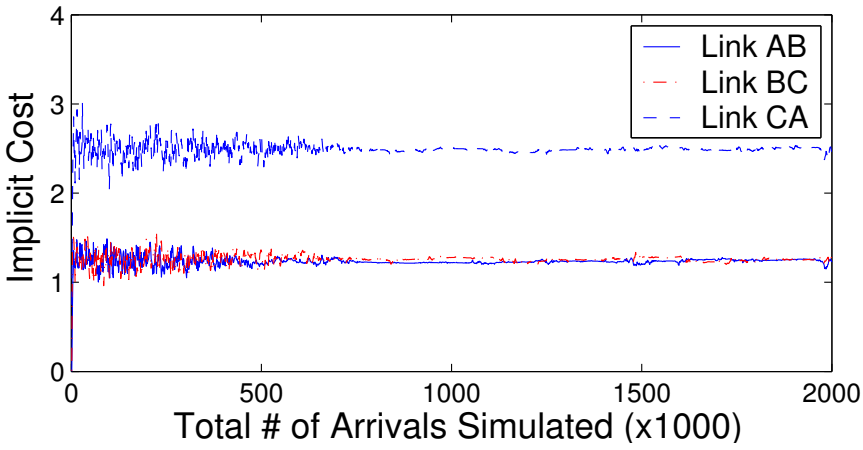


Fig. 3.4. Evolution of the implicit costs when the adaptive stepsize scheme is used.

The initial convergence is almost immediate: the implicit costs quickly jump to a small neighborhood of the solution to the upper bound, thanks to an increase in the stepsize initially. The evolution of the routing probabilities (not shown) follows the same trend. While the misadjustment takes time to die out (as the stepsize becomes smaller), Fig. 3.5 shows that the convergence of the revenue to its stationary value is achieved must faster (note that the range on the x-axis is smaller). As far as the overall revenue is concerned, the fluctuations of the implicit costs appear to cancel themselves out.

We have also simulated the case when the network condition changes over time, i.e., when the system is non-stationary. Fig. 3.6 and Fig. 3.7 demonstrate the evolution of the implicit costs when the average inter-arrival time of class CA changes according to a square wave and a triangle wave, respectively. We observe that the distributed algorithm with adaptive stepsizes can track the changes in the network condition swiftly.

We next simulate a larger network, i.e., the "ISP" topology in Fig. 3.2, which is reconstructed from an ISP network and has been used in many simulation studies [49–52, 56]. It has 18 nodes and 30 links. We simulate the case with a uniform

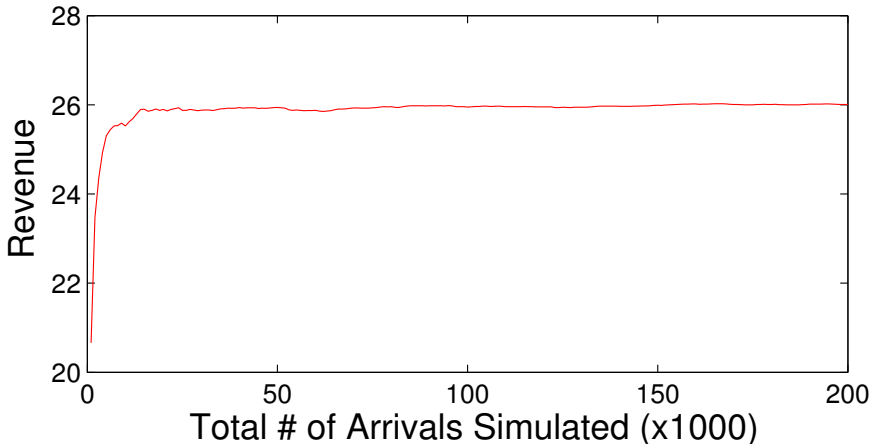


Fig. 3.5. Evolution of the average revenue when the adaptive stepsize scheme is used.

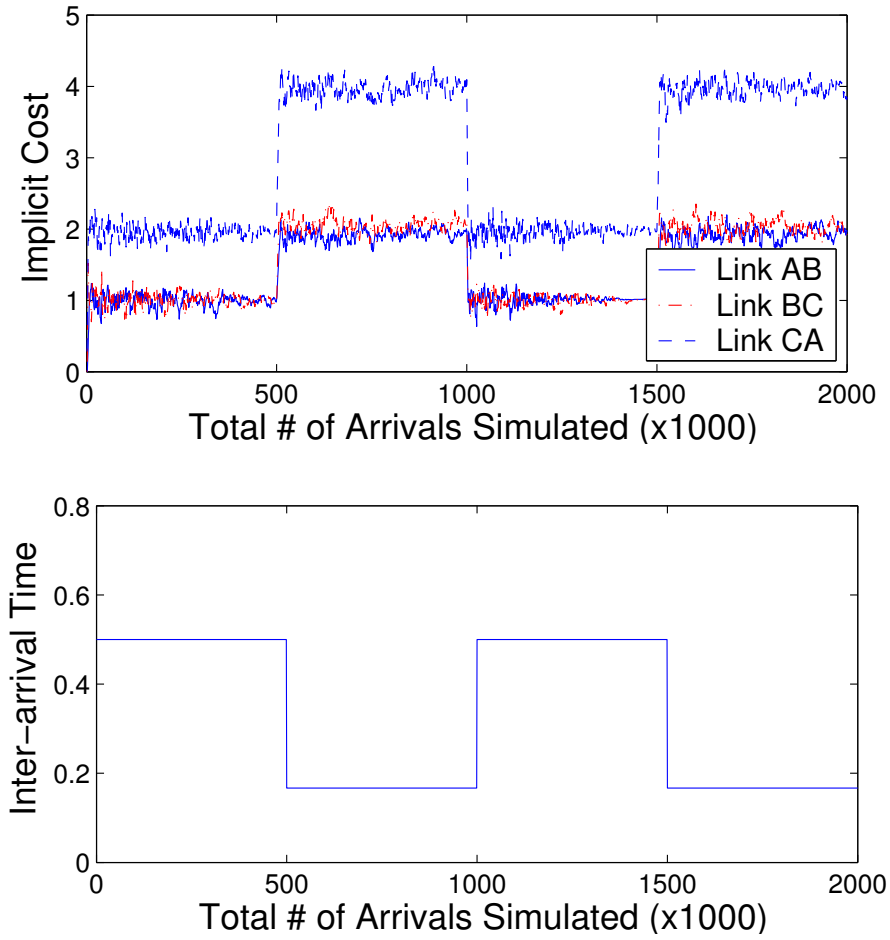


Fig. 3.6. Evolution of the implicit costs (top) when the average inter-arrival time of class CA changes according to a square wave (bottom).

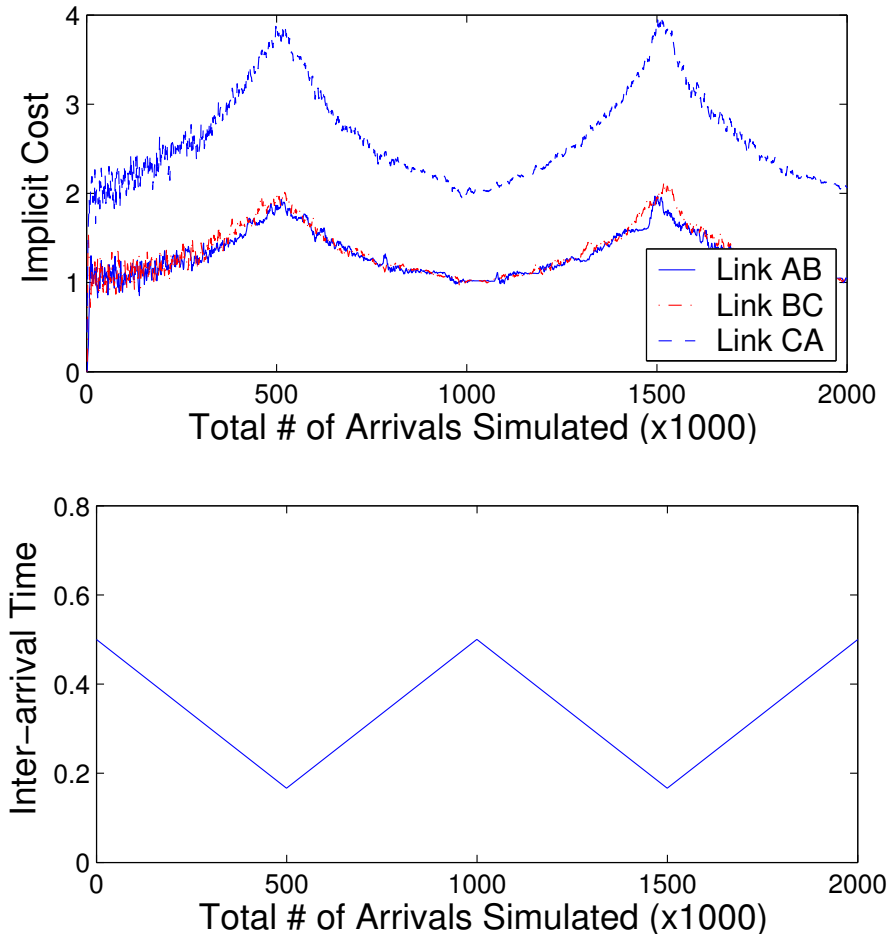


Fig. 3.7. Evolution of the implicit costs (top) when the average inter-arrival time of class CA changes according to a triangle wave (bottom).

demand matrix: flows arrive at each node according to a Poisson process with rate λ, and the destinations are chosen uniformly among all other nodes. The bandwidth requirement of each connection is one *bandwidth unit*. Revenue v_i is 1. We use a Pareto service time distribution with shape parameter 2.5, to capture the heavy-tailed characteristic of the traffics on the Internet. The mean service time is 100 *time units*. The capacity of each link is 1000 *bandwidth units*.

There are a total of $18 \times 17 = 306$ source-destination pairs (i.e., classes). When the simulation is initialized, the set of alternate paths for each source-destination pair consists of all minimum-hop paths. Once simulation starts, new paths can be added following Option 2 in Section 3.3.3. To simplify the simulation, we adopt an upper limit of 10 on the number of alternate paths for each source-destination pair: when a new path is found, if there are already 10 alternate paths, the old path with the smallest routing probability will be replaced by the new path.

We choose the utility function to be of the following form

$$U_i(p) = h_i \ln p - (h_i - 1)p,$$

where h_i is the minimal number of hops between source-destination pair i. This utility function improves the admission probability for flows that traverse a larger number of hops. (At the same level of admission probability $p < 1$, the marginal utility $\frac{dU_i}{dp} = h_i/p - (h_i - 1)$ is larger for flows that traverse a long path.)

We simulate the optimization based approach using the distributed algorithm and compare, in Fig. 3.8, the revenue and the total blocking probability over all classes against the values determined by the upper bound. We vary the per-node flow arrival rate λ from 1.0 to 10.0 *flows per time unit*. As we can see from these figures, our distributed algorithm tracks the upper bound consistently over all loads. With a network of this size (each link can hold 1000 flows) the difference between the upper bound and the simulation of our distributed algorithm is already small.

We also compare the performance of the Widest-Shortest-Path (WSP) algorithm. WSP has been used in many simulation studies [49,50,56]. Among all feasible paths, the WSP algorithm will first choose paths that have the smallest number of hops.

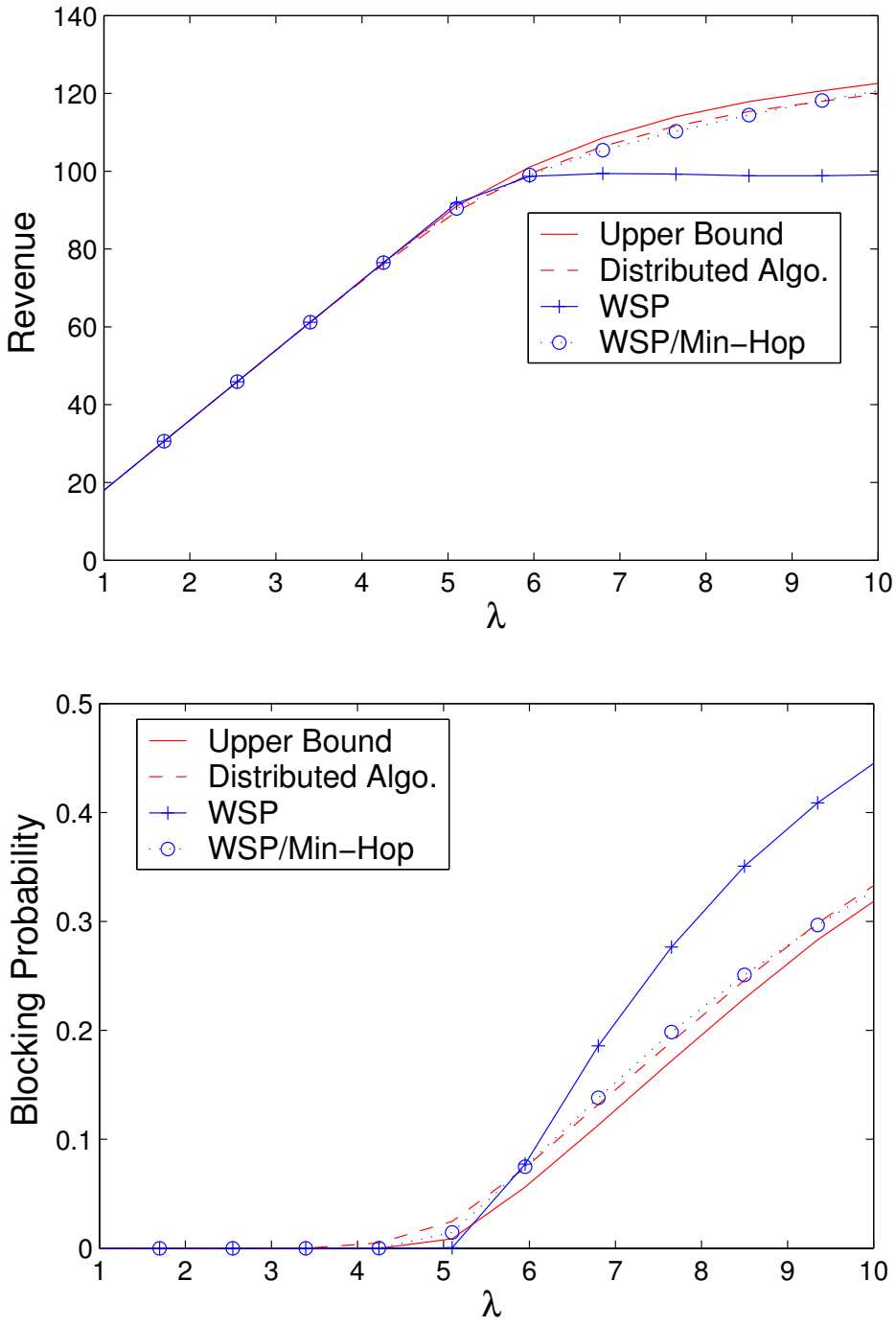


Fig. 3.8. The revenue (top) and the blocking probability (bottom) of the distributed algorithm compared with the upper bound and WSP.

If there are multiple such paths, the WSP algorithm will choose the one with the largest available bandwidth. However, as shown in Fig. 3.8, the performance of a faithful implementation of WSP starts to taper off at $\lambda = 5.0$ *flows per time unit*. The performance degradation of WSP is due to its selection of non-minimal hop paths, which could result in sub-optimal configurations for the whole network. If we constrain WSP to minimum-hop paths only, the performance degradation will disappear in this example, as shown by the curve labeled "WSP/Min-Hop." However, from this, we should not draw the conclusion that such a practice is always better. By constraining WSP to minimum-hop paths, one also reduces the capability of WSP to use other potentially less congested paths. The end result depends on the topology of the network and the demand pattern. For example, in the "shortcut" topology in Fig. 3.2, assume that the capacities of all links are the same. If flows from S to D is to only use the minimum-hop path (S-1-6-D), once this path is full, no more flows can be admitted. However, if the flows use the non-minimum-hop paths S-1-2-3-D and S-4-5-6-D, twice as many flows can be admitted. Hence it is not always better to restrict on minimum-hop paths.

Our distributed algorithm, on the other hand, will always be able to find the right balance by solving the upper bound. It *consistently* tracks the upper bound under all load conditions. This provable optimality is an attractive feature of our optimization based approach as it ensures that the routing decision will always be close to optimal.

The strength of the optimization based approach is even more evident when the computation and link state updates become infrequent. To show this, we pick $\lambda = 6.0$ *flows per time unit* and simulate both the distributed algorithm and the WSP (with minimum-hop path only) when we vary the interval between link-state updates. To ensure a fair comparison of the performance and the overhead of the two schemes, we adopt the following settings for our simulation. For the distributed algorithm, we choose to simulate the case where the implicit costs are advertised with each link state update, and computation is carried out after each link state update. (That

is, we are not simulating the "piggy-back" approach in Section 3.4.1.) For WSP, in contrast to the suggestion given in [51], we do not allow WSP to *recompute* paths when a connection routed to a precomputed path is later rejected. The reason is that one cannot reduce the computational overhead too much if such recomputation is allowed: for example, when the blocking probability is around 10%, on average 1 out of 10 arrivals will trigger recomputation! For a similar reason, we also do not use the *triggered* link state update strategy of [51] for WSP. When the triggered strategy is used, changes in available bandwidth that exceed certain percentage of the past advertised available bandwidth will trigger a new link state update. When the network operates at a high utilization level, the available bandwidth is small. Even small changes in available bandwidth will trigger frequent updates. Hence, one can not reduce the communication overhead too much using a triggered update strategy.

Simulation results are presented in Fig. 3.9. The performance of the distributed algorithm changes little as the link state update interval becomes larger and larger, while the performance of WSP decreases significantly. (The unit time on the x-axis is the mean inter-arrival time of flows at each node.) In the worst case, WSP blocks *twice as many connections* compared to the case when it has perfect link states. We have also simulated the case when the network condition changes over time (i.e., when the system is non-stationary). In Fig. 3.10, we change the average inter-arrival time at each node according to the triangle wave in Fig. 3.10(b), and plot the overall blocking probability in Fig. 3.10(a) when we vary the interval between link-state updates. The performance of the distributed algorithm is again insensitive to the link state update interval, while the performance of WSP decreases significantly as the link state update interval increases. Note that the exact level of this performance degradation for WSP is a complex function that depends on many factors, such as the topology and the demand of the network, etc. Again, the strength of the optimization based approach is that it consistently achieves near optimal performance, even when the computation and communication overhead are greatly reduced.

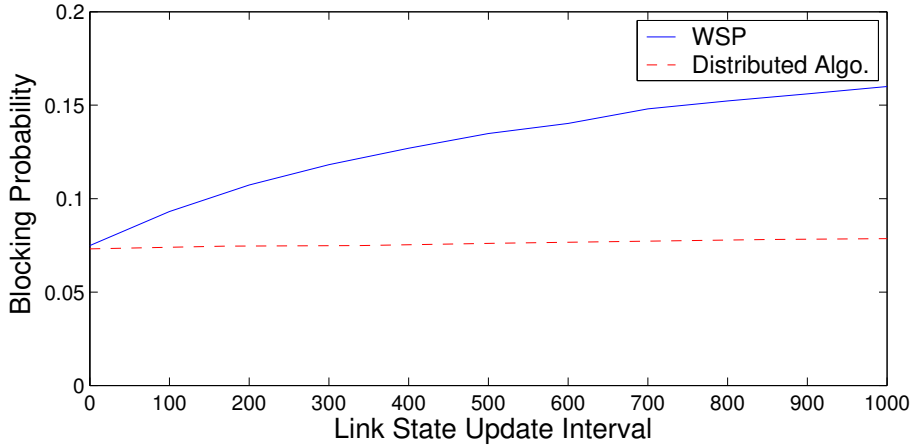


Fig. 3.9. The blocking probability as the link state update interval increases. The unit on the x-axis is the mean inter-arrival time of flows at each node. The arrival rate at each node is fixed at $\lambda = 6.0$ *flows per unit time*.

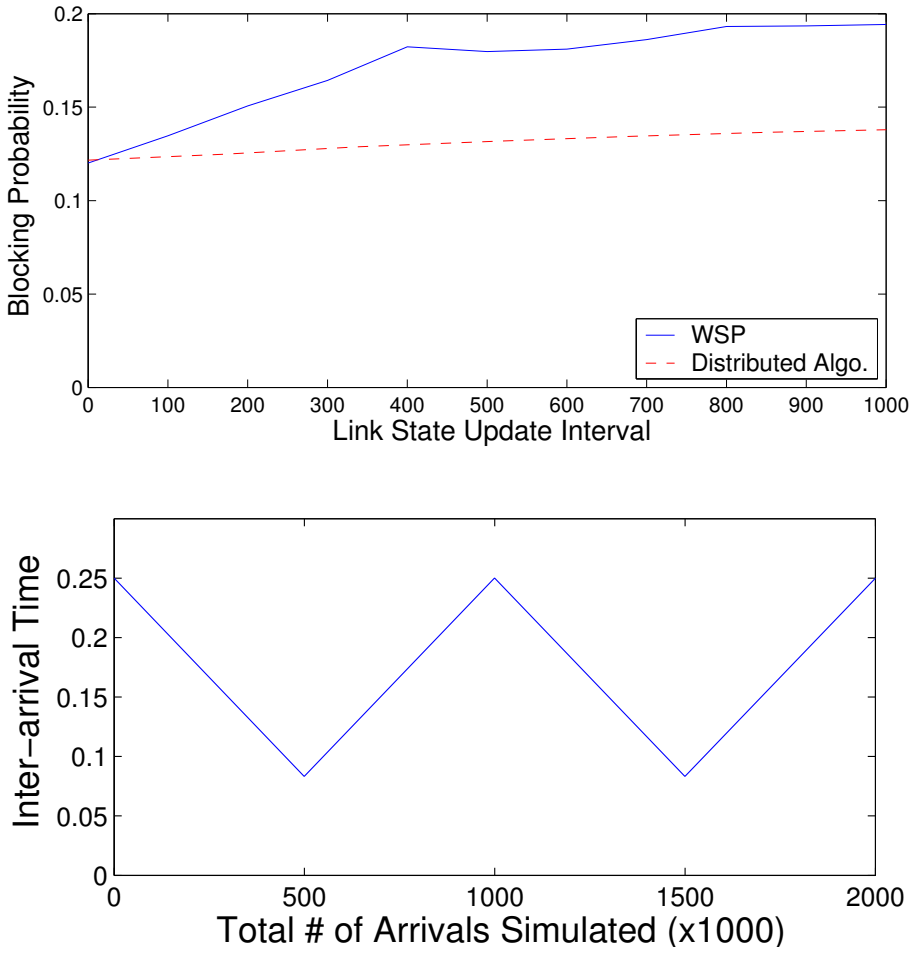


Fig. 3.10. The blocking probability versus the link state update interval (top) when the average inter-arrival time at each node changes according to the triangle wave (bottom). The unit on the x-axis of Figure (a) is the mean inter-arrival time of flows at each node. The unit on the x-axis of Figure (b) corresponds to 1000 total arrivals simulated. The average arrival rate at each node is $\lambda = 8.0$ *flows per unit time*.

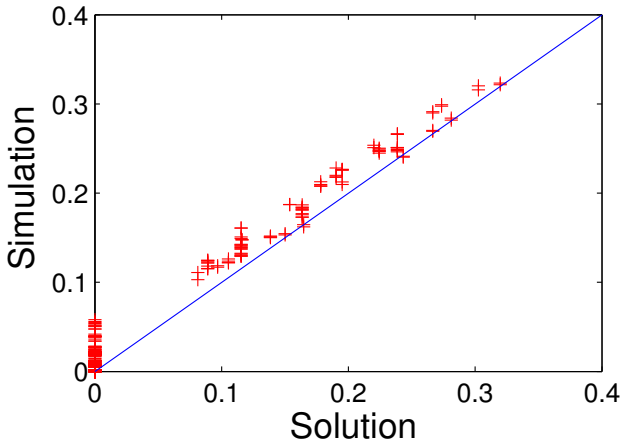


Fig. 3.11. The blocking probability predicted by the upper bound compared with that collected from the simulation of the distributed algorithm. The arrival rate at each node is fixed at $\lambda = 6.0$ *flows per time unit*.

When our optimization based approach to QoS routing is used, designers can predict the operating point of the network by analytically solving the upper bound. This is shown in Fig. 3.11 where each point represents the blocking probability of one source-destination pair computed by the upper bound (along the x-axis) and that collected from the simulation of the distributed algorithm (along the y-axis). The points follow the diagonal line, which indicates that the simulation matchs the theory. In contrast, the analysis of dynamic QoS routing schemes (such as WSP) appears to be an intractable problem, especially when the computation becomes infrequent and the link state information becomes inaccurate. One usually has to resort to simulation to find out the operation of a QoS routing algorithm.

3.6 Conclusion and Future Work

In this chapter, we developed an optimization based approach for Quality of Service routing in high-bandwidth networks. We view a network that employs QoS routing as an entity that carries out a distributed optimization. By solving the op-

timization problem, the network is driven to an efficient operating point. When the capacity of the network is large, this optimization takes on a simple form. We develop a distributed and adaptive algorithm that can efficiently solve the optimization online. The proposed optimization based approach has several advantages in reducing the computation and communication overhead, and in improving the predictability and controllability of the operating characteristics of the network.

We now briefly outline directions for future work: (1) In this chapter we propose to update the implicit costs by measuring the arrived load. Other methods are possible, for example, by taking into account the utilization levels of the links. (2) A deeper understanding of the transient behavior of the distributed algorithm is important. The adaptive stepsize scheme in Section 3.4 that improves the speed of the convergence is of particular interest. (3) We assume that the capacity of the network is uniformly large. If some part of the network is not so large (for example, at the network edge), one then has to study a finer level of dynamics in these parts of the network. It would be interesting to study hybrid schemes that combine our results with some further details of the dynamics of smaller links. (4) In this chapter we take a source routing model. Adapting our result to the distributed routing or hierarchical routing paradigms is also a possible direction for future work. A related issue is how to deal with the case when routers do not allow arbitrary splitting of traffic among multiple paths. (5) Finally, from a theoretical viewpoint, it would be important to prove the convergence of the distributed algorithm under more general settings, such as with asynchronous computation.

4. CONVERGENCE PROPERTIES OF ALGORITHM A FOR SOLVING THE MULTI-PATH UTILITY MAXIMIZATION PROBLEM

4.1 Introduction

This chapter is dedicated to the study of the convergence properties of Algorithm $\mathcal{A}$ that was proposed in Chapter 3. Since these results will also be of interest to problems other than QoS routing, we will carry out our analyses for the following problem with a more general form:

$$\max_{x_{ij}\geq 0, m_i \leq \sum_{j=1}^{\theta(i)} x_{ij} \leq M_i, i=1,...,I} \quad \sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij}) \tag{4.1}$$

$$\text{subject to} \quad \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l} x_{ij} \leq R^{l} \quad \text{for all } l = 1, ..., L. \tag{4.2}$$

Optimization problems of this form appear in several resource allocation problems in communication networks, when each user or (class of users) can have multiple alternative paths through the network [22]. Generically, problem (4.1) amounts to allocating resources $R^1, ..., R^L$ from network components $l = 1, 2, ..., L$ to users $i = 1, 2, ..., I$ such that the total system "utility" is maximized. The "utility" function $f_i(\cdot)$ represents the performance, or level of "satisfaction," of user i when a certain amount of resource is allocated to it. In practice, this performance measure can be in terms of revenue, welfare, or admission probability, etc., depending on the problem setting. We assume throughout that $f_i(\cdot)$ is concave. Each user i can have $\theta(i)$ alternative paths (a *path* consists of a subset of the network components). Let x_{ij} denote the amount of resources allocated to user i on path j. Then the

utility $f_i(\sum_{j=1}^{\theta(i)} x_{ij})$, subject to $m_i \leq \sum_{j=1}^{\theta(i)} x_{ij} \leq M_i$, is a function of the *sum* of the resources allocated to user i on all paths. Hence, the resources on alternative paths are considered equivalent and interchangeable for user i. The constants E_{ij}^l represent the routing structure of the network: each unit of resource allocated to user i on path j will consume E_{ij}^l units of resource on network component l. ($E_{ij}^l = 0$ for network components that are not on path j of user i.) The inequalities in (4.2) represent the resource constraints at the network components (hence R^l can be viewed as the capacity of network component l, and $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}$ is the total amount of resources consumed at network component l summed over all users and all alternative paths). We assume that $R^l > 0$, $E_{ij}^l \geq 0$, $m_i \geq 0$ and $M_i > 0$ (M_i could possibly be $+\infty$).

Remark: Note that for the model in Chapter 3, x_{ij} corresponds to the routing probability p_{ij}, $E_{ij}^l = \frac{\lambda_i}{\mu_i} r_i H_{ij}^l$, and $f_i(\cdot) = \frac{\lambda_i}{\mu_i} v_i U_i(\cdot)$.

We will refer to problem (4.1) as the *multi-path utility maximization problem* [22]. In this chapter, we are interested in solutions to this problem that are amenable to *online* implementation. There exist several resource allocation problems in communication networks that can be modeled as (4.1), including the optimal pricing problem in Chapter 2, the optimal QoS routing problem in Chapter 3, and the multi-path flow control problem [14, 58–60]. Essentially, once the network can support multi-path routing, the resource allocation problem changes from a *single-path* utility maximization problem to a *multi-path* utility maximization problem. As we will soon see, the multi-path nature of the problem leads to several difficulties in constructing solutions suitable for online implementation. One of the main difficulties is that, once some users have multiple alternative paths, the objective function of problem (4.1) is no longer strictly concave, and hence the dual of the problem may not be differentiable at every point. Note that this lack of strict concavity is mainly due to the linearity $\sum_{j=1}^{\theta(i)} x_{ij}$. (The objective function in (4.1) is still not strictly concave even if the utility functions f_i are strictly concave.) On the other hand, the requirement that the solutions must be implementable online also imposes a number

of important restrictions on our design space, some of which were already discussed in Chapter 3. We outline these restrictions below:

- The solution has to be distributed because these communication networks can be very large and centralized solutions are not scalable.

- In order to lower the communication overhead, the solution has to limit the amount of information exchanged between the users and different network components. For example, a solution that can adjust resource allocation based on online measurements is preferable to one that requires explicit signaling mechanisms to communicate information.

- It is also important that the solution does not require the network components to store and maintain per-user information (or *per-flow* information, as it is referred to in some of the networking literature). Since the number of users sharing a network component can be large, solutions that require maintaining per-user information will be costly and will not scale to large networks.

- In the case where the solution uses online measurements to adjust the resource allocation, the solution should also be resilient to measurement noise due to estimation errors.

In this chapter, we first re-derive the corresponding form of Algorithm $\mathcal{A}$ as a distributed solution for the multi-path utility maximization problem (4.1). We will then show that our distributed solution has the aforementioned attributes desirable for online implementation. Our main technical contributions are as follows:

1) We provide a rigorous analysis of the convergence of our distributed algorithm. This analysis is done without requiring the *two-level convergence structure* that is typical in standard techniques in the convex programming literature for dealing with the lack of strict concavity of the problem. Note that algorithms based on these standard techniques are required to have an *outer level* of iterations where each outer iteration consists of an *inner level* of iterations. For the convergence of this class

of algorithms to hold, the inner level of iterations must converge before each outer iteration can proceed. Such a two-level convergence structure may be acceptable for off-line computation, but not suitable for online implementation because in practice it is difficult for the network to decide in a distributive fashion when the inner level of iterations can stop. A main contribution of this work is to establish the convergence of our distributed algorithm without requiring such a two-level convergence structure.

2) By proving convergence, we are able to provide easy-to-verify bounds on the parameters (i.e., step-sizes) of our algorithm to ensure convergence. Note that when distributed algorithms based on our solution are implemented online, a practically important question is how to choose the parameters of the algorithm to ensure efficient network control. Roughly speaking, the step-sizes used in the algorithm should be small enough to ensure stability and convergence, but not so small such that the convergence becomes unnecessarily slow. The main part of this work addresses the question of parameter selection by providing a rigorous analysis of the convergence of the distributed algorithm.

3) We also study the convergence of the algorithm in the presense of measurement noise and provide guidelines on how to choose the step-sizes to reduce the disturbance in the resource allocation due to noise.

4) Our studies reveal how the inherent multi-path nature of the problem can potentially lead to such difficulties as instability and oscillation, and how these difficulties should be addressed by the selection of the parameters in the distributed algorithm.

We believe that these results and insights are important for network designers who face these types of resource allocation problems. The rest of the chapter are organized as follows. After discussing related work in Section 4.1.1, we will first re-derive our distributed solution (i.e., Algorithm $\mathcal{A}$) in Section 4.2. The convergence of the distributed algorithm will be studied in Sections 4.3 when there is no measurement noise in the system, and in Section 4.4 when there is measurement noise. Simulation results are presented in Section 4.5. Then we conclude.

4.1.1 Related Work

The *single-path* utility maximization problem, i.e., when each user (or class) has only one path, has been extensively studied in the past, mainly in the context of Internet flow control (see, for example, [14–16, 41, 61] and the reference therein). However, the *multi-path* utility maximization problem has received less attention in the literature [14, 58–60]. In [14], after studying the single-path utility maximization problem, the authors briefly discuss the extension to the multi-path case. They categorize the solutions into primal algorithms and dual algorithms. Global convergence of the primal algorithms is studied in [14] for the case when feedback delays are negligible. (On the other hand, the dual algorithms there have an oscillation problem as we will discuss soon). Local stability of primal algorithms with feedback delays is further studied in [60]. Since [14] and [60] use a penalty-function approach, in general the algorithms there can only produce *approximate* solutions to the original problem (4.1).

The method in [59] can be viewed as an extension of the primal algorithms in [14] for computing *exact* solutions to problem (4.1). It employs a (discontinuous) binary feedback mechanism from the network components to the users: each network component will send a feedback signal of 1 when the total amount of resources consumed at the network component is greater than its capacity, and it sends a feedback signal of 0 otherwise. The authors of [59] show that, if each network component can measure the total amount of consumed resources *precisely*, their algorithm will converge to the *exact* optimal solution of problem (4.1). However, their algorithm will not work properly *in the presence of measurement noise*: if the network component can only estimate the amount of consumed resources with some random error (we will see in Section 4.2.1 how such situations arise), it could send a feedback signal of 1 erroneously even if the true amount of resources consumed is less than its capacity (or, a feedback signal of 0 even if the true amount of resources consumed exceeds its capacity). Therefore, the algorithm in [59] cannot produce

the exact optimal solution when there is measurement noise. The *AVQ* algorithm [41, 68] is also worth mentioning as an extension of the primal algorithms in [14] for computing *exact* solutions. However, the literature on the *AVQ* algorithm has focused on the single-path case. The extension to the multi-path case has not been rigorously studied.

Dual algorithms that can produce *exact* solutions are developed in [15] for the single-path case. When extended to the multi-path case, both this algorithm and the dual algorithm in [14] share the same *oscillation problem*. That is, although the dual variables in their algorithms may converge, the more meaningful primal variables (i.e., the resource allocation x_{ij}) will not converge. (We will illustrate this problem further in Section 4.2.) This difficulty arises mainly because the objective function of problem (4.1) is not *strictly concave* in the primal variables x_{ij} once some users have multiple paths. The authors in [58] attempt to address the oscillation problem by adding a quadratic term onto the objective function. Their approach bears some similarities to the idea that we use in this chapter. However, they do not provide rigorous proofs of convergence for their algorithms.

Another method that is standard in convex programming for dealing with the lack of strict concavity is the Alternate Direction Method of Multipliers (ADMM) [64, p249, P253]. It has known convergence property (when there is no measurement noise) and can also be implemented in a distributed fashion. However, when implemented in a network setting, the ADMM algorithm requires substantial communication overhead. At each iteration, the ADMM algorithm requires that each network component divides the amount of unallocated capacity equally among all users sharing the network component and communicates the share back to each user. Each user not only needs to know the cost of each path (as in the distributed algorithm we will propose in this chapter), but also needs to know its share of unallocated capacity at each network component. Further, in a practical network scenario where the set of active users in the system keep changing, unless the network has a reliable signaling mechanism, even keeping track of the current number of active users in the

system requires maintaining per-user information. It is also unclear how the ADMM algorithm would behave in the presence of measurement noise. In this chapter, we will study new solutions that are specifically designed for online implementation, and that do not require each network component to store and maintain per-user information.

4.2 The Distributed Algorithm

Similar to what we found in Chapter 3, one of the main difficulties in solving (4.1) is that, once some users have multiple alternative paths, the objective function of (4.1) is not strictly concave. As we go into the details of the analysis, we will see the manifestation of this difficulty in different aspects. At a high level, since the primal problem is not strictly concave, the dual problem may not be differentiable at every point. In this chapter, we would still like to use a duality based approach (parallel to the approach in Chapter 3), because the dual problem usually has simpler constraints and is easily decomposable. Again, to circumvent the difficulty due to the lack of strict concavity, we use ideas from Proximal Optimization Algorithms [64, p232]. The idea is to add a quadratic term to the objective function. Let $\vec{x}_i = [x_{ij}, j = 1, ..., \theta(i)]$ and

$$C_i = \{\vec{x}_i | x_{ij} \geq 0 \text{ for all } j \text{ and } \sum_{j=1}^{\theta(i)} x_{ij} \in [m_i, M_i]\}, \quad i = 1, ..., I. \tag{4.3}$$

Let $\vec{x} = [\vec{x}_1, ..., \vec{x}_I]^T$ and let C denote the Cartesian product of C_i, i.e., $C = \bigotimes_{i=1}^{I} C_i$. We now introduce an auxiliary variable y_{ij} for each x_{ij}. Let $\vec{y}_i = [y_{ij}, j = 1, ..., \theta(i)]$ and $\vec{y} = [\vec{y}_1, .., \vec{y}_I]^T$. The optimization then becomes:

$$\begin{aligned} \max_{\vec{x}\in C, \vec{y}\in C} \quad & \sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij}) \quad - \quad \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij})^2 \\ \text{subject to} \quad & \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij} \leq R^l \quad \text{for all } l, \end{aligned} \tag{4.4}$$

where c_i is a positive number chosen for each i. It is easy to show that the optimal value of (4.4) coincides with that of (4.1). In fact, Let $\vec{x^*}$ denote the maximizer of

(4.1), then $\vec{x} = \vec{x^*}, \vec{y} = \vec{x^*}$ is the maximizer of (4.4). Note that although a maximizer of (4.1) always exists, it is usually not unique since the objective function is not strictly concave.

The standard Proximal Optimization Algorithm then proceeds as follows:

Algorithm $\mathcal{P}$:

At the t-th iteration,

P1) Fix $\vec{y} = \vec{y}(t)$ and maximize the augmented objective function with respect to $\vec{x}$. To be precise, this step solves:

$$\begin{aligned} \max_{\vec{x} \in C} \quad & \sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij}) \quad - \quad \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij})^2 \qquad (4.5) \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij} \leq R^l \quad \text{for all } l. \end{aligned}$$

Note that the maximization is taken over $\vec{x}$ only. With the addition of the quadratic term $\sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij})^2$, for any fixed $\vec{y}$, the primal objective function is now strictly concave with respect to $\vec{x}$. Hence, the maximizer of (4.5) exists and is unique. Let $\vec{x}(t)$ be the solution to this optimization.

P2) Set $\vec{y}(t+1) = \vec{x}(t)$.

It is easy to show that such iterations will converge to the optimal solution of problem (4.1) as $t \to \infty$ [64, p233].

Step P1 still needs to solve a global non-linear programming problem at each iteration. Since the objective function in (4.5) is now strictly concave, we can use standard duality techniques. Let $q^l, l = 1, ..., L$ be the Lagrange Multipliers for the constraints in (4.5). Let $\vec{q} = [q^1, ..., q^L]^T$. Define the Lagrangian as:

$$\begin{aligned} L(\vec{x}, \vec{q}, \vec{y}) &= \sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij}) - \sum_{l=1}^{L} q^l(\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij} - R^l) - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij})^2 \\ &= \sum_{i=1}^{I} \left\{ f_i(\sum_{j=1}^{\theta(i)} x_{ij}) - \sum_{j=1}^{\theta(i)} x_{ij} \sum_{l=1}^{L} E_{ij}^l q^l - \sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij})^2 \right\} + \sum_{l=1}^{L} q^l R^l. \end{aligned} \qquad (4.6)$$

Let $q_{ij} = \sum_{l=1}^{L} E_{ij}^l q^l$, $\vec{q}_i = [q_{ij}, j = 1, ..., \theta(i)]$. The objective function of the dual problem is then:

$$D(\vec{q}, \vec{y}) = \max_{\vec{x} \in C} L(\vec{x}, \vec{q}, \vec{y}) = \sum_{i=1}^{I} B_i(\vec{q}_i, \vec{y}_i) + \sum_{l=1}^{L} q^l R^l, \tag{4.7}$$

where

$$B_i(\vec{q}_i, \vec{y}_i) = \max_{\vec{x}_i \in C_i} \left\{ f_i(\sum_{j=1}^{\theta(i)} x_{ij}) - \sum_{j=1}^{\theta(i)} x_{ij} q_{ij} - \sum_{j=1}^{\theta(i)} \frac{c_i}{2} (x_{ij} - y_{ij})^2 \right\}. \tag{4.8}$$

The dual problem of (4.5), given $\vec{y}$, then corresponds to minimizing D over the dual variables $\vec{q}$, i.e.,

$$\min_{\vec{q} \geq 0} D(\vec{q}, \vec{y}).$$

Since the objective function of the primal problem (4.5) is strictly concave, the dual is always differentiable. Let $\vec{q} = \vec{q}(t')$. The gradient of D is

$$\frac{\partial D}{\partial q^l} = R^l - \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}^0(t'),$$

where $x_{ij}^0(t')$ solves (4.8) for $\vec{q} = \vec{q}(t')$. The step P1 can then be solved by gradient descent iterations on the dual variables $\vec{q}$, i.e.,

$$q^l(t'+1) = \left[q^l(t') + \alpha^l (\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}^0(t') - R^l) \right]^+, \tag{4.9}$$

where $[\cdot]^+$ denotes the projection to $[0, +\infty)$. It is again easy to show that, given $\vec{y}$, the dual update (4.9) will converge to the minimizer of $D(\vec{q}, \vec{y})$ as $t' \to \infty$, provided that the step-sizes α^l are sufficiently small [64, p214].

Remark (The Oscillation Problem Addressed): From (4.8) we can observe the potential *oscillation problem* caused by the multi-path nature of problem (4.1), and the crucial role played by the additional quadratic term in dampening this oscillation. Assume that there is no additional quadratic term, i.e., $c_i = 0$. Readers can verify that, when (4.8) is solved for any user i that has multiple alternative paths, only paths that have the least q_{ij} will have positive x_{ij}. That is, $q_{ij} > \min_k q_{ik} \Rightarrow x_{ij} = 0$.

This property can easily lead to oscillation of x_{ij} when the dual variables $\vec{q}$ are being updated. To see this, assume that a user i has two alternative paths, and the sum of the dual variables on these two paths, $q_{i,1} = \sum_{l=1}^{L} E_{i,1}^{l} q^l$ and $q_{i,2} = \sum_{l=1}^{L} E_{i,2}^{l} q^l$, are close to each other. At one time instant $q_{i,1}$ could be greater than $q_{i,2}$, in which case the maximum point of (4.8) satisfies $x_{i,1} = 0$ and $x_{i,2} > 0$. At the next time instant, since more resources are consumed on network components on path 2, the dual variables $\vec{q}$ could be updated such that $q_{i,2} > q_{i,1}$ (see the update equation (4.9)). In this case, the maximum point of (4.8) will require that $x_{i,1} > 0$ and $x_{i,2} = 0$, i.e., the resource allocation will move *entirely* from path 2 over to path 1. This kind of flip-floping can continue forever and is detrimental to network control. When $c_i > 0$, however, the maximum point $\vec{x}_i$ of (4.8) is continuous in $\vec{q}_i$ (shown later in Lemma 4.3.1). Hence, the quadratic term serves a crucial role to dampen the oscillation and stablize the system.

4.2.1 Towards Constructing Online Solutions

The algorithm $\mathcal{P}$ that we have constructed requires the *two-level convergence* structure typical in proximal optimization algorithms. The algorithm $\mathcal{P}$ consists of an outer level of iterations, i.e., iterations P1 and P2, where each outer iteration P1 consists of an inner level of iterations (4.9). For the convergence of algorithm $\mathcal{P}$ to hold, the inner level of iterations (4.9) must converge before each outer iteration can proceed from step P1 to P2. Such a two-level convergence structure is unsuitable for online implementation because in practice, it is difficult for the network elements to decide in a distributive fashion when the inner level of iterations should stop.

Despite this difficulty, the main building blocks (4.8) and (4.9) of algorithm $\mathcal{P}$ have several attractive attributes desirable for online implementation. In particular, all computation can be carried out based on local information, and hence can be easily distributed. More precisely, in the definition of the dual objective function $D(\vec{q}, \vec{y})$ in (4.7), we have decomposed the original problem into I separate subprob-

lems for each user $i = 1, ..., I$. Given $\vec{q}$, each subproblem B_i (4.8) can now be solved independently[1]. If we interpret q^l as the *implicit cost* per unit resource on network component l, then q_{ij} is the cost per unit resource on path j of user i. We can call q_{ij} the cost of path j of user i. Thus the costs $q_{ij}, j = 1, ..., \theta(i)$, capture all the information that each user i needs to know in order to determine its resource allocation x_{ij}. Further, according to (4.9), the implicit cost q^l can be updated at each network component l based on the difference between the capacity R^l and the aggregate load $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}^0(t')$. In many applications, this aggregate load can be measured by each network component directly. For example, in a multi-path flow control problem where x_{ij} represents the data rate of user (flow) i on path j [22], the aggregate load $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}$ is simply the aggregate data rate going through link l, which can be estimated by counting the total amount of data forwarded on the link over a certain time window. Hence, *no per-user information needs to be stored or maintained*. In some applications, there is yet another reason why the measurement-based approach is advantageous. That is, by measuring the aggregate load directly, the algorithm does not need to rely on prior knowledge of the parameters of the system, and hence can automatically adapt to the changes of these parameters. For example, in the optimal routing problem in Chapter 3, each link l needs to estimate the aggregate load $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l$. Since the probability that a user of class i is routed to path j is p_{ij}, the arrival process of users of class i on link l is a Poisson process with rate $\lambda_i p_{ij}$. Assume that neither the mean arrival rate λ_i nor the mean service time $1/\mu_i$ are known *a priori*, but each user knows its own service time in advance. Each link can then estimate the aggregate load as follows[2]: over a certain time window W, each link l collects the information of the arriving users *from all classes* to link l. Let w be the total number of arrivals during W. Let $r_k, T_k, k = 1, ...w$ denote the bandwidth requirement and the service time, respectively, of the k-th arrival. (This

[1]Note that an efficient algorithm can solve each subproblem B_i (4.8) in at most $O[\theta(i) \log \theta(i)]$ steps (see Appendix B.2).

[2]This is precisely the procedure we proposed in Section 3.3.2.

information can be carried along with the connection setup message when the user arrives.) Let

$$\theta = \frac{\sum_{k=1}^{w} r_k T_k}{W}.$$

Then, it is easy to check that

$$\mathbf{E}[\theta] = \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} r_i p_{ij} H_{ij}^l,$$

i.e., θ is an unbiased estimate of the aggregate load. Note that no prior knowledge on the demand parameters λ_i and μ_i is needed in the estimator. Hence, the algorithm can automatically track the changes in the arrival rates and service times of the users [21].

4.2.2 The Algorithm A

In the rest of the chapter, we will study the following algorithm that generalizes algorithm $\mathcal{P}$. (This is precisely the Algorithm $\mathcal{A}$ that we used in Chapter 3.)

Algorithm $\mathcal{A}$:

Fix $K \geq 1$. At the t-th iteration:

A1) Fix $\vec{y} = \vec{y}(t)$ and use gradient descent iteration (4.9) on the dual variable $\vec{q}$ for K times. To be precise, let $\vec{q}(t,0) = \vec{q}(t)$. Repeat the following procedure for each $k = 0, 1, ...K-1$:

Let $\vec{x}(t,k)$ be the primal variable that solves (4.8) given the dual variable $\vec{q}(t,k)$, i.e., $\vec{x}(t,k) = \underset{\vec{x} \in C}{\operatorname{argmax}} L(\vec{x}, \vec{q}(t,k), \vec{y}(t))$. Update the dual variables by

$$q^l(t,k+1) = \left[q^l(t,k) + \alpha^l (\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t,k) - R^l) \right]^+, \text{ for all } l. \quad (4.10)$$

A2) Let $\vec{q}(t+1) = \vec{q}(t,K)$. Let $\vec{z}(t)$ be the primal variable that solves (4.8) given the new dual variable $\vec{q}(t+1)$, i.e., $\vec{z}(t) = \underset{\vec{x} \in C}{\operatorname{argmax}} L(\vec{x}, \vec{q}(t+1), \vec{y}(t))$. Set

$$y_{ij}(t+1) = y_{ij}(t) + \beta_i (z_{ij}(t) - y_{ij}(t)), \text{ for all } i, j, \quad (4.11)$$

where $0 < \beta_i \leq 1$ for each i.

As discussed in Section 4.2.1, in certain applications the aggregate load $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t,k)$ is estimated through online measurement with non-negligible noises. The update (4.10) should then be replaced by:

$$q^l(t,k+1) = \left[q^l(t,k) + \alpha^l \left(\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t,k) - R^l + n^l(t,k) \right) \right]^+, \tag{4.12}$$

where $n^l(t,k)$ represents the measurement noise at link l.

From now on, we will refer to (4.10) or (4.12) as the *dual update*, and (4.11) as the *primal update*. A *stationary point of the algorithm* $\mathcal{A}$ is defined to be a primal-dual pair $(\vec{y^*}, \vec{q^*})$ such that

$$\vec{y^*} = \underset{\vec{x} \in C}{\operatorname{argmax}} L(\vec{x}, \vec{q^*}, \vec{y^*}), \quad \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l y_{ij}^* \leq R^l \text{ for all } l, \tag{4.13}$$

$$q^{l,*} \geq 0, \text{ and } q^{l,*} \left(\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l y_{ij}^* - R^l \right) = 0 \text{ for all } l.$$

These are precisely the *complementary slackness* conditions for the problem (4.1). By standard duality theory, for any stationary point $(\vec{y^*}, \vec{q^*})$ of the algorithm $\mathcal{A}$, $\vec{x} = \vec{y^*}$ solves the problem (4.1).

The main components of algorithm $\mathcal{A}$ (i.e., the primal and dual updates) are essentially the same as that of the standard proximal optimization algorithm $\mathcal{P}$. Therefore, our new algorithm $\mathcal{A}$ inherits from algorithm $\mathcal{P}$ those attributes desirable for online implementation. However, the main difference is that, in algorithm $\mathcal{A}$, only K number of dual updates are executed at each iteration of step A1. If $K = \infty$, then algorithm $\mathcal{A}$ and algorithm $\mathcal{P}$ will be equivalent, i.e., at each iteration of step A1 the optimization (4.5) is solved exactly. As we discussed earlier, such a two-level convergence structure is inappropriate for online implementation because it would be impractical to carry out an algorithm in phases where each phase consists of an *infinite* number of dual updates. Further, because each phase only serves to solve the augmented problem (4.5), such a two-level convergence structure is also likely to

slow the convergence of the entire algorithm as too many dual updates are wasted at each phase.

On the other hand, when $K < \infty$, at best an approximate solution to (4.5) is obtained at each iteration of step A1. If the accuracy of the approximate solution can be controlled appropriately (see [65]), one can still show convergence of algorithm $\mathcal{A}$. However, in this case the number of dual updates K in step A1 has to depend on the required accuracy and usually needs to be large. Further, for online implementation, it is also difficult to control the accuracy of the approximate solution to (4.5) in a distributed fashion.

In this work, we take an *entirely different* approach. We do not require a two-level convergence structure and we allow an arbitrary choice of $K \geq 1$. Hence, our approach does not impose any requirement on the accuracy of the approximate solution to (4.5). As we just discussed, relaxing the algorithm in such a way is a crucial step in making the algorithm amenable to online distributed implementation. Somewhat surprisingly, we will show in the next section that algorithm $\mathcal{A}$ will converge for any $K \geq 1$.

4.3 Convergence without Measurement Noise

In this section, we study the convergence of algorithm $\mathcal{A}$ when there is no measurement noise, i.e., when the dynamics of the system are described by (4.10) and (4.11). The convergence of algorithm $\mathcal{A}$ can be most easily understood by looking at its continuous-time approximation as follows:

Algorithm $\mathcal{AC}$:

A1-C) dual update:

$$\frac{d}{dt}q^l(t) = \begin{cases} \hat{\alpha}^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) - R^l) & \text{if } q^l(t) > 0 \text{ or } \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) \geq R^l \\ 0 & \text{otherwise} \end{cases}, \tag{4.14}$$

where $\vec{x}(t) = \underset{\vec{x}\in C}{\operatorname{argmax}}\, L(\vec{x}, \vec{q}(t), \vec{y}(t))$.

A2-C) primal update:

$$\frac{d}{dt}y_{ij}(t) = \hat{\beta}_i(x_{ij}(t) - y_{ij}(t)). \tag{4.15}$$

Note that $\hat{\alpha}^l\,dt$ and $\hat{\beta}_i\,dt$ would correspond to the step-sizes α^l and β_i in the discrete-time algorithm $\mathcal{A}$. The continuous-time algorithm $\mathcal{AC}$ can be view as the functional limit of the discrete-time algorithm by *driving the step-sizes α^l and β_i to zero* and by appropriately rescaling time (see Section 4.4 and Appendix C.3).

For the sake of brevity, we will use the following vector notation for the rest of the chapter. Let E denote the matrix with L rows and $\sum_{i=1}^{I}\theta(i)$ columns such that the $(l, \sum_{k=1}^{i-1}\theta(k)+j)$ element is E^l_{ij}. Let $R = [R^1, R^2, ...R^l]^T$. Then the constraint of problem (4.1) can be written as $E\vec{x} \leq R$. Let V and $\hat{B}$ be $\sum_{i=1}^{I}\theta(i) \times \sum_{i=1}^{I}\theta(i)$ diagonal matrices, where the $(\sum_{k=1}^{i-1}\theta(k)+1)$-th to $(\sum_{k=1}^{i}\theta(k))$-th diagonal elements are c_i and $\hat{\beta}_i$, respectively (i.e., each c_i or $\hat{\beta}_i$ is repeated $\theta(i)$ times). Let $\hat{A}$ be the $L \times L$ diagonal matrix whose l-th diagonal element is $\hat{\alpha}^l$. It will also be convenient to view the objective function in (4.1) as a concave function of $\vec{x}$, i.e, $\mathbf{f}(\vec{x}) = \sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij})$. Further, we can incorporate the constraint $\vec{x} \in C$ into the definition of the function $\mathbf{f}$ by setting $\mathbf{f}(\vec{x}) = -\infty$ if $\vec{x} \notin C$. Then the function $\mathbf{f}$ is still concave, and the problem (4.1) can be simply rephrased as maximizing $\mathbf{f}(\vec{x})$ subject to $E\vec{x} \leq R$. The Lagrangian (4.6) also becomes:

$$L(\vec{x}, \vec{q}, \vec{y}) \;\; = \;\; \mathbf{f}(\vec{x}) - \vec{x}^T E^T \vec{q} - \frac{1}{2}(\vec{x} - \vec{y})^T V(\vec{x} - \vec{y}) + \vec{q}^T R. \tag{4.16}$$

The continuous time algorithm $\mathcal{AC}$ can then be viewed as the *projected forward iteration* for solving the zeros of the following *monotone mapping* [69]:

$$\mathcal{T}: \quad [\vec{y}, \vec{q}] \rightarrow [\vec{u}, \vec{v}], \tag{4.17}$$

with

$$\vec{u}(\vec{y}, \vec{q}) = -V(\vec{x^0}(\vec{y}, \vec{q}) - \vec{y}), \qquad \vec{v}(\vec{y}, \vec{q}) = -(E\vec{x^0}(\vec{y}, \vec{q}) - R),$$

where $\vec{x^0}(\vec{y}, \vec{q}) = \text{argmax}_{\vec{x}}\, L(\vec{x}, \vec{q}, \vec{y})$. Define the inner product

$$\langle\, [\vec{y}, \vec{q}], [\vec{u}, \vec{v}]\, \rangle = \vec{y}^T\vec{u} + \vec{q}^T\vec{v}, \tag{4.18}$$

and the following norms:

$$||\vec{q}||_{\hat{A}} = \vec{q}^T \hat{A}^{-1} \vec{q}, \quad ||\vec{y}||_V = \vec{y}^T V \vec{y}, \quad ||\vec{y}||_{\hat{B}V} = \vec{y}^T \hat{B}^{-1} V \vec{y}. \tag{4.19}$$

Part 3 of the following Lemma shows that the mapping $\mathcal{T}$ is *monotone* [69]. Note that a mapping $\mathcal{T}$ is *monotone* if

$$\langle\, X_1 - X_2, \mathcal{T}X_1 - \mathcal{T}X_2 \,\rangle \geq 0 \text{ for any } X_1 \text{ and } X_2. \tag{4.20}$$

Lemma 4.3.1 *Fix $\vec{y} = \vec{y}(t)$. Let $\vec{q}_1, \vec{q}_2$ be two implicit cost vectors, and let $\vec{x}_1$, $\vec{x}_2$ be the corresponding maximizers of the Lagrangian (4.16), i.e., $\vec{x}_1 = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}_1, \vec{y}(t))$ and $\vec{x}_2 = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}_2, \vec{y}(t))$. Then,*

1. $(\vec{q}_2 - \vec{q}_1)^T E(\vec{x}_2 - \vec{x}_1) \leq -(\vec{x}_2 - \vec{x}_1)^T V(\vec{x}_2 - \vec{x}_1)$.
2. $(\vec{x}_2 - \vec{x}_1)^T V(\vec{x}_2 - \vec{x}_1) \leq (\vec{q}_2 - \vec{q}_1)^T E V^{-1} E^T (\vec{q}_2 - \vec{q}_1)$, *and*
3. $\langle\, [\vec{y}_1 - \vec{y}_2, \vec{q}_1 - \vec{q}_2], \mathcal{T}[\vec{y}_1, \vec{q}_1] - \mathcal{T}[\vec{y}_2, \vec{q}_2] \,\rangle \geq 0$ *for any $(\vec{y}_1, \vec{q}_1)$ and $(\vec{y}_2, \vec{q}_2)$.*

Remark: Part 2 of Lemma 4.3.1 also shows that, given $\vec{y}$, the mapping from $\vec{q}$ to $\vec{x}$ is continuous.

Proof We start with some additional notation. For $\vec{x^0} = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}, \vec{y})$, by taking subgradients (see [65]) of the Lagrangian (4.16) with respect to $\vec{x}$, we can conclude that there must exist a subgradient $\nabla \mathbf{f}(\vec{x^0})$ of $\mathbf{f}$ at $\vec{x^0}$ such that

$$\nabla \mathbf{f}(\vec{x^0})|_{\vec{y},\vec{q}} - E^T \vec{q} - V(\vec{x^0} - \vec{y}) = 0. \tag{4.21}$$

Note that $\nabla \mathbf{f}(\vec{x^0})|_{\vec{y},\vec{q}}$ defined above depends not only on the function $\mathbf{f}$ and the vector $\vec{x^0}$, but also on $\vec{y}$ and $\vec{q}$. However, in the derivation that follows, the dependence on $\vec{y}$ and $\vec{q}$ is easy to identify. Hence, for the sake of brevity, we will drop the subscripts and write $\nabla \mathbf{f}(\vec{x^0})$ when there is no ambiguity. Similarly, let $(\vec{y^*}, \vec{q^*})$ denote a stationary point of algorithm $\mathcal{A}$. Then $\vec{y^*} = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q^*}, \vec{y^*})$, and we can define $\nabla \mathbf{f}(\vec{y^*})$ as the subgradient of $\mathbf{f}$ at $\vec{y^*}$ such that

$$\nabla \mathbf{f}(\vec{y^*}) - E^T \vec{q^*} = 0. \tag{4.22}$$

Applying (4.21) for $\vec{q}_1$ and $\vec{q}_2$, and taking difference, we have,

$$E^T(\vec{q}_2 - \vec{q}_1) \;=\; [\nabla \mathbf{f}(\vec{x}_2) - \nabla \mathbf{f}(\vec{x}_1)] - V(\vec{x}_2 - \vec{x}_1).$$

The concavity of $\mathbf{f}$ dictates that, for any $\vec{x}_1$, $\vec{x}_2$ and $\nabla \mathbf{f}(\vec{x}_1)$, $\nabla \mathbf{f}(\vec{x}_2)$,

$$[\nabla \mathbf{f}(\vec{x}_2) - \nabla \mathbf{f}(\vec{x}_1)]^T (\vec{x}_2 - \vec{x}_1) \leq 0. \tag{4.23}$$

Hence,

$$\begin{aligned}(\vec{q}_2 - \vec{q}_1)^T E(\vec{x}_2 - \vec{x}_1) &= [\nabla \mathbf{f}(\vec{x}_2) - \nabla \mathbf{f}(\vec{x}_1)]^T (\vec{x}_2 - \vec{x}_1) - (\vec{x}_2 - \vec{x}_1)^T V(\vec{x}_2 - \vec{x}_1) \\ &\leq -(\vec{x}_2 - \vec{x}_1)^T V(\vec{x}_2 - \vec{x}_1).\end{aligned}$$

Part 2 of the Lemma can be shown analogously. To show Part 3, let $\vec{x}_2^{\,'} = \text{argmax}_{\vec{x} \geq 0} L(\vec{x}, \vec{q}_2, \vec{y}_2)$. Applying (4.21) for $\vec{q}_1, \vec{y}_1$ and $\vec{q}_2, \vec{y}_2$, and taking difference, we have

$$E^T(\vec{q}_2 - \vec{q}_1) \;=\; \left[\nabla \mathbf{f}(\vec{x}_2^{\,'}) - \nabla \mathbf{f}(\vec{x}_1)\right] - V(\vec{x}_2^{\,'} - \vec{x}_1) + V(\vec{y}_2 - \vec{y}_1).$$

Hence, using (4.23) again, we have,

$$\begin{aligned}&\langle\, [\vec{y}_1 - \vec{y}_2, \vec{q}_1 - \vec{q}_2], \mathcal{T}[\vec{y}_1, \vec{q}_1] - \mathcal{T}[\vec{y}_2, \vec{q}_2] \,\rangle \\ =\;& -(\vec{y}_1 - \vec{y}_2)^T V(\vec{x}_1 - \vec{x}_2^{\,'} - (\vec{y}_1 - \vec{y}_2)) - (\vec{q}_1 - \vec{q}_2)^T E(\vec{x}_2^{\,'} - \vec{x}_1) \\ \geq\;& ||\vec{y}_1 - \vec{y}_2||_V^2 - 2(\vec{x}_1 - \vec{x}_2^{\,'})^T V(\vec{y}_1 - \vec{y}_2) + ||\vec{x}_1 - \vec{x}_2^{\,'}||_V^2 \geq 0. \end{aligned} \tag{4.24}$$

■

We can now prove the following result.

Proposition 4.3.1 *The continuous-time algorithm $\mathcal{AC}$ will converge to a stationary point $(\vec{y^*}, \vec{q^*})$ of the algorithm $\mathcal{A}$ for any choice of $\hat{\alpha}^l > 0$ and $\hat{\beta}_i > 0$.*

Proof We can prove Proposition 4.3.1 using the following Lyapunov function. Let

$$\mathcal{V}(\vec{y}, \vec{q}) = ||\vec{q} - \vec{q^*}||_{\hat{A}} + ||\vec{y} - \vec{y^*}||_{\hat{B}V}, \tag{4.25}$$

where the norms are defined in (4.19). We will first show that, if $(\vec{y}(t), \vec{q}(t))$ is governed by the algorithm $\mathcal{AC}$, $\mathcal{V}(\vec{y}(t), \vec{q}(t))$ is non-increasing in t. To see this, note that,

$$\frac{d}{dt}\mathcal{V}(\vec{y}(t), \vec{q}(t)) = 2(\vec{q}(t) - \vec{q^*})^T \hat{A}^{-1} \frac{d}{dt}\vec{q}(t) + 2(\vec{y}(t) - \vec{y^*})^T \hat{B}^{-1} V \frac{d}{dt}\vec{y}(t). \quad (4.26)$$

By (4.14), we have for all l,

$$\begin{aligned} 2(q^l(t) - q^{l,*})\frac{d}{dt}q^l(t) &\leq 2\alpha^l(q^l(t) - q^{l,*})(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) - R^l) \\ &= 2\alpha^l(q^l(t) - q^{l,*})\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l (x_{ij}(t) - y_{ij}^*) \\ &\quad + 2\alpha^l(q^l(t) - q^{l,*})(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l y_{ij}^* - R^l). \end{aligned}$$

Since $(\vec{y^*}, \vec{q^*})$ is a stationary point, $\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l y_{ij}^* - R^l \leq 0$ and $q^{l,*}(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l y_{ij}^* - R^l) = 0$. Further, since $q^l(t) \geq 0$, we obtain,

$$2(q^l(t) - q^{l,*})\frac{d}{dt}q^l(t) \leq 2\alpha^l(q^l(t) - q^{l,*})\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l (x_{ij}(t) - y_{ij}^*). \quad (4.27)$$

Substituting (4.15) and (4.27) into (4.26), we have,

$$\begin{aligned} \frac{d}{dt}\mathcal{V}(\vec{y}(t), \vec{q}(t)) &= 2(\vec{q}(t) - \vec{q^*})^T \hat{A}^{-1} \frac{d}{dt}\vec{q}(t) + 2(\vec{y}(t) - \vec{y^*})^T \hat{B}^{-1} V \frac{d}{dt}\vec{y}(t) \\ &\leq 2(\vec{q}(t) - \vec{q^*})^T E(\vec{x}(t) - \vec{y^*}) + 2(\vec{y}(t) - \vec{y^*})^T V(\vec{x}(t) - \vec{y}(t)) \\ &= -2\langle\, [\vec{y}(t) - \vec{y^*}, \vec{q}(t) - \vec{q^*}], \mathcal{T}[\vec{y}(t), \vec{q}(t)] - \mathcal{T}[\vec{y^*}, \vec{q^*}]\,\rangle, \end{aligned}$$

where the inner product is defined in (4.18). Hence, by Lemma 4.3.1,

$$\frac{d}{dt}\mathcal{V}(\vec{y}(t), \vec{q}(t)) \leq 0.$$

In fact, using (4.24), we can show that

$$\frac{d}{dt}\mathcal{V}(\vec{y}(t), \vec{q}(t)) \leq -2(\vec{x}(t) - \vec{y}(t))^T V(\vec{x}(t) - \vec{y}(t)) \leq 0. \quad (4.28)$$

Therefore, $\mathcal{V}(\vec{y}(t), \vec{q}(t))$ must converge to a limit $\mathcal{V}_0$ as $t \to \infty$, i.e.,

$$\lim_{t\to\infty} ||\vec{q}(t) - \vec{q^*}||_{\hat{A}} + ||\vec{y}(t) - \vec{y^*}||_{\hat{B}V} = \mathcal{V}_0 \geq 0. \tag{4.29}$$

It remains to show that $\mathcal{V}_0 = 0$ for some choice of the stationary point $(\vec{y^*}, \vec{q^*})$. Towards this end, define a set $\mathcal{M}$ to be an *invariant set* with respect to algorithm $\mathcal{AC}$ if, starting from any point in $\mathcal{M}$, the trajectory $(\vec{y}, \vec{q})$ governed by the algorithm $\mathcal{AC}$ will lie entirely in $\mathcal{M}$. Let $\mathcal{M}_0$ denote the set of limit points of $(\vec{y}(t), \vec{q}(t))$. Since $\mathcal{V}(\vec{y}(t), \vec{q}(t))$ is non-negative, by LaSalle's Theorem [70, p115], it follows that $\mathcal{M}_0$ is an invariant set, and $\frac{d}{dt}\mathcal{V}(\vec{y}, \vec{q}) = 0$ for any points in $\mathcal{M}_0$. We will now show that $\mathcal{M}_0$ contains a stationary point of algorithm $\mathcal{A}$. To see this, pick a point $(\vec{y}_0, \vec{q}_0) \in \mathcal{M}_0$. Let $\vec{\tilde{y}}(0) = \vec{y}_0$ and $\vec{\tilde{q}}(0) = \vec{q}_0$. Let $(\vec{\tilde{y}}(t), \vec{\tilde{q}}(t))$ denote the trajectory of algorithm $\mathcal{AC}$ starting from $(\vec{y}_0, \vec{q}_0)$. We first study $\vec{\tilde{y}}(t)$. Using the fact that

$$\frac{d}{dt}\mathcal{V}(\vec{\tilde{y}}(t), \vec{\tilde{q}}(t)) = 0 \text{ for all } t \geq 0,$$

we have, from (4.28),

$$\vec{\tilde{x}}(t) = \vec{\tilde{y}}(t) \text{ for all } t \geq 0,$$

where $\vec{\tilde{x}}(t) = \text{argmax}_{\vec{x}}\, L(\vec{x}, \vec{\tilde{q}}(t), \vec{\tilde{y}}(t))$. Hence,

$$\frac{d}{dt}\vec{\tilde{y}}(t) = \hat{B}(\vec{\tilde{x}}(t) - \vec{\tilde{y}}(t)) = 0.$$

Therefore, $\vec{\tilde{x}}(t) = \vec{\tilde{y}}(t) = \vec{\tilde{y}}(0)$ for all $t \geq 0$. We next study $\vec{\tilde{q}}(t)$. For any link l, there are two possibilities. The first possibility is

$$\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{x}_{ij}(t) = \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{y}_{ij}(0) > R^l. \tag{4.30}$$

If this was true, since

$$\frac{d}{dt}\tilde{q}^l(t) = \alpha^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{x}_{ij}(t) - R^l) = \alpha^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{y}_{ij}(0) - R^l) > 0,$$

we have $\tilde{q}^l(t) \to \infty$, as $t \to \infty$. It implies that $\tilde{x}_{ij}(t) \to 0$, as $t \to \infty$, for all i, j such that $E_{ij}^l > 0$. We then conclude that

$$\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{x}_{ij}(t) \to 0 \text{ as } t \to \infty.$$

This contradicts with the assumption (4.30). Hence, only the other possibility can be true, i.e.,

$$\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l}\tilde{x}_{ij}(t) = \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l}\tilde{y}_{ij}(0) \leq R^{l}.$$

Note that if the above inequality is strict, then $\tilde{q}^{l}(t)$ will decrease to zero after a finite time t. On the other hand, $\tilde{q}^{l}(t)$ will be unchanged if $\tilde{q}^{l}(t) = 0$ or $\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l}\tilde{y}_{ij}(0) = R^{l}$.

Define $(\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$ as

$$\begin{aligned}
\vec{\tilde{y}}_0 &= \vec{y}_0 = \vec{\tilde{y}}(0) \\
\tilde{q}_0^{l} &= \begin{cases} q_0^{l} & \text{if } \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l}\tilde{y}_{ij}(0) = R^{l} \\ 0 & \text{if } \sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^{l}\tilde{y}_{ij}(0) < R^{l} \end{cases}.
\end{aligned}$$

We can now conclude that, starting from $(\vec{y}_0, \vec{q}_0)$, eventually $(\vec{\tilde{y}}(t), \vec{\tilde{q}}(t)) = (\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$. By the definition of $\mathcal{M}_0$, $(\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$ is also a limit point of $(\vec{y}(t), \vec{q}(t))$. Further, $(\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$ satisfies the definition of the stationary point of algorithm $\mathcal{A}$. Pick a subsequence $(\vec{y}(t_h), \vec{q}(t_h)), h = 1, 2, ...$ such that $t_h \to \infty$ and $(\vec{y}(t_h), \vec{q}(t_h)) \to (\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$, as $h \to \infty$. We can now replace $(\vec{y^*}, \vec{q^*})$ by $(\vec{\tilde{y}}_0, \vec{\tilde{q}}_0)$ in (4.29), and conclude that

$$\begin{aligned}
& \lim_{t\to\infty} ||\vec{q}(t) - \vec{\tilde{q}}_0||_{\hat{A}} + ||\vec{y}(t) - \vec{\tilde{y}}_0||_{\hat{B}V} \\
= & \lim_{h\to\infty} ||\vec{q}(t_h) - \vec{\tilde{q}}_0||_{\hat{A}} + ||\vec{y}(t_h) - \vec{\tilde{y}}_0||_{\hat{B}V} \\
= & \ 0.
\end{aligned}$$

The result then follows. ■

We next study the convergence of the discrete-time algorithm $\mathcal{A}$. Since the continuous-time algorithm $\mathcal{AC}$ can be viewed as an approximation of the discrete-time algorithm $\mathcal{A}$ when the step-sizes are close to zero, we can then expect from Proposition 4.3.1 that algorithm $\mathcal{A}$ will converge when the step-sizes α^{l} and β_i are small. However, when these step-sizes are too small, convergence is unnecessarily

slow. Hence, in practice, we would like to choose larger step-sizes, while still preserving the convergence of the algorithm. Such knowledge on the step-size rule can only be obtained by studying the convergence of the discrete-time algorithm directly.

Typically, convergence of the discrete-time algorithms requires stronger conditions on the associated mapping $\mathcal{T}$ defined in (4.17), i.e., the mapping $\mathcal{T}$ needs to be *strictly monotone* [69]. A mapping $\mathcal{T}$ is *strictly monotone* if and only if

$$\langle\, X_1 - X_2, \mathcal{T}X_1 - \mathcal{T}X_2 \,\rangle \geq d||\mathcal{T}X_1 - \mathcal{T}X_2|| \text{ for any vectors } X_1 \text{ and } X_2, \quad (4.31)$$

where d is a positive constant and $||\cdot||$ is an appropriately chosen norm. Note that strict monotonicity in (4.31) is stronger than *monotonicity* in (4.20). Such type of strict monotonicity indeed holds for the case when $K = \infty$, which is why the convergence is much easier to establish under the two-level convergence structure. However, when $K < \infty$, strict monotonicity will not hold for the mapping $\mathcal{T}$ defined in (4.17) *whenever some users in the network have multiple paths.* To see this, choose $X_2 = [\vec{y^*}, \vec{q^*}]$ to be a stationary point of algorithm $\mathcal{A}$ and assume that $E\vec{y^*} = R$. Let $X_1 = [\vec{y}, \vec{q^*}]$ such that $\sum_{j=1}^{\theta(i)} y_{ij} = \sum_{j=1}^{\theta(i)} y^*_{ij}$ for all i. Note that for a user i that has multiple paths, we can still choose $y_{ij} \neq y^*_{ij}$ for some j such that $E\vec{y} \neq R$. By comparing with the complementary slackness conditions (4.13), we have $\vec{x^0}(\vec{y}, \vec{q^*}) \triangleq \text{argmax}_{\vec{x}}\, L(\vec{x}, \vec{q^*}, \vec{y}) = \vec{y}$. Hence,

$$\vec{u}(\vec{y}, \vec{q^*}) = 0, \text{ and } \vec{v}(\vec{y}, \vec{q^*}) = -(E\vec{y} - R).$$

Further, since $\vec{u}(\vec{y^*}, \vec{q^*}) = 0$ and $\vec{v}(\vec{y^*}, \vec{q^*}) = 0$ by the complementary slackness conditions (4.13), we have

$$\begin{aligned}\langle\, X_1 - X_2, \mathcal{T}X_1 - \mathcal{T}X_2 \,\rangle &= \langle\, [\vec{y} - \vec{y^*}, \vec{q^*} - \vec{q^*}], \mathcal{T}[\vec{y}, \vec{q^*}] - \mathcal{T}[\vec{y^*}, \vec{q^*}] \,\rangle \\ &= [\vec{y} - \vec{y^*}]^T(\vec{u}(\vec{y}, \vec{q^*}) - \vec{u}(\vec{y^*}, \vec{q^*})) + \vec{0}^{\,T}(\vec{v}(\vec{y}, \vec{q^*}) - \vec{v}(\vec{y^*}, \vec{q^*})) = 0.\end{aligned}$$

However, $\mathcal{T}X_1 - \mathcal{T}X_2 = [0, -(E\vec{y} - R)] \neq 0$. Hence, the inequality (4.31) will never hold! As we have just seen, it is precisely the multi-path nature of the problem that leads to this lack of strict monotonicity. (One can indeed show that (4.31) would

have held if all users had one single path and the utility functions $f_i(\cdot)$ were strictly concave.)

This lack of strict monotonicity when $K < \infty$ forces us to carry out a more refined convergence analysis than that in the standard convex programming literature. We will need the following key supporting result. Let $(\vec{y^*}, \vec{q^*})$ denote a stationary point of algorithm $\mathcal{A}$. Using (4.23), we have

$$\left[\nabla \mathbf{f}(\vec{x}_1) - \nabla \mathbf{f}(\vec{y^*})\right]^T (\vec{x}_1 - \vec{y^*}) \leq 0. \tag{4.32}$$

The following Lemma can be viewed as an extension of the above inequality. The proof is very technical and is given in Appendix C.1.

Lemma 4.3.2 *Fix $\vec{y} = \vec{y}(t)$. Let $\vec{q}_1, \vec{q}_2$ be two implicit cost vectors, and let $\vec{x}_1$, $\vec{x}_2$ be the corresponding maximizers of the Lagrangian (4.16). Then,*

$$\left[\nabla \mathbf{f}(\vec{x}_1) - \nabla \mathbf{f}(\vec{y^*})\right]^T (\vec{x}_2 - \vec{y^*}) \;\; \leq \;\; \frac{1}{2}(\vec{q}_2 - \vec{q}_1)^T E V^{-1} E^T (\vec{q}_2 - \vec{q}_1),$$

where $\nabla \mathbf{f}(\vec{x}_1)$ and $\nabla \mathbf{f}(\vec{y^*})$ are defined in (4.21) and (4.22), respectively.

Remark: If $\vec{q}_2 = \vec{q}_1$, then $\vec{x}_2 = \vec{x}_1$ and we get back to (4.32). Lemma 4.3.2 tells us that as long as $\vec{q}_1$ is not very different from $\vec{q}_2$, the cross-product on the left hand side will not be far above zero either.

We can then prove the following main result, which establishes the sufficient condition on the step-sizes for the convergence of the discrete-time algorithm $\mathcal{A}$.

Proposition 4.3.2 *Fix $1 \leq K \leq \infty$. As long as the step-size α^l is small enough, algorithm $\mathcal{A}$ will converge to a stationary point $(\vec{y^*}, \vec{q^*})$ of the algorithm, and $\vec{x^*} = \vec{y^*}$ will solve the original problem (4.1). The sufficient condition for convergence is:*

$$\max_l \alpha^l < \begin{cases} \frac{2}{\mathcal{SL}} \min_i c_i & \text{if } K = \infty \\ \frac{1}{2\mathcal{SL}} \min_i c_i & \text{if } K = 1 \\ \frac{4}{5K(K+1)\mathcal{SL}} \min_i c_i & \text{if } K > 1 \end{cases},$$

where $\mathcal{L} = \max\{\sum_{l=1}^{L} E_{ij}^l, i = 1, ..., I, j = 1, ...\theta(i)\}$, and $\mathcal{S} = \max\{\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l, l = 1, ..., L\}$.

Proposition 4.3.2 establishes the convergence of algorithm $\mathcal{A}$ for any value of K (even $K = 1$ is good enough). Hence, the typical two-level convergence structure is no longer required. Further, we observe that the sufficient condition for convergence when $K = 1$ differs from that of $K = \infty$ by only a factor of 4. Note that for $K = \infty$, the sufficient condition in fact ensures the convergence of the dual updates to the solution of the augmented problem (4.5) during *one iteration* of step A1. On the other hand, the sufficient condition for $K = 1$ ensures the convergence of the entire algorithm $\mathcal{A}$. By showing that the sufficient conditions for the two cases differ by only a factor of 4, we can infer that the convergence of the entire algorithm when $K = 1$ is not necessarily much slower than the convergence of one iteration of step A1 when $K = \infty$. Hence, the algorithm $\mathcal{A}$ with $K = 1$ in fact converges much faster. For $K > 1$, our result requires that the step-size be inversely proportional to K^2. This is probably not as tight a result as one could get: we conjecture that the same condition for $K = 1$ would work for any K. However, we leave this for future work. We also note that c_i appears on the right hand side of the sufficient conditions. Hence, by making the objective function more concave, we also relax the requirement on the step-sizes α^l. Finally, Proposition 4.3.2 indicates that convergence will hold for any β_i (the step-size in the primal update) that is in $(0, 1]$. In summary, the discrete-time analysis allows much wider choices of the step-sizes than those predicted by the continuous-time analysis.

Proof **[of Proposition 4.3.2]** In order to highlight the main ideas of the proof, we will focus here on the case when $K = 1$. The other cases can be shown analogously (see Appendix C.2 for details). Define matrices A and B analogously to matrices $\hat{A}$ and $\hat{B}$, respectively, except that their diagonal elements are now filled with the step-sizes α^l and β_i of the discrete-time algorithm $\mathcal{A}$. Define the norms analogously to (4.19). When $K = 1$,

$$\vec{q}(t+1) = [\vec{q}(t) + A(E\vec{x}(t) - R)]^+. \tag{4.33}$$

Let $(\vec{y^*}, \vec{q^*})$ be any stationary point of algorithm $\mathcal{A}$. We will show that the Lyapunov function

$$\mathcal{V}(\vec{y}(t), \vec{q}(t)) = ||\vec{q}(t) - \vec{q^*}||_A + ||\vec{y}(t) - \vec{y^*}||_{BV}$$

is non-increasing in t. Using the property of the projection mapping [64, Proposition 3.2(b), p211], we have

$$(\vec{q}(t+1) - \vec{q^*})^T A^{-1}(\vec{q}(t+1) - [\vec{q}(t) + A(E\vec{x}(t) - R)]) \leq 0. \tag{4.34}$$

Hence,

$$\begin{aligned}
& ||\vec{q}(t+1) - \vec{q^*}||_A \\
= \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2(\vec{q}(t+1) - \vec{q^*})^T A^{-1}(\vec{q}(t+1) - \vec{q}(t)) \\
\leq \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2(\vec{q}(t+1) - \vec{q^*})^T (E\vec{x}(t) - R) \qquad (4.35) \\
\leq \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2(\vec{q}(t+1) - \vec{q^*})^T E(\vec{x}(t) - \vec{y^*}), \qquad (4.36)
\end{aligned}$$

where in the last step we have used the fact that $E\vec{y^*} - R \leq 0$ and $\vec{q^*}^T(E\vec{y^*} - R) = 0$. On the other hand, since $y_{ij}(t+1) = (1-\beta_i)y_{ij}(t) + \beta_i z_{ij}(t)$, we have

$$\begin{aligned}
(y_{ij}(t+1) - y^*_{ij})^2 &\leq (1-\beta_i)(y_{ij}(t) - y^*_{ij})^2 + \beta_i(z_{ij}(t) - y^*_{ij})^2 \\
||\vec{y}(t+1) - \vec{y^*}||_{BV} - ||\vec{y}(t) - \vec{y^*}||_{BV} &\leq ||\vec{z}(t) - \vec{y^*}||_V - ||\vec{y}(t) - \vec{y^*}||_V. \qquad (4.37)
\end{aligned}$$

Hence, combining (4.36) and (4.37), we have,

$$\begin{aligned}
& ||\vec{q}(t+1) - \vec{q^*}||_A + ||\vec{y}(t+1) - \vec{y^*}||_{BV} - (||\vec{q}(t) - \vec{q^*}||_A + ||\vec{y}(t) - \vec{y^*}||_{BV}) \\
\leq \; & -||\vec{q}(t+1) - \vec{q}(t)||_A + 2(\vec{q}(t+1) - \vec{q^*})^T E(\vec{x}(t) - \vec{y^*}) \\
& \quad + ||\vec{z}(t) - \vec{y^*})||_V - ||\vec{y}(t) - \vec{y^*}||_V \\
\leq \; & -||\vec{q}(t+1) - \vec{q}(t)||_A \\
& \quad + \left\{||\vec{z}(t) - \vec{y^*})||_V - ||\vec{y}(t) - \vec{y^*}||_V - 2(\vec{z}(t) - \vec{y}(t))^T V(\vec{x}(t) - \vec{y^*})\right\} \qquad (4.38) \\
& \quad + 2\left[\nabla \mathbf{f}(\vec{z}(t)) - \nabla \mathbf{f}(\vec{y^*})\right]^T (\vec{x}(t) - \vec{y^*}), \qquad (4.39)
\end{aligned}$$

where in the last step we have used (4.21) and (4.22), and consequently

$$E^T(\vec{q}(t+1) - \vec{q^*}) = \nabla \mathbf{f}(\vec{z}(t)) - \nabla \mathbf{f}(\vec{y^*}) - V(\vec{z}(t) - \vec{y}(t)). \tag{4.40}$$

By simple algebraic manipulation, we can show that the second term (4.38) is equal to

$$\begin{aligned} & ||\vec{z}(t) - \vec{y^*}||_V - ||\vec{y}(t) - \vec{y^*}||_V - 2(\vec{z}(t) - \vec{y}(t))^T V(\vec{x}(t) - \vec{y^*}) \\ = \; & ||(\vec{z}(t) - \vec{x}(t)||_V - ||\vec{y}(t) - \vec{x}(t)||_V. \end{aligned} \tag{4.41}$$

Invoking Lemma 4.3.1, part 2,

$$||\vec{z}(t) - \vec{x}(t)||_V \leq (\vec{q}(t+1) - \vec{q}(t))^T E V^{-1} E^T (\vec{q}(t+1) - \vec{q}(t)). \tag{4.42}$$

For the third term (4.39), we can invoke Lemma 4.3.2,

$$2\left[\nabla \mathbf{f}(\vec{z}(t)) - \nabla \mathbf{f}(\vec{y^*})\right]^T (\vec{x}(t) - \vec{y^*}) \leq (\vec{q}(t+1) - \vec{q}(t))^T E V^{-1} E^T (\vec{q}(t+1) - \vec{q}(t)). \tag{4.43}$$

Therefore, by substituting (4.41-4.42) into (4.38), and substituting (4.43) into (4.39), we have

$$\begin{aligned} & \mathcal{V}(\vec{y}(t+1), \vec{q}(t+1)) - \mathcal{V}(\vec{y}(t), \vec{q}(t)) \\ & \quad \leq -(\vec{q}(t+1) - \vec{q}(t))^T C_1 (\vec{q}(t+1) - \vec{q}(t)) - ||\vec{y}(t) - \vec{x}(t)||_V. \end{aligned}$$

where $C_1 = A^{-1} - 2EV^{-1}E^T$. If C_1 is positive definite, then

$$\begin{aligned} & \mathcal{V}(\vec{y}(t+1), \vec{q}(t+1)) - \mathcal{V}(\vec{y}(t), \vec{q}(t)) \\ \leq \; & -(\vec{q}(t+1) - \vec{q}(t))^T C_1 (\vec{q}(t+1) - \vec{q}(t)) - ||\vec{y}(t) - \vec{x}(t)||_V \leq 0. \end{aligned} \tag{4.44}$$

This proves that $\mathcal{V}(\vec{y}(t), \vec{q}(t))$ is non-increasing in t and hence must have a limit, i.e.,

$$\lim_{t \to \infty} ||\vec{q}(t) - \vec{q^*}||_A + ||\vec{y}(t) - \vec{y^*}||_{BV} = \mathcal{V}_0 \geq 0. \tag{4.45}$$

Therefore, the sequence $\{\vec{y}(t), \vec{q}(t), t = 1, ...\}$ is bounded, and there must exist a subsequence $\{\vec{y}(t_h), \vec{q}(t_h), h = 1, ...\}$ that converges to a limit point. Let $(\vec{y}_0, \vec{q}_0)$ be this limit. From (4.44), we have,

$$\lim_{h \to \infty} \vec{q}(t_h + 1) \;\; = \;\; \vec{q}_0 \text{ and } \lim_{h \to \infty} \vec{x}(t_h) = \vec{y}_0.$$

Taking limits at both sides of (4.33) as $h \to \infty$, we have,

$$\vec{q}_0 = [\vec{q}_0 + A(E\vec{y}_0 - R)]^+. \tag{4.46}$$

Hence

$$\vec{q}_0 \geq 0, \quad E\vec{y}_0 \leq R \text{ and } \vec{q}_0^T(E\vec{y}_0 - R) = 0. \tag{4.47}$$

Further, note that $\vec{x}(t_h)$ maximizes $L(\vec{x}, \vec{q}(t_h), \vec{y}(t_h))$ over all $\vec{x}$. Similar to Lemma 4.3.1, one can show that the mapping from $(\vec{y}(t_h), \vec{q}(t_h))$ to $\vec{x}(t_h)$ is continuous. Hence, taking limits as $h \to \infty$, we have

$$\vec{y}_0 \text{ maximizes } L(\vec{x}, \vec{q}_0, \vec{y}_0) \text{ over all } \vec{x} \ .$$

Therefore, $(\vec{y}_0, \vec{q}_0)$ is a stationary point of algorithm $\mathcal{A}$. We now replace $(\vec{y^*}, \vec{q^*})$ by $(\vec{y}_0, \vec{q}_0)$ in (4.45) and thus,

$$\lim_{t\to\infty} ||\vec{q}(t) - \vec{q}_0||_A + ||\vec{y}(t) - \vec{y}_0||_{BV} \quad = \quad \lim_{h\to\infty} ||\vec{q}(t_h) - \vec{q}_0||_A + ||\vec{y}(t_h) - \vec{y}_0||_{BV} = 0.$$

Hence $(\vec{y}(t), \vec{q}(t)) \to (\vec{y}_0, \vec{q}_0)$ as $t \to \infty$. Finally, it is easy to show that a sufficient condition for C_1 to be positive definite is $\max_l \alpha^l < \frac{1}{2\mathcal{SL}} \min_i c_i$ (see Lemma C.4 in Appendix C.2). ■

4.4 Convergence with Measurement Noise

In this section, we will study the convergence of algorithm $\mathcal{A}$ when there is measurement noise, i.e., when the dynamics of the system are governed by (4.11) and (4.12). The convergence of algorithm $\mathcal{A}$ will be established in the "stochastic approximation" sense, i.e., when the step-sizes are driven to zero in an appropriate fashion. To be specific, we replace the step-sizes α^l and β_i by

$$\alpha^l(t) = \eta_t \alpha_0^l, \quad \beta_i(t) = \eta_t \beta_{i,0},$$

for some positive sequence $\{\eta_t, t = 1, 2, ...\}$ that goes to zero as $t \to \infty$. For simplicity, we will focus on the case when $K = 1$ and we will drop the index k in (4.12). Let

$N(t) = [n^l(t), l = 1, ..., L]^T$. Use the vector notation from the previous section and define matrices A_0 and B_0 analogously as the matrices A and B, respectively, except that the diagonal elements are now filled with α_0^l and $\beta_{i,0}$. We can then rewrite algorithm $\mathcal{A}$ as:

Algorithm $\mathcal{AN}$:

A1-N) Let $\vec{x}(t) = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}(t), \vec{y}(t))$. Update the dual variables by

$$\vec{q}(t+1) = [\vec{q}(t) + \eta_t A_0(E\vec{x}(t) - R + N(t))]^+. \tag{4.48}$$

A2-N) Let $\vec{z}(t) = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}(t+1), \vec{y}(t))$. Update the primal variables by

$$\vec{y}(t+1) = \vec{y}(t) + \eta_t B_0(\vec{z}(t) - \vec{y}(t)).$$

Proposition 4.4.1 *If*

$$\sum_{t=1}^{\infty} \eta_t = \infty, \sum_{t=1}^{\infty} {\eta_t}^2 < \infty,$$

and

$$\mathbf{E}[N(t)|\vec{x}(s), \vec{y}(s), \vec{q}(s), s \leq t] = 0, \tag{4.49}$$

$$\sum_{t=1}^{\infty} {\eta_t}^2 \mathbf{E}||N(t)||^2 < \infty, \tag{4.50}$$

then algorithm $\mathcal{AN}$ will converge almost surely to a stationary point $(\vec{y^*}, \vec{q^*})$ of algorithm $\mathcal{A}$.

Assumption (4.49) simply states that the noise term $N(t)$ should be un-biased. Assumption (4.50) is also quite general. For example, it will hold if the variance of the noise, i.e., $\mathbf{E}[(n^l(t))^2]$ is bounded for all l and t. We can prove Proposition 4.4.1 by first extending the analysis of Proposition 4.3.2 to show that, as $t \to \infty$, $\mathcal{V}(\vec{y}(t), \vec{q}(t))$ converges almost surely to a finite non-negative number. This implies that $(\vec{y}(t), \vec{q}(t))$ is bounded almost surely. We can then use the ODE method of [66] to show that, as $t \to \infty$, the limiting behavior of the stochastic approximation algorithm will

converge to that of the ordinary differential equations defined by the continuous-time algorithm $\mathcal{AC}$ in Section 4.3 with $\hat{A} = A_0$ and $\hat{B} = B_0$. Proposition 4.3.1 can then be invoked to show that $(\vec{y}(t), \vec{q}(t))$ converges to a stationary point. Details of the proof are available in Appendix C.3.

We now comment on the step-size rule used in Proposition 4.4.1. As is typical for stochastic approximation algorithms, the convergence of algorithm $\mathcal{AN}$ is established when the step-sizes are driven to zero. When this type of stochastic approximation algorithms are employed online, we usually use step-sizes that are away from zero (e.g., constants). In this case, the trajectory $(\vec{y}(t), \vec{q}(t))$ (or $(\vec{x}(t), \vec{q}(t))$) will fluctuate in a neighborhood around the set of stationary points, instead of converging to one stationary point. In practice, we are interested in knowing how to choose the step-sizes so that the trajectory stays in a close neighborhood around the solutions. Since Proposition 4.4.1 requires that both α^l and β_i be driven to zero, we would expect that, if we were to choose both α^l and β_i small enough (but away from zero), the trajectory $(\vec{x}(t), \vec{q}(t))$ will be kept in a close neighborhood around the solutions. This choice of the step-sizes might seem overly conservative at first sight. In particular, since the noise terms $n^l(t)$ are only present in the dual update (4.12), it appears at first quite plausible to conjecture that only α^l needs to be driven to zero in Proposition 4.4.1 (in order to average out the noise), while β_i can be kept away from zero. If this conjecture were true, it would imply that, in order to keep the trajectory $(\vec{x}(t), \vec{q}(t))$ in a close neighborhood around the set of stationary points, only α^l needs to be small. However, our simulation results with constant step-sizes seem to suggest the opposite. We observe that, when there is measurement noise, the disturbance in the primal variables $\vec{x}(t)$ cannot be effectively controlled by purely reducing the step-sizes α^l at the links. We will elaborate on this observation in the next section with a numerical example, and we will show that the required step-size rule (i.e., both α^l and β_i needs to small) is again a consequence of the multi-path nature of the problem.

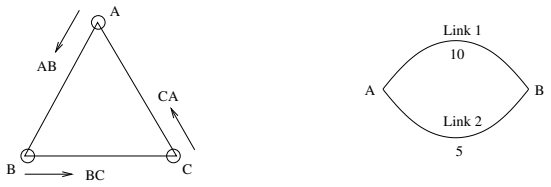


Fig. 4.1. Network topologies

4.5 Numerical Results

In this section, we present some simulation results for algorithm $\mathcal{A}$. For all simulations, we have chosen $K = 1$, i.e., we do not use the two-level convergence structure. We will use the setting of the multi-path flow control problem as an example, but the results here apply to other problems as well [22]. Specifically, x_{ij} corresponds to the data rate of user i on path j, and $E_{ij}^l = 1$ if path j of user i uses link l, $E_{ij}^l = 0$ otherwise. We first simulate the case when there is no measurement noise. We use the "Triangle" network in Fig. 4.1. There are three users (AB, BC, CA). For each user, there are two alternate paths, i.e., a direct one-link path (path 1), and an indirect two-link path (path 2). For example, user AB can take the one-link path $A \to B$ or the two-link path $A \to C \to B$. The utility functions for all three users are of the form:

$$f_i(\sum_{j=1}^{\theta(i)} x_{ij}) = w_i \ln(\sum_{j=1}^{\theta(i)} x_{ij}),$$

where w_i is the "weight" of user i, and x_{ij} is the data rate of user i on path j. We choose the weights as follows: $w_{AB} = 5.5, w_{BC} = 2.5, w_{CA} = 0.5$. The capacity on each link is 10 units.

Fig. 4.2 demonstrates the evolution over time of the implicit costs q^l and the users' data rates x_{ij}, respectively, for algorithms $\mathcal{A}$. We choose $c_i = 1.0$ for all users.

The step-sizes are $\alpha^l = 0.1$ for all links, and $\beta_i = 1.0$ for all users. We observe that all quantities of interest converge to the optimal solution, which is

$$q^{AB} = 0.425, \quad q^{BC} = 0.354, \quad q^{CA} = 0.071,$$
$$x_{AB,1} = 10, \quad x_{AB,2} = 2.94, \quad x_{BC,1} = x_{CA,1} = 7.06, \quad x_{BC,2} = x_{CA,2} = 0.$$

Note that at the stationary point, user AB will use both alternative paths while users BC and CA will only use the direct paths. Because the weight of the utility function of user AB is larger than that of the other users, algorithm $\mathcal{A}$ automatically adjusts the resource allocation of users BC and CA to give way to user AB.

Fig. 4.3 demonstrates the evolution of algorithm $\mathcal{A}$ for the same network when there is measurement noise. We assume that an *i.i.d.* noise term uniformly distributed within $[-2, 2]$ is added to each x_{ij} when each link estimates the aggregate load $\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} H_{ij}^l x_{ij}$. The step-sizes are $\alpha^l = 0.003$ for all links, and $\beta_i = 0.1$ for all users. We can observe that all quantities of interest eventually fluctuate around a small neighborhood of the solution.

We now investigate how the choice of the step-sizes α^l and β_i affect the level of fluctuation on the implicit costs and the users' data rates when there is measurement noise. We use a simpler "Two-Link" topology in Fig. 4.1. The capacity of the two links is 10 and 5, respectively. There is only one user, which can use both links. Its utility function is given by

$$f_i(x) = 5.5 \ln x.$$

The noise term $n^l(t)$ is *i.i.d.* and uniformly distributed within $[-2, 2]$.

Fig. 4.4 shows the evolution over time of the implicit costs (top) and that of the users' data rates (bottom) of algorithm $\mathcal{A}$ for different choices of the step-sizes. In the first three columns, we keep β unchanged and reduce the step-size α from 0.01 to 0.0001. We observe that, although the fluctuation in the implicit costs becomes smaller as the step-size α is reduced, the fluctuation in the data rates decreases only a little. *Note that the unit on the x-axis becomes larger as we move from the first column to the third column.* These figures indicate that, by reducing the step-size

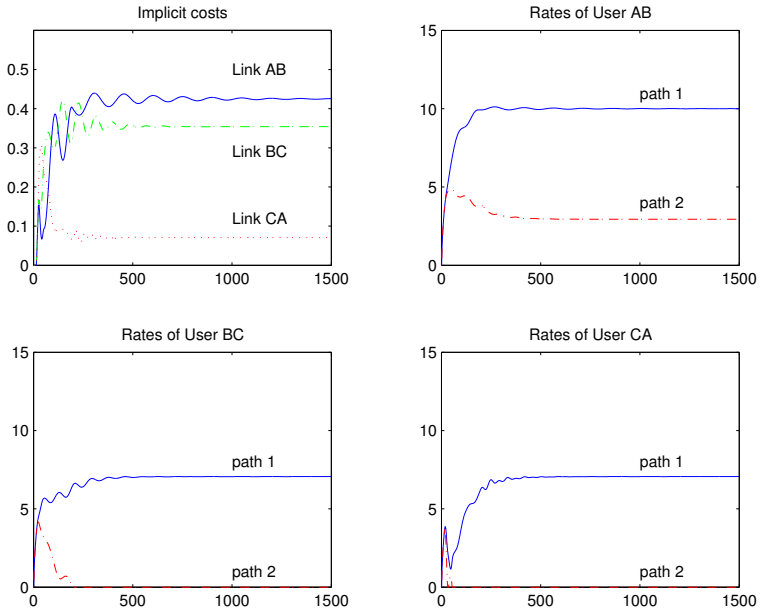


Fig. 4.2. Evolution of the implicit costs and the data rates when there is no measurement noise. $\alpha^l = 0.1$, $\beta_i = 1.0$.

Fig. 4.3. Evolution of the implicit costs and the data rates when there is measurement noise. $\alpha^l = 0.003$, $\beta_i = 0.1$.

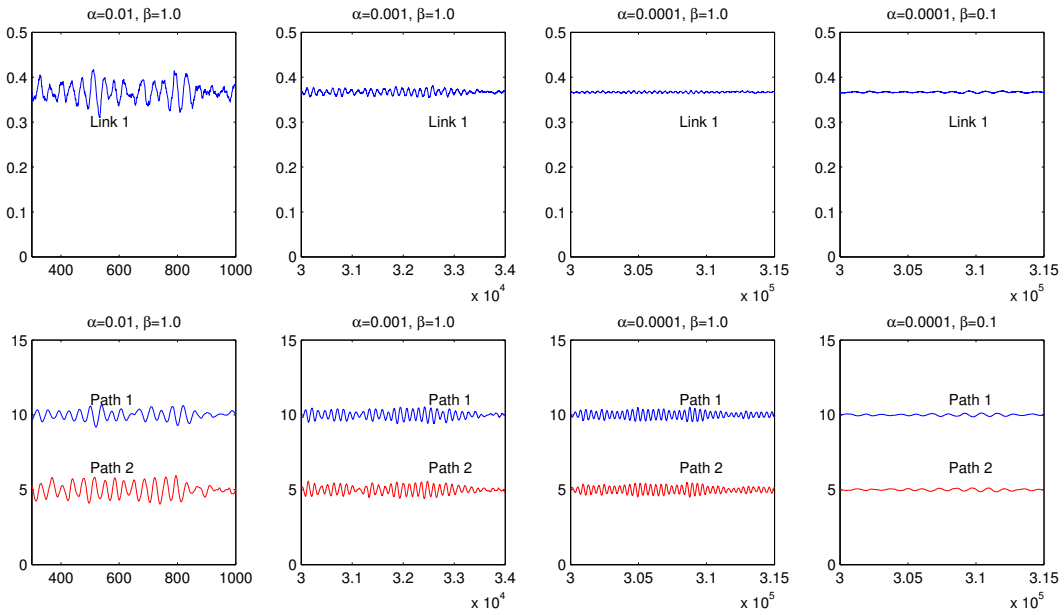


Fig. 4.4. Simulation of Algorithm A with measurement noise. Top: the implicit costs. Bottom: the data rates. *Note that the unit on the x-axis becomes larger as we move from the first column to the third column.*

α alone, the fluctuation in the data rates becomes slower, but the magnitude of the fluctuation changes little. In the fourth column, we decrease *both* β *and* α. The fluctuation in the data rates is now effectively reduced.

Although somewhat counter-intuitive, these observations are consistent with Proposition 4.4.1 where we require both α and β to be driven to zero for the convergence of the stochastic approximation algorithm to hold. As we will show next by studying the linearized version of the system, this step-size rule appears to be necessitated by the multi-path nature of the problem. Assume that algorithm $\mathcal{A}$ has a unique stationary point $(\vec{y^*}, \vec{q^*})$. We can linearize the continuous-time system (4.14)-(4.15) around this unique stationary point, and use Laplace transforms to study the frequency response of the system in the presence of noise $N(t)$. Without loss of generality, we can assume that $x_{ij}(t) > 0$ for all i, j and $q^l(t) > 0$ for all l. (Otherwise, we can eliminate the paths with $x_{ij}(t) = 0$ and the links with $q^l(t) = 0$

from the analysis because they do not contribute to the dynamics of the linearized system.) Let $\mathcal{X}(s)$ and $\mathcal{N}(s)$ denote the Laplace transform of the perturbation of $\vec{x}(t)$ and the noise $N(t)$, respectively. We can then compute the transfer function from $N(t)$ to $\vec{x}(t)$ as (see Appendix C.4 for the detail)

$$\mathcal{X}(s) = H(s)\mathcal{N}(s)$$

with

$$\begin{aligned} H(s) &= -\left\{\hat{B}^{-1}s\mathcal{I} + G + V^{-1}(\mathcal{I}-G)\frac{E^T\hat{A}E}{s}\hat{B}^{-1}(s\mathcal{I}+\hat{B})\right\}^{-1} \\ &\times \quad \hat{B}^{-1}(s\mathcal{I}+\hat{B})V^{-1}(\mathcal{I}-G)\frac{E^T\hat{A}}{s}, \end{aligned}$$

where the matrices $E, \hat{A}, \hat{B}$ and V are defined as in Section 4.3, $\mathcal{I}$ is the $\sum_{i=1}^{I}\theta(i) \times \sum_{i=1}^{I}\theta(i)$ identity matrix, $G = \mathbf{diag}\{G_i, i = 1, ..., I\}$ and each G_i is a $\theta(i)\times\theta(i)$ matrix whose elements are all

$$g_i = \frac{f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*)}{\theta(i)f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) - c_i}.$$

If the utility function $f_i(\cdot)$ is strictly concave, g_i will be positive. However, since each G_i has all identical elements, the matrix G is not invertible whenever some users have multiple paths. Its presence in the denominator of $H(s)$ turns out to be a source of instability. To see this, we compute $H(s)$ for the above Two-Link example. Note that

$$\begin{aligned} &E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad G = g\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \text{ for some } g > 0, \\ &\hat{A} = \alpha\mathcal{I}, \quad \hat{B} = \beta\mathcal{I}, \text{ and } V = c\mathcal{I}. \end{aligned}$$

Let $s = j\omega$. We have,

$$H(j\omega) = -\left\{\frac{j\omega}{\beta}\mathcal{I} + G + \frac{\alpha}{c}\frac{j\omega+\beta}{j\omega\beta}(\mathcal{I}-G)\right\}^{-1}\frac{\alpha}{c}\frac{j\omega+\beta}{j\omega\beta}(\mathcal{I}-G). \tag{4.51}$$

The terms in the denominator can be collected into

$$\left[\frac{\alpha}{\beta c}+\frac{j\omega}{\beta}(1-\frac{\alpha\beta}{c\omega^2})\right]\mathcal{I}+\left[(1-\frac{\alpha}{\beta c})+j\frac{\alpha}{c\omega}\right]G. \tag{4.52}$$

Since the matrix G is not invertible, if the terms that are associated with $\mathcal{I}$ is small, a "spike" in the transfer function $H(j\omega)$ will appear. This will happen when $\omega \approx \omega^*$, where ω^* is determined by

$$1-\frac{\alpha\beta}{c(\omega^*)^2}=0,$$

i.e.,

$$\omega^*=\sqrt{\frac{\alpha\beta}{c}}. \tag{4.53}$$

Substituting ω^* into (4.51) and (4.52) and assuming that $\alpha \ll \beta$, we have

$$\begin{aligned} H(j\omega^*) &\approx -\frac{\frac{\alpha}{c}(\frac{1}{\beta}+\frac{1}{j\omega})}{\frac{\alpha}{\beta c}}(\mathcal{I}-G) \\ &\approx j\frac{\frac{\alpha}{\omega c}}{\frac{\alpha}{\beta c}}(\mathcal{I}-G)\approx j\sqrt{\frac{c\beta}{\alpha}}(\mathcal{I}-G). \end{aligned} \tag{4.54}$$

We can draw the following conclusions from equations (4.53) and (4.54). If we keep β fixed and reduce α alone, the cutoff frequency ω^* will decrease with α. However, the gain $H(j\omega^*)$ at the cutoff frequency will increase! In Fig. 4.5, we plot $||H(j\omega)||_2$ with respect to ω for different values of α and β. We can easily observe the increased spike when α alone is reduced from 0.1 (the solid curve) to 0.001 (the dotted curve). If we further assume that $n^l(t)$ is white noise with unit energy, then the total energy of the fluctuation of $\vec{x}$ can be estimated by the area under the curve $||H(j\omega)||_2$ in Fig. 4.5. Due to the increased spike, the total energy of the fluctuation of $\vec{x}(t)$ will not decrease much when α alone is reduced, even though the frequency of the fluctuation becomes smaller. On the other hand, if we reduce β as well as α, the gain at the cutoff frequency ω^* will remain the same as the cutoff frequency itself decreases (shown as the dashed curve in Fig. 4.5 when β is also reduced to 0.001). Hence, the total energy of the fluctuation in $\vec{x}(t)$ is effectively reduced. These conclusions are thus consistent with our simulation results in Fig. 4.4. Therefore, the step-size rule (i.e., both α^l and β_i needs to be reduced) is necessary to address the potential instability in the system due to the multi-path nature of the problem.

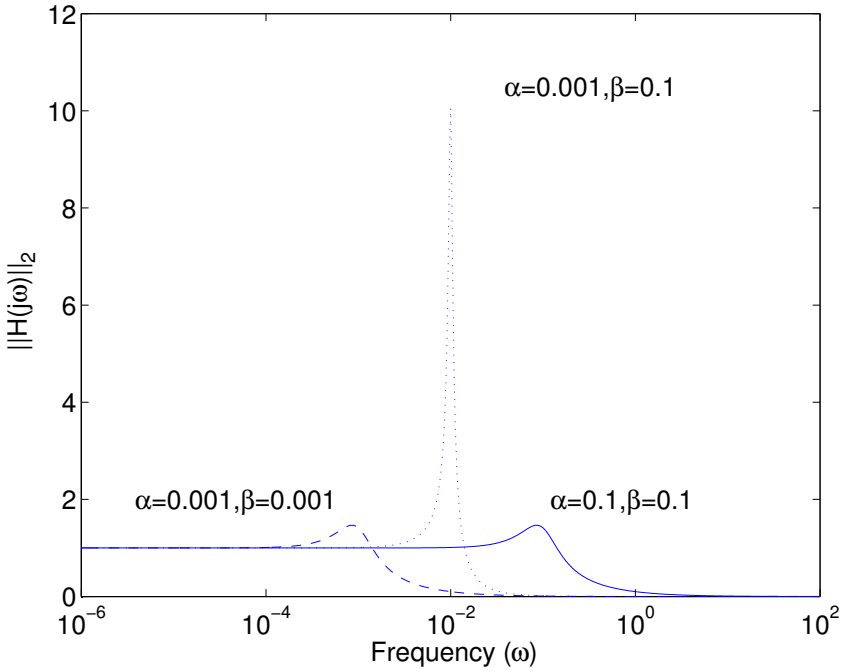


Fig. 4.5. The frequency response $||H(j\omega)||_2$

4.6 Concluding Remarks

In this chapter, we have developed a distributed algorithm for the utility maximization problem in communication networks that have the capability to allow multi-path routing. We have studied the convergence of our algorithm in both continuous-time and discrete-time, with and without measurement noise. We have shown how the multi-path nature of the problem can potentially lead to difficulties such as instability and oscillation, and our analyses provide important guidelines on how to choose the parameters of the algorithm to address these difficulties and to ensure efficient network control. When there is no measurement noise, our analysis gives easy-to-verify conditions on how large the step-sizes of the algorithm can be while still ensuring convergence. When there is measurement noise, we find that the step-sizes in the updates of both the user algorithm and the network component algorithm have to decrease at the same time in order to reduce the fluctuation of the resource allocation. Reducing only the step-sizes in the update of the network component algorithm will reduce the frequency of the fluctuation, but not necessarily its magnitude. These guidelines are confirmed by our simulation results.

We briefly discuss possible directions for future work. The analysis in this chapter has focused on the case when all computation is synchronized. An interesting problem is to study the convergence and stability of the algorithm when the computation is asynchronous and when feedback delays are non-negligible. Simulations suggest that our distributed algorithm may still be used in those situations, however, the step-size rules may need to change. Another direction is to extend our solution to resource allocation problems in wireless networks. In wireline networks, the resource constraints of different network components are orthogonal to each other. In wireless networks, however, the capacity of a link is a function of the signal to interference ratio, which depends not only on its own transmission power, but also on the power assignments at other links. Hence, the resource constraints in wireless networks are of a more complex form than that of wireline networks. It would be

interesting to see whether the results of this chapter can be extended to multi-path utility maximization problems in wireless networks.

5. CAPACITY-DELAY TRADEOFF IN LARGE MOBILE WIRELESS NETWORKS

5.1 Introduction

In Chapters 2-4, we have studied the simplification of network control in communication networks with large capacity. By exploiting the largeness of these networks, we have developed simple and efficient control algorithms for pricing-based network control and for Quality-of-Service routing. Note that the large-capacity paradigm is suitable for wire-line networks. In fact, due to the advance in fiber-optic technology, it is not uncommon for the backbone links in today's wire-line networks (such as the Internet) to have Gibabits-per-second or even Terabits-per-second capacity. Therefore, the results that we have developed in the earlier chapters can be readily applied to large-capacity wire-line networks.

In this chapter and the next chapter, we will turn to wireless networks, where the situation is quite different. In wireless networks, the network capacity is fundamentally limited by the radio spectrum, and thus is usually not very large. Hence, the results and techniques of the previous chapters do not directly apply to wireless networks.

Can we still obtain simplicity results for wireless networks? In the following chapters we will address this question. The answer is yes, however, we need to take some new approaches that are different from those in the previous chapters. In this chapter, we will study simplicity due to largeness in terms of the number of nodes. Note that in wireless ad hoc and sensor networks, the number of nodes can be quite large. We will show that this type of largeness can also lead to simple and critical insights in control. In particular, we will exploit this type of largeness

to obtain simple relationships that characterize the fundamental tradeoff between the capacity and the delay in large mobile wireless networks. Then, in the next chapter, we will study how to simplify control by designing an appropriate system architecture for multihop wireless systems.

5.1.1 Capacity and Delay in Mobile Wireless Networks

In this chapter, we are interested in the performance study of wireless networks that have the *multihop* communication capability, i.e., terminals can use each other as relays. This is in contrast to cellular systems where terminals always communicate directly with the base-station (i.e., by one hop). Studies have shown that the capacity of a wireless network can be substantially improved by using multi-hop communications [71, 72]. These multihop networks can either be used to extend the coverage of an existing infrastructure, or to deploy in areas where an infrastructure can not be established (e.g., as ad hoc networks for military or disastrous scenarios, or as sensor networks). They can be either static networks (i.e., nodes do not move), or mobile networks (i.e., nodes move from time to time). Our work in this chapter will focus on mobile networks.

Among the many problems in the research of multihop wireless network, capacity and delay are two of the most important ones. The capacity of the network determines how much information can be carried by the network, and the packet delay can significantly affect the performance of many applications as well. The difficulty in studying the capacity and delay of multihop wireless networks lies in the complexity. Although one can develop deterministic algorithms that solve for the optimal capacity [72–74], the complexity of these algorithms is usually too high once the number of nodes in the network is large.

In the past few years, there have been some advances in using asymptotic methods to understand the capacity and delay of large wireless networks [23, 71, 75–85]. Although an asymptotical analysis usually does not give the precise optimum, it

is still an attractive approach for two reasons. First, an asymptotical analysis can remain mathematically tractable even when the number of nodes is large. Secondly, an asymptotical analysis is usually able to reveal the most important tradeoffs in the system, which will then provide us with insights on how to design simple control algorithms that can remain effective even when the network is large.

For a static network (where nodes do not move), Gupta and Kumar have shown that the per-node capacity decreases as $O(1/\sqrt{n \log n})$[1] as the number of nodes n increases [71]. The capacity of wireless networks can be improved when *mobility* is taken into account. When the nodes are mobile, Grossglauser and Tse show that per-node capacity of $\Theta(1)$ is achievable [23], which is much better than that of static networks. This capacity improvement is achieved at the cost of excessive packet delays. In fact, it has been pointed out in [23] that the packet delay of the proposed scheme could be unbounded.

There have been several recent studies that attempt to address the relationship between the achievable capacity and the packet delay in mobile wireless networks. In fact, our work is first motivated by the research on this problem under the *i.i.d.* mobility model [75]. In the work by Neely and Modiano [75], it was shown that the maximum achievable per-node capacity of a mobile wireless network is bounded by $O(1)$. Under an *i.i.d.* mobility model, the authors of [75] present a scheme that can achieve $\Theta(1)$ per-node capacity and incur $\Theta(n)$ delay, provided that the load is strictly less than the capacity. Further, they show that it is possible to reduce packet delay if one is willing to sacrifice capacity. In [75], the authors formulate and prove a fundamental tradeoff between the capacity and delay. Let the average

[1]We use the following notation throughout:

$$\begin{aligned}
f(n) = o(g(n)) &\leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0, \\
f(n) = O(g(n)) &\leftrightarrow \limsup_{n\to\infty} \frac{f(n)}{g(n)} < \infty, \\
f(n) = \omega(g(n)) &\leftrightarrow g(n) = o(f(n)), \\
f(n) = \Theta(g(n)) &\leftrightarrow f(n) = O(g(n)) \text{ and } g(n) = O(f(n)).
\end{aligned}$$

end-to-end delay be bounded by D. For D between $\Theta(1)$ and $\Theta(n)$, [75] shows that the maximum per-node capacity λ is upper bounded by

$$\lambda \leq O(\frac{D}{n}). \tag{5.1}$$

The authors of [75] develop schemes that can achieve $\Theta(1)$, $\Theta(1/\sqrt{n})$, and $\Theta(1/(n \log n))$ per-node capacity, when the delay constraint is on the order of $\Theta(n)$, $\Theta(\sqrt{n})$, and $\Theta(\log n)$, respectively.

Inequality (5.1) leads to the *pessimistic* conclusion that a mobile wireless network can sustain at most $O(1/n)$ per-node capacity with a constant delay bound. This capacity is even worse than that of static networks. It turns out that this pessimistic conclusion is due to certain restrictive assumptions that are implicit in the work in [75] (we will elaborate on these assumptions in Section 5.7). In fact, Toumpis and Goldsmith [76] present a scheme that can achieve a per-node capacity of $\Theta(n^{(d-1)/2}/\log^{5/2} n)$ when the delay is bounded by $O(n^d)$. The result of [76] has incorporated the effect of fading. If we remove fading, the per-node capacity will be of the order $\Theta(n^{(d-1)/2}/\log^{3/2} n)$. Ignoring the logarithmic term, we find that in [76] the following capacity-delay tradeoff is achievable:

$$\lambda^2 = \Theta(\frac{D}{n}). \tag{5.2}$$

This is better than (5.1). In particular, the authors of [76] present a scheme that can achieve $\Theta(1/(\sqrt{n}\log^{3/2} n))$ per-node capacity with a constant delay bound. (The capacity will be $\Theta(1/(\sqrt{n \log n}))$ with no fading.) This capacity is now *comparable* to that of the static wireless networks.

An open question that still remains is: *what is the optimal capacity-delay tradeoff in mobile wireless networks*? Inequality (5.1) is clearly not optimal. Without a careful study of the various inherent tradeoffs within the system, a constructive methodology such as the one in [76] will only produce a lower bound like (5.2). Note that the search for the optimal capacity-delay tradeoff is important for two reasons. First, it will allow us to see where the fundamental limits (i.e., upper bounds) are,

and how far existing schemes could possibly be improved. Secondly, as has happened in previous works [71], a careful study of the upper bound is usually able to reveal the delicate tradeoffs inherent to the problem. A complete understanding of these tradeoffs will help us identify the possible points of inefficiency in existing schemes and provide directions for further improvement. The ultimate goal is to find a scheme that can achieve the optimal capacity-delay tradeoff.

In this chapter, we will use a systematic methodology to investigate the fundamental tradeoff between the capacity and the delay in mobile wireless networks. Our methodology is as follows. We first identify several key parameters of a general class of scheduling schemes, and investigate the inherent tradeoffs among the capacity, the delay, and these scheduling parameters. Based on these inherent tradeoffs (in the form of inequalities), we are then able to compute an upper bound on the maximum per-node capacity of a large mobile wireless network under given delay constraints. In the process of proving the upper bound, we are also able to identify the optimal values of the key scheduling parameters. Knowing these optimal values, we can then develop scheduling schemes that achieve the upper bound up to some logarithmic factor, which suggests that our upper bound is tight. We have applied this methodology to three different mobility models, i.e., the *i.i.d.* mobility model [24, 75, 76], the random way-point mobility model [84, 85], and the Brownian-motion mobility model [83, 84, 86]. For the *i.i.d.* mobility model, the inherent tradeoffs that our methodology is based on can be analytically established [24–26]. For the random way-point mobility model, we use a combination of analytical and numerical techniques to establish these inherent tradeoffs [25]. In both cases, we are able to obtain new insights on the optimal choices of the key scheduling parameters, and develop new scheduling schemes that can achieve larger capacity than previous proposals under the same delay constraints. For example, under the *i.i.d* mobility model, we can achieve a new capacity-delay tradeoff of

$$\lambda^3 \geq \Theta(\frac{\bar{D}}{n}/\log^{9/2} n). \tag{5.3}$$

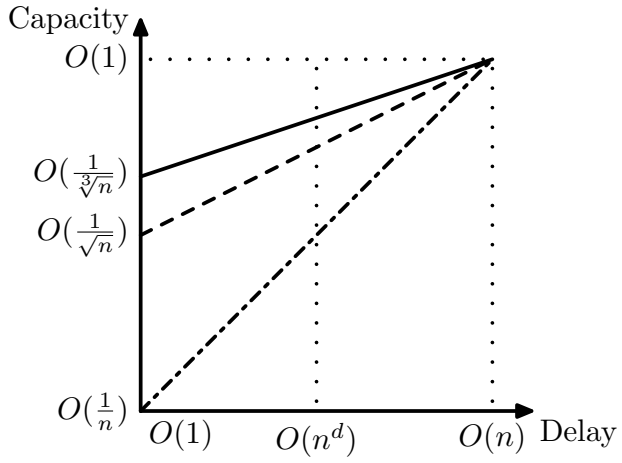


Fig. 5.1. Comparison of the achievable capacity-delay tradeoffs (ignoring the logarithmic terms).

In Fig. 5.1, we draw this new tradeoff (the top line) alongside the capacity-delay tradeoffs achieved by the schemes in [75] and [76] (the bottom line and the middle line, respectively). Our new scheme clearly achieves larger capacity when the delay constraints are small. In particular, when the delay is bounded by a constant, our scheme can achieve $\Theta(n^{-1/3}/\log^{3/2} n)$ per-node capacity. Unlike previous works, this result shows that, even for a constant delay bound, the per-node capacity of mobile wireless networks can be larger than that of the static networks! Our methodology can be extended to incorporate additional constraints of the scheduling schemes. Finally, for the Brownian-motion mobility model, we will find that the delay-capacity tradeoff is radically different from those under the *i.i.d.* mobility model and the random way-point mobility model [27].

The rest of the chapter is structured as follows. In Section 5.2, we outline the network and mobility models. We will first focus on the *i.i.d.* mobility model in Sections 5.3-5.7. In Section 5.3, we outline the general class of scheduling policies that we will consider. We then identify a number of key scheduling parameters and study their inherent tradeoffs in Section 5.4. We establish the upper bound on the optimal capacity-delay tradeoff in Section 5.5 and present a scheme in Section 5.6

that achieves a capacity-delay tradeoff close to the upper bound. In Section 5.7, we discuss how to treat additional scheduling constraints such as those that appear in previous works [75,76]. The capacity-delay tradeoff under the random way-point mobility model is studied in Section 5.8, and that under the Brownian-motion mobility model is studied in Section 5.9. Then we conclude.

5.2 Network and Mobility Model

We consider a mobile wireless network with n nodes moving within a certain area with unit size. For simplicity, we assume the following traffic model similar to the models in [75, 76]. We assume that the number of nodes n is even and the nodes can be labeled in such a way that node $2i-1$ communicates with node $2i$, and node $2i$ communicates with node $2i-1$, $i = 1, 2, ..., n/2$. The communication between any source-destination pairs can go through multiple other nodes as *relays*. That is, the *source* can either send a message directly to the *destination*; or, it can send the message to one or more *relay* nodes; the relay nodes can further forward the message to other relay nodes (possibly after moving to another position); and finally some relay node forwards the message to the destination.

We assume the following Protocol Model from [71] that governs *direct radio transmissions* between nodes. Let W be the bandwidth of the system. Let X_i denote the position of node i, $i = 1, ..., n$. Let $|X_i - X_j|$ be the Euclidean distance between nodes i and j. At any time, node i can communicate *directly* with another node j at W bits per second if and only if the following interference constraint is satisfied [71]:

$$|X_j - X_k| \geq (1 + \Delta)|X_i - X_j|$$

for every other node $k \neq i, j$ that is simultaneously transmitting. Here, Δ is some positive number. Note that an alternative model for direct radio transmission is the Physical Model [71, 76]. In the Physical Model, a node can communicate with another node if the signal-to-interference ratio is above a given threshold. It has been shown that, under certain conditions, the Physical Model can be reduced to

the Protocol Model with an appropriate choice of Δ [71]. Hence, we will not consider the Physical Model any further in this chapter. We also assume that no nodes can transmit and receive over the same frequency at the same time.

We will study three types of mobility models.

1) **The *i.i.d.* Mobility Model:** In the *i.i.d* mobility model [75], the time is divided into slots of unit length. At each time slot, the positions of each node are *i.i.d.* and uniformly distributed within a unit square[2]. Between time slots, the distributions of the positions of the nodes are independent. Although the *i.i.d.* mobility model is somewhat restrictive in assuming the distribution of the node positions to be independent across time slots, its mathematical tractability allows us to gain important insights into the structure of the problem, which can then be extended to other more realistic mobility models.

2) **The Random Way-point (*RWP*) Mobility Model:** In the random way-point (*RWP*) mobility model, we assume that nodes move within a unit square and the unit square wraps around like a *torus*, i.e., a node can move out of the unit square from an edge and immediately move into the unit square from the opposite edge[3]. The initial positions of the nodes at time $t = 0$ are *i.i.d.* and uniformly distributed within the unit square. Each node then moves independently in *trips*: for each *trip*, the node picks a target position uniformly distributed within the unit square, and moves towards the target position along the shortest path at a constant speed v. (Note that since the unit square is a torus, the shortest path may not always be the straight line.) When the node reaches the target position, it immediately starts another *trip* by picking a new target position randomly. Unlike [79], we assume that, when a node is picked as the relay node for a message, the information about the future motion of the relay node is not available to the scheduler. (On the other hand, the scheduling scheme in [79] has exploited this knowledge to obtain a different

[2]Note that changing the shape of the area from a square to a circle or other topologies will not affect our main results for the *i.i.d.* mobility model.

[3]The assumption of a torus could be removed. It is included here for mathematical convenience so that we do not need to deal with the edge effects.

capacity-delay tradeoff than ours under a somewhat similar *uniform mobility model*.) Following the convention in related studies [84], we assume that the speed v scales as $v(n) = \Theta(1/\sqrt{n})$ when the number of nodes n increases[4].

3) **The Brownian Motion Mobility Model**: In the Brownian-motion model, we assume that nodes move independently on the surface S of a unit sphere as in [84]. (A similar Brownian motion model on a 2-d torus is also considered in [83].) It is easier to describe the motion of each node using the spherical coordinates. Let θ_t and ϕ_t denote the colatitude and longitude, respectively, of the position of a particular node at time t ($0 \leq \theta_t \leq \pi$ and $0 \leq \phi_t < 2\pi$). When a node moves according to the Brownian motion model on the unit sphere S, the (Itô) stochastic differential equations for the process (θ_t, ϕ_t) are given by [87]:

$$d\theta_t = \sigma_n dB_t + \frac{\sigma_n^2}{2\tan\theta_t} dt, \tag{5.4}$$

and

$$d\phi_t = \frac{\sigma_n}{\sin\theta_t} dB'_t, \tag{5.5}$$

where B_t and B'_t are independent standard one-dimensional Brownian motions (i.e., with variance 1). We call σ_n^2 the *variance* of the Brownian Motion described in (5.4) and (5.5). We assume that the initial positions of the nodes are *i.i.d.* and uniformly distributed on the unit sphere. This implies that the positions of the nodes will remain uniform at all times.

Under all three mobility models, we assume the following *separation of time scales*, i.e., radio transmission can be scheduled at a time scale much faster than that of node mobility. This is usually a reasonable assumption in real networks. Hence, a message may be divided into multiple bits and each bit can be forwarded "instantaneously" across multiple hops as if the positions of all nodes are frozen.

We assume a uniform traffic pattern, that is, all source nodes communicate with their destination nodes at the same rate λ. Let $\bar{D}$ be the mean delay averaged over

[4]It is also possible to extend our methodology to the case when the speed is randomly distributed between $[v(n), cv(n)]$ for some $c > 1$, and to the case when nodes pause between trips.

all messages and all source-destination pairs. Both λ and $\bar{D}$ will depend on how the transmissions between mobile nodes are scheduled. We are interested in capturing the fundamental tradeoff between the achievable capacity λ and the delay $\bar{D}$. That is, over all possible ways of scheduling the radio transmissions, what is the maximum per-node capacity λ given certain constraint on the delay $\bar{D}$.

5.3 The Class of Scheduling Policies

In Sections 5.3-5.7, we will focus on the *i.i.d* mobility model, and we will defer the study of the other mobility models until later sections. In this section, we define the class of scheduling policies that we will consider for the *i.i.d.* mobility model. Because we are interested in the fundamental *achievable* capacity for given delay constraints, we will assume that there exists a scheduler that has all the information about the current and past status of the network, and can schedule any radio transmission in the current and future time slots. At each time slot t, for each bit b that has not been delivered to its destination yet, the scheduler needs to perform the following two functions:

- *Capture:* The scheduler needs to decide whether to deliver the bit b to the destination within the current time slot. If yes, the scheduler then needs to choose one relay node (possibly the source) that has a copy of the bit b at the beginning of the time slot t, and schedule radio transmissions to forward this bit to the destination *within the same time slot*, using possibly multi-hop transmissions. When this happens successfully, we say that the chosen relay node has successfully *captured* the destination of bit b, or a successful *capture* has occurred for bit b.

- *Replication:* If capture does not occur for bit b, the scheduler needs to decide whether to *replicate* bit b to other nodes that do not have the bit at the beginning of the time slot t. The scheduler also needs to decide which nodes

to relay from and relay to, and how to schedule radio transmissions to forward the bit to these new relay nodes.

The *capture* function and the *replication* function are mutually exclusive for a given bit b. Once a successful *capture* occurs, the bit b will be delivered to its destination within the same time slot, and hence can exit the system. Note that once a successful capture occurs, it is important to forward the bit b to the destination within a single time slot. Otherwise, since the chosen relay node may move arbitrarily far away from the destination in the next time slot, the nodes that received the bit b in the current time slot will only count as new relay nodes for the bit b, and they have to capture again in the next time slot.

In this chapter, we will consider the class of *causal* scheduling policies that perform the above two functions at each time slot. The causality assumption essentially requires that, when the scheduler makes the capture decision and the replication decision, it can only use information about the current and the past status of the network. In particular, at any time slot t, the scheduler cannot use information about the *future* positions of the nodes at any time slot $s > t$.

This class of scheduling policies is clearly very general, and encompasses nearly any practical scheduling scheme we can think of. (Note that even *predictive* scheduling schemes have to rely on current and past information only.) Some remarks on the *capture* process are in order. Although we do allow for other less intuitive alternatives, in a typical scheduling policy a successful *capture* usually occurs when some relay nodes are within an area close to the destination node, so that fewer resources will be needed to forward the information to the destination. For example, a relay node could enter a disk of a certain radius around the destination, or a relay node could enter the same cell as the destination. We call such an area a *capture neighborhood*. The relay nodes that has the bit b at the beginning of the time slot t are called *mobile relays* for bit b. The mobile relay that is chosen to forward the bit b to the destination is called the *last mobile relay* for bit b.

The following examples are illustrative of the possible scheduling policies within this broad class. The schemes in previous works are all special cases or variants of these examples.

Example A: The number of mobile relays R is fixed and the capture neighborhood is chosen to be a disk with a fixed radius ρ around the destination. Once a bit b enters the system, it is immediately broadcast to the nearest $R-1$ neighboring nodes. When any of the R mobile relays (including the source node) move within distance ρ from the destination, the bit b is then forwarded from the nearest mobile relay to the destination.

Example B **(The Cell-based Scheme):** The unit area is divided into a number of cells. Once a bit b enters the system, it is immediately broadcast to all other nodes in the same cell. The number of mobile relays for the bit b then stay unchanged. Note that the actual number of mobile relays depends on the number of nodes that reside in the same cell as the source (at the time slot when the bit b enters the system), and is thus a random variable. When one of the mobile relays moves into the same cell as the destination, the bit b is then forwarded from the nearest mobile relay to the destination.

Example C: In the above two schemes, no replication for bit b is carried out except at the first time slot when the bit enters the system. A more sophisticated strategy is to use an "opportunistic replication scheme" such as the example below. The unit area is divided into a number of cells. After a bit b enters the system, at each time slot t, if one of the mobile relays moves into the same cell as the destination, bit b is then forwarded from the nearest mobile relay to the destination. Otherwise, the source node (or, alternatively, the current mobile relays) broadcasts the bit to all other nodes that reside at the same cell. Hence, replication may occur at each time slot until bit b is delivered to its destination.

5.4 Inherent Tradeoffs Among the Key Scheduling Parameters

In the sequel, we will prove several key inequalities that capture the various tradeoffs inherent in this broad class of scheduling policies. Intuitively, the larger the number of mobile relays and the larger the capture neighborhood, the smaller the delay. On the other hand, in order to improve capacity, we need to consume fewer radio resources, which implies a smaller number of mobile relays and a shorter distance from the last mobile relay to the destination. As we will see later, these tradeoffs will determine the fundamental relationship between achievable capacity and delay in mobile wireless networks.

We first define three parameters R_b, D_b, and l_b that are the key to characterize the various tradeoffs in the system.

5.4.1 Notations

Let $(\Omega, \mathcal{F}, P)$ be the probability space on which the random mobility of the mobile nodes is defined. Let $X(i,t)$ be the random variable that denotes the position of node i at time slot t. Let b denote a bit that needs to be communicated from a source node $S(b)$ to destination node $D(b)$. Let $t_0(b)$ be the time slot when bit b first enters the system. Let $I_b(i,t)$ be an indicator function, where $I_b(i,t) = 1$ if node i has a copy of bit b at the beginning of time slot t, $I_b(i,t) = 0$ otherwise. By definition, $I_b(S(b), t_0(b)) = 1$, and $I_b(i,t) = 0$ for all i and $t < t_0(b)$. Let $\mathcal{F}_t$ be the σ-algebra generated by the random variables $X(i,s)$ and $I_b(i,s)$ for all $s \leq t$. Hence $\{\mathcal{F}_t, t = 0, 1, ...\}$ is a filtration [88, p231] and $\mathcal{F}_t$ captures all information about the "history" up to time slot t.

Fix any scheduling policy and fix a bit b that enters the system at time slot $t_0(b)$. For any time slot $t \geq t_0(b)$, let $C_b(t) = 1$ if the scheduler decides that a successful capture occurs at this time slot. $C_b(t) = 0$, otherwise. If $C_b(t) = 1$, the scheduler then picks one mobile relay that has a copy of the bit b at the beginning of the time slot to forward the bit towards the destination *within the same time slot* t,

using possibly multi-hop transmissions. Let $\tilde{l}_b(t)$ be the distance from the chosen mobile relay to the destination of the bit b. Let $\tilde{l}_b(t) = \infty$ if $C_b(t) = 0$. Finally, let $r_b(t+1)$ denote the number of mobile relays holding the bit b *at the end* of the time slot t, i.e., $r_b(t+1)$ is the cardinality of the set $\{i : I_b(i, t+1) = 1\}$. Since the random variables $C_b(t), \tilde{l}_b(t)$ and $r_b(t+1)$ are all outcomes of the scheduling policy, the causality assumption implies that they are all $\mathcal{F}_t$-measurable[5].

Let

$$s_b \triangleq \min\{t : t \geq t_0(b) \text{ and } C_b(t) = 1\}$$

be the first time when a successful capture for bit b occurs. Thus s_b is a stopping time [88, p234] with respect to the filtration $\{\mathcal{F}_t, t = 0, 1, ...\}$. Let $R_b \triangleq r_b(s_b)$ denote the number of mobile relays holding the bit b at the time of capture. Let $D_b \triangleq s_b - t_0(b)$ denote the number of time slots from the time bit b enters the system to the time of capture. Let $l_b \triangleq \tilde{l}_b(s_b)$ denote the distance from the chosen last mobile relay node to the destination. The quantities R_b, D_b, and l_b are essential for the tradeoffs that follow. Note that D_b includes possible queueing delays at the source node or at the relay nodes.

We are now ready to state the inherent tradeoffs among capacity, delay and these key parameters. In this section, we will state these tradeoffs precisely for the *i.i.d.* mobility model, and defer the discussion on other mobility models until Sections 5.8 and 5.9.

5.4.2 Tradeoff I : D_b versus R_b and l_b

Proposition 5.4.1 *Under the* i.i.d. *mobility model, the following inequality holds for any causal scheduling policy when $n \geq 3$,*

$$c_1 \log n \mathbf{E}[D_b] \geq \frac{1}{(\mathbf{E}[l_b] + \frac{1}{n^2})^2 \mathbf{E}[R_b]} \text{ for all bits } b, \tag{5.6}$$

where c_1 is a positive constant.

[5]Here we have excluded *probablistic* scheduling policies. Otherwise, $\mathcal{F}_t$ should be augmented with a σ-algebra that is independent of node mobility in future time slots.

The proof is available in Appendix D.1. This new result is one of the cornerstones for deriving the optimal capacity-delay tradeoff in mobile wireless networks. It captures the following tradeoff: the smaller the number R_b of mobile relays the bit b is replicated to, and the shorter the targeted distance l_b from the last mobile relay to the destination, the longer it takes to capture the destination. This seemingly odd relationship is actually motivated by some simple examples. Consider Example A in Section 5.3. When R_b and the area of the capture neighborhood A_b are constants, then $1-(1-A_b)^{R_b}$ is the probability that any one out of the R_b nodes can capture the destination in one time slot. It is easy to show that, the average number of time slots needed before a successful capture occurs, is,

$$\mathbf{E}[D_b] = \frac{1}{1-(1-A_b)^{R_b}} \geq \frac{1}{A_b R_b}.$$

If, as in Example B, R_b and possibly A_b are random but *fixed after the first time slot* $t_0(b)$, then

$$\mathbf{E}[D_b|R_b, A_b] \geq \frac{1}{A_b R_b}.$$

By Hőlder's Inequality [88, p15],

$$\mathbf{E}^2[\frac{1}{\sqrt{A_b}}] \leq \mathbf{E}[R_b]\mathbf{E}[\frac{1}{A_b R_b}].$$

Hence,

$$\begin{aligned}\mathbf{E}[D_b] &\geq \mathbf{E}[\frac{1}{A_b R_b}] \geq \mathbf{E}^2[\frac{1}{\sqrt{A_b}}]\frac{1}{\mathbf{E}[R_b]} \\ &\geq \frac{1}{\mathbf{E}^2[\sqrt{A_b}]\mathbf{E}[R_b]},\end{aligned}$$

where in the last step we have applied Jensen's Inequality [88, p14]. Note that on average l_b is on the order of $\sqrt{A_b}$. Hence,

$$\mathbf{E}[D_b] \geq \frac{c_1'}{\mathbf{E}^2[l_b]\mathbf{E}[R_b]} \text{ for all bits } b, \tag{5.7}$$

where c_1' is a positive constant. It may appear that, when an "opportunistic replication scheme" such as the one in Example C is employed, such a scheme might achieve a better tradeoff than (5.7) by starting off with fewer mobile relays and a

smaller capture neighborhood, if the node positions at the early time slots after the bit's arrival turns out to be favorable. However, Proposition 5.4.1 shows that no scheduling policy can improve the tradeoff by more than a $\log n$ factor. For details, please refer to Appendix D.1.

5.4.3 Tradeoff II : Multihop

Once a successful capture occurs, the chosen mobile relay (i.e., the *last mobile relay*) will start transmitting the bit to the destination *within a single time slot*, using possibly other nodes as relays. We will refer to these latter relay nodes as *static relays*. The static relays are only used for forwarding the bit to the destination *after a successful capture occurs*. Let h_b be the number of hops it takes from the last mobile relay to the destination. Let S_b^h denote the transmission range of each hop $h = 1, .., h_b$. The following relationship is trivial.

Proposition 5.4.2 *The sum of the transmission ranges of the h_b hops must be no smaller than the straight-line distance from the last mobile relay to the destination, i.e.,*

$$\sum_{h=1}^{h_b} S_b^h \geq l_b. \tag{5.8}$$

5.4.4 Tradeoff III : Radio Resources

It consumes radio resources to replicate each bit to mobile relays and to forward the bit to the destination. Proposition 5.4.3 below captures the following tradeoff: the larger the number of mobile relays R_b and the further the multi-hop transmissions towards the destination have to traverse, the smaller the achievable capacity. Consider a large enough time interval T. The total number of bits communicated end-to-end between all source-destination pairs is λnT.

Proposition 5.4.3 *Assume that there exist positive numbers c_2 and N_0 such that $D_b \leq c_2 n^2$ for $n \geq N_0$. If the positions of the nodes within a time slot are* i.i.d.

and uniformly distributed within the unit square, then there exist positive numbers N_1 *and* c_3 *that only depend on* c_2, N_0 *and* Δ*, such that the following inequality holds for any causal scheduling policy when* $n \geq N_1$*,*

$$\sum_{b=1}^{\lambda nT} \frac{\Delta^2}{4} \frac{\mathbf{E}[R_b] - 1}{n} + \mathbf{E}[\sum_{b=1}^{\lambda nT} \sum_{h=1}^{h_b} \frac{\pi \Delta^2}{4} (S_b^h)^2] \leq c_3 WT \log n. \tag{5.9}$$

The assumption that $D_b \leq c_2 n^2$ for large n is not as restrictive as it appears. It has been shown in [75] and [84] that the maximal achievable per-node capacity is $\Theta(1)$ and this capacity can be achieved with $\Theta(n)$ delay under both the *i.i.d.* mobility model and the random way-point mobility model. Hence, we are most interested in the case when the delay is not much larger than the order $\Theta(n)$. Further, Proposition 5.4.3 only requires that the stationary distribution of the positions of the nodes within a time slot is *i.i.d.* It does not require the distribution between time slots to be independent.

We briefly outline the motivation behind the inequality (5.9). The details of the proof are quite technical and available in Appendix D.2. Consider nodes i, j that directly transmit to nodes k and l, respectively, at the same time. Then, according to the interference constraint:

$$\begin{aligned} |X_j - X_k| &\geq (1 + \Delta)|X_i - X_k] \\ |X_i - X_l| &\geq (1 + \Delta)|X_j - X_l]. \end{aligned}$$

Hence,

$$\begin{aligned} |X_j - X_i| &\geq |X_j - X_k| - |X_i - X_k| \\ &\geq \Delta |X_i - X_k|. \end{aligned}$$

Similarly,

$$|X_i - X_j| \geq \Delta |X_j - X_l|.$$

Therefore,

$$|X_i - X_j| \geq \frac{\Delta}{2} (|X_i - X_k| + |X_j - X_l|).$$

That is, disks of radius $\frac{\Delta}{2}$ times the transmission range centered at the transmitter are disjoint from each other[6]. This property can be generalized to *broadcast* as well. We only need to define the transmission range of a broadcast as the distance from the transmitter to the furthest node that can successfully receive the bit. The above property motivates us to measure the radio resources each transmission consumes by the areas of these disjoint disks [71]. For unicast transmissions from the last mobile relay to the destination, the area consumed by each hop is $\frac{\pi\Delta^2}{4}(S_b^h)^2$. For replication to other nodes, broadcast is more beneficial since it consumes fewer resources. Assume that each transmitter chooses the transmission range of the broadcast independently of the positions of its neighboring nodes. If the transmission range is s, then on average no greater than $n\pi s^2$ nodes can receive the broadcast, and a disk of radius $\frac{\Delta}{2}s$ (i.e., area $\frac{\pi\Delta^2}{4}s^2$) centered at the transmitter will be disjoint from other disks. Therefore, we can use $\frac{\Delta^2}{4}\frac{\mathbf{E}[R_b]-1}{n}$ as a lower bound on the expected area consumed by replicating the bit to R_b-1 mobile relays (excluding the source node). This lower bound will hold even if the replication process is carried out over multiple time slots, because the average number of *new* mobile relays each broadcast can cover is at most proportional to the area consumed by the broadcast. Therefore, inspired by [71], the amount of radio resources consumed must satisfy

$$\sum_{b=1}^{\lambda nT}\frac{\Delta^2}{4}\frac{\mathbf{E}[R_b]-1}{n}+\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}\frac{\pi\Delta^2}{4}(S_b^h)^2]\leq c_3'WT, \tag{5.10}$$

where c_3' is a positive constant.

However, $\frac{\Delta^2}{4}\frac{\mathbf{E}[R_b]-1}{n}$ may fail to be a lower bound on the expected area consumed by replicating to R_b-1 mobile relays if the following *opportunistic broadcast scheme* is used. The source may choose to broadcast *only when there are a larger number of nodes close by.* If the source can afford to wait for these "good opportunities", an *opportunistic broadcast scheme* may consume less radio resources than a non-opportunistic scheme to replicate the bit to the same number of mobile relays.

[6]A similar observation is used in [71] except that they take a receiver point of view.

Nonetheless, Proposition 5.4.3 shows that no scheduling policies can improve the tradeoff by more than a $\log n$ factor. For details, please refer to Appendix D.2.

5.4.5 Tradeoff IV : Half Duplex

Finally, since we assume that no node can transmit and receive over the same frequency at the same time (a practically necessary assumption for most wireless devices), the following property can be shown as in [71].

Proposition 5.4.4 *The following inequality holds,*

$$\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b} 1 \leq \frac{WT}{2}n. \tag{5.11}$$

5.5 The Upper Bound on the Capacity-Delay Tradeoff

Our first main result is to derive, from the above four tradeoffs, the upper bound on the optimal capacity-delay tradeoff of mobile wireless networks under the *i.i.d.* mobility model. Since the maximal achievable per-node capacity is $\Theta(1)$ and this capacity can be achieved with $\Theta(n)$ delay by the scheme of [75], we are only interested in the case when the mean delay is $o(n)$.

Proposition 5.5.1 *Let $\bar{D}$ be the mean delay averaged over all bits and all source-destination pairs, and let λ be the throughput of each source-destination pair. If $\bar{D} = O(n^d), 0 \leq d < 1$, the following upper bound holds for any causal scheduling policy under the* i.i.d. *mobility model,*

$$\lambda^3 \leq O(\frac{\bar{D}}{n}\log^3 n).$$

Proof Our goal is to reduce the four inequalities (5.6), (5.8), (5.9) and (5.11) to one inequality and eliminate all variables except $\bar{D}$ and λ. Using the Cauchy-Schwartz inequality, we have

$$\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b} S_b^h\right)^2 \leq \left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b} 1\right)\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2\right)$$

$$\leq \frac{WTn}{2}\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2, \tag{5.12}$$

where in the last step we have used Tradeoff IV (5.11). Equality holds in (5.12) when inequality (5.11) is tight and when S_b^h is equal for all b and h. We thus have,

$$\begin{aligned}
\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2] &\geq \frac{2}{WTn}\mathbf{E}[\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h\right)^2] \\
&\geq \frac{2}{WTn}\left(\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h]\right)^2 \qquad (5.13)\\
&\geq \frac{2}{WTn}\left(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2, \qquad (5.14)
\end{aligned}$$

where in the last two steps we have used Jensen's Inequality and the Tradeoff II (5.8), respectively. Inequality (5.13) is tight when $\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h$ is almost surely a constant, and (5.14) is tight when (5.8) is tight.

From Tradeoff I (5.6), we have

$$\sum_{b=1}^{\lambda nT}\mathbf{E}[R_b] \geq \sum_{b=1}^{\lambda nT}\frac{1}{c_1\log n}\frac{1}{(\mathbf{E}[l_b]+\frac{1}{n^2})^2\mathbf{E}[D_b]}. \tag{5.15}$$

Let

$$\bar{D} = \frac{\sum_{b=1}^{\lambda nT}\mathbf{E}[D_b]}{\sum_{b=1}^{\lambda nT}1} = \frac{\sum_{b=1}^{\lambda nT}\mathbf{E}[D_b]}{\lambda nT}.$$

Using Jensen's Inequality and Hőlder's Inequality, we have,

$$\frac{1}{\left(\frac{\sum_{b=1}^{\lambda nT}(\mathbf{E}[l_b]+\frac{1}{n^2})}{\sum_{b=1}^{\lambda nT}1}\right)^2} \leq \left(\frac{\sum_{b=1}^{\lambda nT}\frac{1}{(\mathbf{E}[l_b]+\frac{1}{n^2})}}{\sum_{b=1}^{\lambda nT}1}\right)^2 \leq \frac{\sum_{b=1}^{\lambda nT}\frac{1}{(\mathbf{E}[l_b]+\frac{1}{n^2})^2\mathbf{E}[D_b]}}{\sum_{b=1}^{\lambda nT}1}\frac{\sum_{b=1}^{\lambda nT}\mathbf{E}[D_b]}{\sum_{b=1}^{\lambda nT}1}. \tag{5.16}$$

Equality holds when $\mathbf{E}[l_b]$ is the same for all b and $\mathbf{E}[D_b] = \bar{D}$ for all b. Substituting (5.16) in (5.15), we have

$$\sum_{b=1}^{\lambda nT}\mathbf{E}[R_b] \geq \frac{1}{c_1\log n}\frac{\left(\sum_{b=1}^{\lambda nT}1\right)^3}{\bar{D}\left(\sum_{b=1}^{\lambda nT}(\mathbf{E}[l_b]+\frac{1}{n^2})\right)^2}. \tag{5.17}$$

Substituting (5.14) and (5.17) into Tradeoff III (5.9), we have

$$\begin{aligned}\frac{4c_3WT\log n}{\Delta^2} &\geq \sum_{b=1}^{\lambda nT}\frac{\mathbf{E}[R_b]-1}{n}+\pi\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2] \\ &\geq \frac{1}{c_1 n\log n}\frac{(\lambda nT)^3}{\bar{D}\left(\sum_{b=1}^{\lambda nT}(\mathbf{E}[l_b]+\frac{1}{n^2})\right)^2} \\ &\quad +\frac{2\pi}{WTn}\left(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2-\lambda T.\end{aligned}$$

To obtain a relationship between λ and $\bar{D}$, it remains to eliminate l_b. There are two cases that we need to consider.

Case 1: If $\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\leq\frac{\lambda T}{n}$, then

$$\begin{aligned}\frac{4c_3WT\log n}{\Delta^2} &\geq \frac{1}{c_1 n\log n}\frac{(\lambda nT)^3}{\bar{D}\left(\frac{2\lambda T}{n}\right)^2}-\lambda T \\ &= \frac{1}{4c_1\log n}\frac{\lambda Tn^4}{\bar{D}}-\lambda T.\end{aligned}$$

When $\bar{D}=O(n^d), d<1$, the first term dominates when n is large. Hence, for n large enough,

$$\begin{aligned}\frac{4c_3WT\log n}{\Delta^2} &\geq \frac{1}{8c_1\log n}\frac{\lambda Tn^4}{\bar{D}} \\ \lambda &\leq \frac{32c_1c_3W}{\Delta^2}\frac{\bar{D}\log^2 n}{n^4}. \qquad (5.18)\end{aligned}$$

Case 2: If $\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\geq\frac{\lambda T}{n}$, then

$$\frac{4c_3WT\log n}{\Delta^2} \geq \frac{1}{c_1 n\log n}\frac{(\lambda nT)^3}{\bar{D}\left(2\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2}+\frac{2\pi}{WTn}\left(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2-\lambda T \qquad (5.19)$$

$$\geq 2\sqrt{\frac{1}{c_1\log n}\frac{2\pi}{WTn^2}\frac{(\lambda nT)^3}{4\bar{D}}}-\lambda T \qquad (5.20)$$

$$= 2\sqrt{\frac{\pi}{2c_1\log n}\frac{\lambda^3 nT^2}{\bar{D}W}}-\lambda T. \qquad (5.21)$$

Therefore, either

$$\lambda \leq O(\frac{\bar{D}\log n}{n}), \tag{5.22}$$

or, if $\lambda = \omega(\frac{\bar{D}\log n}{n})$, then the first term in (5.21) dominates when n is large. In the latter case, for n large enough,

$$\begin{aligned} \frac{4c_3WT\log n}{\Delta^2} &\geq \sqrt{\frac{\pi}{2c_1\log n}\frac{\lambda^3 nT^2}{\bar{D}W}} \\ \lambda^3 &\leq \frac{32c_1c_3^2W^3}{\pi\Delta^4}\frac{\bar{D}\log^3 n}{n}. \end{aligned} \tag{5.23}$$

Finally, we compare the three inequalities we have obtained, i.e., (5.18), (5.22) and (5.23). Since $\bar{D} = o(n^d), d < 1$, inequality (5.23) will eventually be the loosest for large n. Hence, the optimal capacity-delay tradeoff is upper bounded by

$$\lambda^3 \leq O(\frac{\bar{D}}{n}\log^3 n).$$

■

5.6 An Achievable Lower Bound on the Capacity-Delay Tradeoff

The capacity-delay tradeoff in Proposition 5.5.1 is better than those reported in [75] and [76]. Assuming that the delay bound is $\Theta(n^d)$, $0 \leq d < 1$, the achievable per-node capacity is $O(n^{-(1-d)})$ by the scheme in [75], and $O(n^{-(1-d)/2})$ by the scheme in [76]. Our upper bound, however, implies a per-node capacity of $O(n^{-(1-d)/3})$ (we have ignored all $\log n$ factors). Since $d < 1$, there is clearly room to substantially improve existing schemes (see Fig. 5.1).

In this section, we will show how the study of the upper bound also helps us to develop a new scheme that can achieve a capacity-delay tradeoff that is close to the upper bound. Precisely, we met several inequalities (5.12)-(5.20) during the derivation of the upper bound. By studying the conditions under which these inequalities are tight, we will be able to identify the optimal choices of various key parameters of the scheduling policy. In the end, the knowledge of the optimal choices of the parameters will help us develop a new scheme that is superior to existing ones.

5.6.1 Choosing the Optimal Values of the Key Parameters

Assume that the mean delay is bounded by $n^d, d < 1$. By Proposition 5.5.1, we have,

$$\lambda \leq \Theta(\sqrt[3]{\frac{\bar{D}}{n}\log^3 n}) = \Theta(n^{\frac{d-1}{3}}\log n). \tag{5.24}$$

In order to achieve the maximum capacity on the right hand side, all inequalities (5.12)-(5.20) should hold with equality. By checking the conditions when (5.12)-(5.16) are tight, we can infer that the parameters (such as $S_b^h, \mathbf{E}[l_b], \mathbf{E}[D_b]$) of each bit b should be about the same and should concentrate on their respective average values. This implies that the scheduling policy should use the same parameters for all bits. From now on, we will assume that all key parameters (such as R_b, l_b, etc.) are indeed the same for all bits.

The inequality (5.20) is essential for deriving the optimal values of these parameters. Note that equality holds in (5.20) if and only if

$$\frac{1}{4c_1 n\log n}\frac{(\lambda nT)^3}{\bar{D}(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2} = \frac{2\pi}{WTn}(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2.$$

Substituting $\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b] = \lambda nTl_b$, we can solve for l_b,

$$\begin{aligned}\frac{1}{4c_1 n\log n}\frac{\lambda nT}{\bar{D}l_b^2} &= \frac{2\pi}{WTn}(\lambda nT)^2 l_b^2 \\ l_b^4 &= \frac{1}{8\pi c_1}\frac{W}{\bar{D}\lambda n\log n}.\end{aligned}$$

Substituting $\lambda = \Theta(n^{(d-1)/3}\log n)$ and $\bar{D} = n^d$, we obtain the optimal value of l_b,

$$l_b = \Theta(n^{-\frac{1+2d}{6}}\log^{-\frac{1}{2}} n).$$

A reasonable choice for the area of capture neighborhood, A_b, is then,

$$A_b = l_b^2 = \Theta(n^{-\frac{1+2d}{3}}/\log n).$$

By setting (5.11) of Tradeoff IV to equality, we have

$$\begin{aligned}\lambda nTh_b &= \frac{WTn}{2} \\ h_b &= \frac{W}{2\lambda} = \Theta(n^{\frac{1-d}{3}}/\log n).\end{aligned}$$

Table 5.1
The order of the optimal values of the parameters when the mean delay is bounded by n^d.

R_b: # of Duplicates	$\Theta(n^{(1-d)/3})$
l_b: Distance to Destination	$\Theta(n^{-(1+2d)/6}/\log^{1/2} n)$
h_b: # of Hops	$\Theta(n^{(1-d)/3}/\log n)$
S_b^h: Transmission Range of Each Hop	$\Theta(\sqrt{\frac{\log n}{n}})$

By setting (5.8) of Tradeoff II to equality, we have

$$S_b^h = \frac{l_b}{h_b} = \Theta(\sqrt{\frac{\log n}{n}}).$$

Finally, by setting (5.6) of Tradeoff I to equality, we have

$$R_b \;\; = \;\; \Theta(\frac{1}{c_1 \log n}\frac{1}{l_b^2 \bar{D}}) = \Theta(n^{\frac{1-d}{3}}).$$

The optimal values of these parameters are summarized in Table 5.1.

Several remarks are in order. Since it is sufficient to control all parameters around these optimal values, simple cell-based schemes such as the one in Example B of Section 5.4 suffice. Secondly, the optimal values for R_b and l_b can provide guidelines on how to choose the cell partitioning. Thirdly, the optimal value for S_b^h is roughly the average distance between neighboring nodes when n nodes are uniformly distributed in a unit square. Hence, it is desirable to use multi-hop transmission over neighboring nodes to forward the information from the last mobile relay to the destination. These guidelines have sketched a blueprint of the optimal scheduling scheme for us. We next present schemes that can achieve capacity-delay tradeoffs that are close to the upper bound up to a logarithmic factor.

5.6.2 Achievable Capacity with $\Theta(n^d)$ Delay

We now present a scheme that can achieve the optimal capacity-delay tradeoff in (5.24) up to a logarithmic factor. Our scheme can achieve $\Theta(n^{\frac{d-1}{3}}/\log^{3/2} n)$ per-node capacity with $\Theta(n^d)$ delay, $0 \leq d < 1$. When $d = 0$, i.e., when the delay is bounded by a constant, our scheme can achieve $\Theta(n^{-1/3}/\log^{3/2} n)$ per-node capacity with $\Theta(1)$ delay[7]. This is an encouraging result for mobile networks because we know that the per-node capacity of static networks is $O(1/\sqrt{n \log n})$ [71]. Hence, mobility increases the capacity even with constant delay. This is the first such result of its kind in the literature.

We will need the following Lemma before stating the main scheduling scheme. We will repeatedly use the following type of cell-partitioning. Let m be a positive integer. Divide the unit square into $m \times m$ cells (in m rows and m columns, see Fig. 5.2). Each cell is a square of area $1/m^2$. As in [76], we call two cells *neighbors* if they share a common boundary, and we call two nodes *neighbors* if they lie in the same or neighboring cells. We say that a group of cells can be *active* at the same time when one node in each cell can successfully transmit to or receive from a neighboring node, subject to the interference from other cells that are active at the same time. Let $\lfloor x \rfloor$ be the largest integer smaller than or equal to x. The proof of the following Lemma is available in Appendix D.3.

Lemma 5.6.1 *There exists a scheduling policy such that each cell can be active for at least $1/c_4$ amount of time, where c_4 is a constant independent of m.*

Group every $\lfloor n^d \rfloor$ time slots into a *super-frame*. The capacity achieving scheme is as follows.

Capacity Achieving Scheme:

[7]The scheme for $d = 0$ can be further refined to achieve $\Theta(n^{-1/3}/\log n)$ per-node capacity with $\Theta(1)$ delay [24].

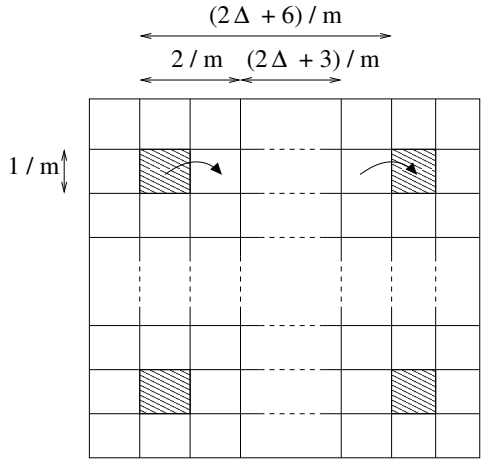


Fig. 5.2. Cells that are $\lfloor 2\Delta + 6 \rfloor / m$ apart (i.e., the shaded cells in the figure) can be active together.

1) In every *odd* super-frame (i.e., the 1st, 3rd, 5th, ... super-frame), we schedule transmissions from the sources to the relays. We will refer to the $\lfloor n^d \rfloor$ time slots in each odd super-frame as the *sending* time slots. At each sending time slot:

- We divide the unit square into $g_1(n) = \lfloor \left(\frac{n^{\frac{2+d}{3}}}{16 \log n} \right)^{\frac{1}{2}} \rfloor^2$ cells. Each cell is a square of area $1/g_1(n)$. We refer to each cell in the sending time slot as a *sending cell*. By Lemma 5.6.1, each cell can be active for $\frac{1}{c_4}$ amount of time in each sending time slot.

- When a cell is scheduled to be active, each node in the cell broadcasts a new packet to all other nodes in the same cell for $\frac{1}{4096 c_4 n^{(1-d)/3} \log^{3/2} n}$ amount of time (Fig. 5.3). The length of the packet is $\frac{W}{4096 c_4 n^{(1-d)/3} \log^{3/2} n}$. These other nodes then serve as mobile relays for the packet. The nodes within the same sending cell coordinate themselves to broadcast sequentially in each sending time slot.

- If, in any of the $\lfloor n^d \rfloor$ sending time slots, there exists a sending cell that has more than $64 n^{(1-d)/3} \log n$ nodes, we refer to it as a Type-I error [76]. If no Type-I errors occur, each source can broadcast a total of $\lfloor n^d \rfloor$ distinct packets, each of length $\frac{W}{4096 c_4 n^{(1-d)/3} \log^{3/2} n}$, during the entire odd super-frame.

2) In every *even* super-frame (i.e., the 2nd, 4th, 6th, ... super-frame), we schedule transmissions from the mobile relays to the destination nodes. We will refer to the $\lfloor n^d \rfloor$ time slots in each even super-frame as the *receiving* time slots. At each receiving time slot:

- We divide the unit square into $g_2(n) = \lfloor \left(n^{\frac{1+2d}{3}} \right)^{\frac{1}{2}} \rfloor^2$ cells. Each cell is a square of area $1/g_2(n)$. We refer to each cell in the receiving time slot as a *receiving cell*.

- Note that there are $n \lfloor n^d \rfloor$ distinct packets that are injected into the system at the previous odd super-frame. *Capture* occurs for each packet k when one of its mobile relays moves within the same receiving cell as the destination node (see

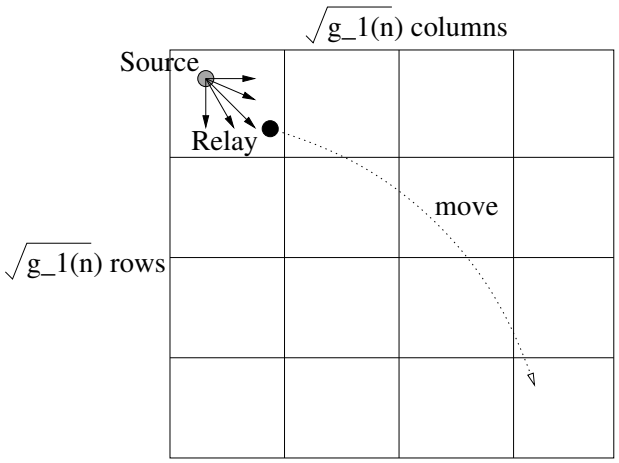


Fig. 5.3. Transmission schedule in each time slot of the odd superframe. $g_1(n) = \lfloor \left(\frac{n^{\frac{2+d}{3}}}{16 \log n} \right)^{\frac{1}{2}} \rfloor^2$.

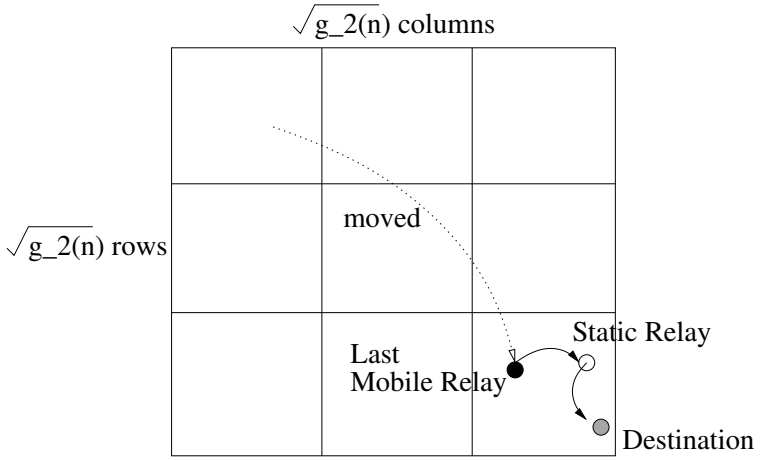


Fig. 5.4. Transmission schedule in each time slot of the even super-frame. $g_2(n) = \lfloor \left(n^{\frac{1+2d}{3}} \right)^{\frac{1}{2}} \rfloor^2$.

Fig. 5.4). Let $\mathcal{Y}_j(t)$ denote the set of packets that meet the criteria for *capture* in receiving cell j at receiving time slot t. We will refer to these packets as the *active packets*. Among the $n\lfloor n^d \rfloor$ distinct packets, if there exists a packet k that does not meet the criteria for *capture* in any of the $\lfloor n^d \rfloor$ receiving time slots, i.e., packet k does not belong to $\mathcal{Y}_j(t)$ for any $j = 1, ..., g_2(n)$ and $t = 1, ..., \lfloor n^d \rfloor$, we will refer to it as a Type-II error. Unless a Type-II error occurs, each of these $n\lfloor n^d \rfloor$ distinct packets will have at least one opportunity to be carried into the same receiving cell as its destination node (see Fig. 5.4).

- The active packets in each set $\mathcal{Y}_j(t)$ are then forwarded to their destination nodes (residing in the same receiving cell j) within the same time slot t in the following multi-hop fashion. We further divide the receiving cell j into $g_3(n) = \lfloor \left(\frac{n^{\frac{2-2d}{3}}}{8 \log n} \right)^{\frac{1}{2}} \rfloor^2$ *mini-cells* (in $\sqrt{g_3(n)}$ rows and $\sqrt{g_3(n)}$ columns, see Fig. 5.5). Each mini-cell is a square of area $1/(g_2(n)g_3(n))$. By Lemma 5.6.1, there exists a scheduling scheme where each mini-cell can be active for $\frac{1}{c_4}$ amount of time in time slot t. When each mini-cell is active, it forwards an active packet (or a part of the packet) to one other node in the neighboring

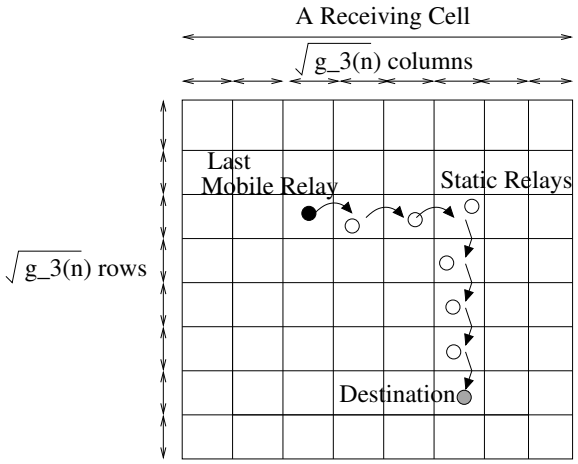


Fig. 5.5. Multi-hop transmissions within a receiving cell.

mini-cell. If the destination of the active packet is in the neighboring mini-cell, the packet is forwarded directly to the destination node. The active packets from each mobile relay are first forwarded towards neighboring mini-cells along the X-axis, then to their destination nodes along the Y-axis (see Fig. 5.5). In this fashion, a successful schedule will be able to deliver all active packets in $\mathcal{Y}_j(t)$ to their respective destination nodes (in the same receiving cell j) within the time slot t. For details on constructing such a schedule, see Appendix D.4. If no such schedule exists, we refer to it as a Type-III error.

- At the end of each even super-frame, any packets that remain in the buffer of the mobile relays (i.e., that have not been delivered to the destination nodes) are dropped.

It is possible that the above scheme delivers the same packet more than once to the same destination node. However, our analysis shows that such replication does not change the order of the asymptotic achievable capacity. We assume that each packet has a sequence number so that the destination node can detect replicate packets and only keep the new packets. We can show that, as $n \to \infty$, the probabilities of errors of Type-I, II, and III, will all go to zero. The following proposition

then holds, which shows that our scheme can achieve $\Theta(n^{(d-1)/3}/\log^{3/2} n)$ per-node capacity with $\Theta(n^d)$ delay. The proof is available in Appendix D.4.

Proposition 5.6.1 *Assume the* i.i.d. *mobility model. With probability approaching one, as $n \to \infty$, the above scheme allows each source to send $\lfloor n^d \rfloor$ packets of length $\frac{W}{4096 c_4 n^{(1-d)/3} \log^{3/2} n}$ to its respective destination node within $2\lfloor n^d \rfloor$ time slots.*

Remark: Our schemes belong to the class of cell-based schemes (see Example B in Section 5.3). However, our scheme uses different cell-partitioning in the odd super-frames than that in the even super-frames. Note that in previous works [75, 76], the cell structure remains the same over all time slots. Our judicious choice of the cell-structures is the key to the derivation of a tighter lower bound for the capacity. In particular, the size of the sending cell is chosen such that the average number of nodes in each cell, $n/g_1(n) = \Theta(n^{(1-d)/3} \log n)$, is close to the optimal value of R_b in Section 5.6.1. The size of the receiving cell is chosen such that its area, $1/g_2(n) = \Theta(n^{-(1+2d)/3})$, is close to the optimal value of l_b^2. Finally, the size of the mini-cell is chosen such that each hop to the neighboring cell is of length $1/\sqrt{g_2(n)g_3(n)} = \Theta(\sqrt{\log n/n})$, which is close to the optimal value of S_b^h.

5.6.3 The Effect of Queueing

When we defined the delay D_b of each bit b in Section 5.4, it included possible queueing delays at the source node and at the relay nodes. The upper bound on the capacity-delay tradeoff (Proposition 5.5.1) thus holds regardless of the queueing discipline used in the system, and $\bar{D}$ also includes the queueing delay. We now show how to analyze the queueing delay of the capacity-achieving scheme in Section 5.6.2. We first consider the case when $d = 0$. For $d = 0$, our scheme will attempt to deliver one packet of length $\frac{W}{4096 c_4 n^{1/3} \log^{3/2} n}$ for each source-destination pair every two time slots. Let p_s be the probability that a packet is successfully delivered to the destination at the end of the even time slot. (Note that p_s is the same for all source-destination pairs due to symmetry, and by Proposition 5.6.1, $p_s \to 1$ as

$n \to \infty$.) Assume that, if such delivery is unsuccessful, packets that have not been delivered to the destinations at the end of each even time slot are removed from the buffers of the relay nodes and have to be retransmitted by their source nodes at the next time slot. Further, assume that packets of length $\frac{W}{4096c_4 n^{1/3}\log^{3/2} n}$ arrive at each source according to a certain stochastic process. Then packets may get enqueued at the source nodes. If we observe the system at the end of each even time slot, the number of packets queued for each source-destination pair will evolve as that of a single-server discrete-time queue with geometric service time distributions [89], and the queues for each source-destination pair can be studied independently. If we know the packet arrival process, we can then compute the queueing delay. For example, fix a source-destination pair and assume that the distribution of the number of packets (denoted by A) that arrive at the source node every two time slots is *i.i.d.* with $\mathbf{P}[A = i] = e_i, i = 0, 1, ..., \infty$. Let Λ denote the mean packet arrival rate in every two time slots, i.e., $\Lambda = \mathbf{E}[A]$. Let $\hat{A}(z)$ denote the moment generating function of A, i.e., $\hat{A}(z) = \sum_{i=0}^{\infty} e_i z^i, |z| \leq 1$. Using results for Geom$^{[X]}$/Geom/1 discrete-time queues [89, p89], we can compute the average number of packets queued at the source node as:

$$\mathbf{E}[Q] = \hat{A}'(1) + \frac{\hat{A}''(1) + 2\hat{A}'(1)(1-p_{\mathrm{s}})}{2(p_{\mathrm{s}} - \hat{A}'(1))}.$$

Using Little's Law and the fact that

$$\hat{A}'(1) = \mathbf{E}[A] = \Lambda, \text{ and } \hat{A}''(1) = \mathrm{Var}[A] + \mathbf{E}^2[A] - \mathbf{E}[A] = \mathrm{Var}[A] + \Lambda^2 - \Lambda,$$

we can compute the mean queueing delay (in number of time slots) as

$$\mathbf{E}[D] = 2\frac{\mathbf{E}[Q]}{\Lambda} = 2\left(1 + \frac{\mathrm{Var}[A]}{2\Lambda(p_{\mathrm{s}} - \Lambda)} + \frac{\Lambda - 1}{2(p_{\mathrm{s}} - \Lambda)} + \frac{1 - p_{\mathrm{s}}}{p_{\mathrm{s}} - \Lambda}\right).$$

Because $p_{\mathrm{s}} \to 1$ as $n \to \infty$, we have,

$$\mathbf{E}[D] \to 1 + \frac{\mathrm{Var}[A]}{\Lambda(1-\Lambda)}, \text{ as } n \to \infty.$$

In particular, if the arrival process is Bernoulli, i.e., one new packet arrives at the source node every two time slots with probability Λ, then

$$\mathrm{Var}[A] = \Lambda(1 - \Lambda).$$

Hence,

$$\mathbf{E}[D] \to 2, \text{ as } n \to \infty.$$

On the other hand, if the arrival process is Poisson with rate $\Lambda/2$, then the number of packets that arrive at the source node every two time slots is a Poisson random variable with mean Λ. Hence,

$$\text{Var}[A] = \Lambda,$$

and

$$\mathbf{E}[D] \to 1 + \frac{1}{1-\Lambda} = \frac{2-\Lambda}{1-\Lambda}, \text{ as } n \to \infty.$$

Note that in both cases, the mean queueing delay is at most a constant multiple of 2 (time slots) provided that $1 - \Lambda$ (i.e., the difference between the arrival rate and the capacity) is positive and bounded away from zero as $n \to \infty$. Hence, the capacity-achieving scheme in Section 5.6.2 can sustain $\Theta(n^{-1/3}/\log^{3/2} n)$ per-node throughput (in bits per time slot) with $O(1)$ *queueing delay*.

The above analysis can be generalized to $d > 0$ as well. Let $N = \lfloor n^d \rfloor$. Our scheme in Section 5.6.2 will attempt to deliver N packets of length $\frac{W}{4096c_4 n^{(1-d)/3} \log^{3/2} n}$ for each source-destination pair every two super-frames. Let p_{s} be the probability that all Nn packets are successfully delivered to the destination at the end of the even super-frame. Because $p_{\text{s}} \to 1$ as $n \to \infty$, we will simply use $p_{\text{s}} = 1$ in the following derivation. Assume that packets of length $\frac{W}{4096c_4 n^{(1-d)/3} \log^{3/2} n}$ arrive at each source according to certain stochastic process. If we observe the system at the end of each even super-frame, the number of packets queued for each source-destination pair will evolve as that of a discrete-time queue with N servers and deterministic service times (equal to the length of two super-frames), and the queues for each source-destination pair can again be studied independently. Fix a source-destination pair and assume that the distribution of the number of packets (denoted by A) that arrive at the source node every two super-frames is *i.i.d.* with $\mathbf{P}[A = i] = e_i, i = 0, 1, ..., \infty$. We can then study the mean queueing delay using results from discrete-time $\text{Geom}^{[X]}/D/N$ queues [90, 91]. The exact analysis is usually quite tedious [90].

However, simple upper bounds are available in [90, 91]. Let $N\Lambda$ denote the mean packet arrival rate in every two super-frames, i.e., $N\Lambda = \mathbf{E}[A]$. Let $\hat{A}(z)$ denote the moment generating function of A. Using results in [90, Lemma 1], we can compute the upper bound on the average number of packets queued at the source node as:

$$\mathbf{E}[Q] \leq Q_L + \frac{N}{2},$$

where

$$Q_L = N\Lambda + \frac{\hat{A}''(1) - N\Lambda(N-1)}{2N(1-\Lambda)}.$$

Using Little's Law and the fact that,

$$\hat{A}'(1) = \mathbf{E}[A] = N\Lambda, \text{ and } \hat{A}''(1) = \mathrm{Var}[A] + \mathbf{E}^2[A] - \mathbf{E}[A] = \mathrm{Var}[A] + N^2\Lambda^2 - N\Lambda,$$

we can compute the upper bound on the mean queueing delay (in number of super-frames) as[8]

$$\mathbf{E}[D] \leq 2\frac{\mathbf{E}[Q]}{N\Lambda} \leq 1 + \frac{1}{\Lambda} + \frac{\mathrm{Var}[A]}{N^2\Lambda(1-\Lambda)}.$$

If Λ is bounded away from 1 and $\mathrm{Var}[A] = O(N^2)$, then the mean queueing delay is at most a constant multiple of 2 (super-frames). Since each super-frame consists of $\lfloor n^d \rfloor$ time slots, we have thus shown that the capacity-achieving scheme in Section 5.6.2 can sustain $\Theta(n^{(d-1)/3}/\log^{3/2} n)$ per-node throughput (in bits per time slot) with $O(n^d)$ *queueing delay* (in number of time slots).

5.6.4 Simulation Results

We have simulated the capacity achieving scheme in Section 5.6.2 for different values of the delay exponent d and the number of nodes n. In our simulation, we use Bernoulli packet arrival processes, i.e., one packet arrives at each source-destination pair every time-slot with probability p. Hence, $\Lambda = 2p$. The packet size scales as $\frac{W}{4096c_4 n^{(1-d)/3}\log^{3/2} n}$. For a fixed p, the offered load for each source-destination pair

[8]This upper bound may explode when $\Lambda \to 0$. However, since we are interested in the *achievable* capacity, we only need to deal with the case when Λ is bounded away from zero. Tighter upper bounds for Λ close to zero are given in [91].

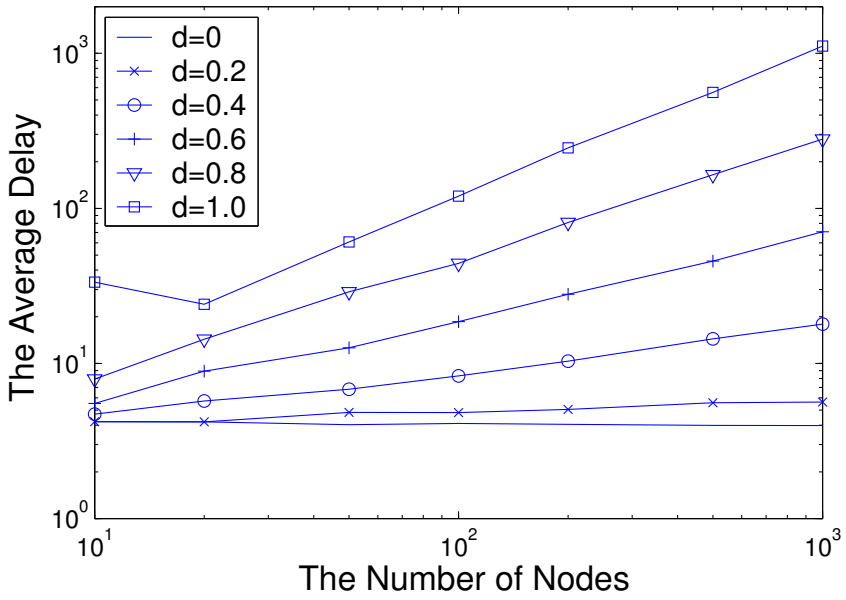


Fig. 5.6. Mean queueing delay (in number of time slots) versus the number of nodes.

then scales as $\Theta(n^{(d-1)/3}/\log^{3/2} n)$ (in bits per time slot). We use $p = 0.4$ in the simulation. Fig. 5.6 shows the mean queueing delay (in number of time slots) versus the number of nodes n at different values of d. We observe that, for large n, the mean queueing delay evolves as $\Theta(n^d)$ for all values of d. This figure demonstrates that our scheme can indeed sustain $\Theta(n^{(d-1)/3}/\log^{3/2} n)$ per-node throughput with $\Theta(n^d)$ delay.

5.7 The Limiting Factors in Existing Schemes

In Section 5.6, we have shown that choosing the optimal values of the scheduling parameters is the key to achieve the optimal capacity-delay tradeoff. In this section, we will show that deviating from these optimal values will lead to suboptimal capacity-delay tradeoffs. In particular, we will identify the limiting factors in the existing schemes in [75] and [76] by comparing the optimal values of scheduling parameters in Section 5.6.1 with those used by the existing schemes. Our model

in Section 5.5 can be extended to study the upper bounds on the capacity-delay tradeoff when one imposes additional restrictive assumptions that correspond to these limiting factors. We will see that these new upper bounds are inferior to the capacity-delay tradeoff reported in Sections 5.5 and 5.6. The existing schemes of [75] and [76] in fact achieve capacity-delay tradeoffs that are close to the respective upper bounds. These results will give us new insights on which schemes to use under different conditions.

5.7.1 The Limiting Factor in the Scheme of Neely and Modiano

The scheme by Neely and Modiano [75] divides the unit square into n cells each of area $1/n$. A mobile relay will forward messages to the destination only when they both reside in the same cell. Hence, the distance from the last mobile relay to the destination, l_b, is on average on the order of $O(1/\sqrt{n})$, regardless of the delay constraints. However, we have shown in Section 5.6.1 that the optimal choice for l_b should be on the order of $\Theta(n^{-(1+2d)/6}\log^{-1/2} n)$, when the mean delay is bounded by $\Theta(n^d)$. The next Proposition shows that the restrictive choice of l_b is indeed the limiting factor of the scheme in [75]. The proof is available in Appendix D.5.

Proposition 5.7.1 *Let $\bar{D}$ be the mean delay averaged over all bits and all source-destination pairs, and let λ be the throughput of each source-destination pair. If $\bar{D} = O(n^d), 0 \leq d < 1$ and $\mathbf{E}[l_b] = O(1/\sqrt{n})$, then for any causal scheduling policy,*

$$\lambda \leq O(\frac{\bar{D}}{n}\log^2 n).$$

Remark: The scheme of [75] achieves the above upper bound up to a logarithmic factor.

5.7.2 The Limiting Factor in the Scheme of Toumpis and Goldsmith

In the scheme by Toumpis and Goldsmith [76], a mobile relay will always use single-hop transmission to forward the messages directly to the destination. That

is, the number of hops from the last mobile relay to the destination node, h_b, is always 1. However, we have shown in Section 5.6.1 that the optimal value of h_b is $\Theta(n^{(1-d)/3}/\log n)$ when the mean delay is bounded by $\Theta(n^d)$. The next Proposition shows that the restriction on h_b is indeed the limiting factor of the scheme in [76]. The proof is available in Appendix D.6.

Proposition 5.7.2 *Let $\bar{D}$ be the mean delay averaged over all bits and all source-destination pairs, and let λ be the throughput of each source-destination pair. If $\bar{D} = O(n^d), 0 \leq d < 1$ and $h_b = O(1)$, then for any causal scheduling policy,*

$$\lambda^2 \leq O(\frac{\bar{D}}{n} \log^3 n).$$

Remark: The scheme of [76] achieves the above upper bound up to a logarithmic factor.

Propositions 5.5.1, 5.7.1 and 5.7.2 present three different upper bounds on the capacity-delay tradeoff of mobile wireless networks under different assumptions. Assume that the mean delay is bounded by $n^d, 0 \leq d < 1$. When the capacity is the main concern, Proposition 5.5.1 shows that the per-node throughput is at most $O(n^{(d-1)/3} \log n)$. The capacity-achieving scheme reported in Section 5.6 can achieve close to this upper bound up to a logarithmic factor. However, this capacity-achieving scheme requires sophisticated coordination among the mobile nodes. Hence, it may not be suitable *when simplicity is the main concern.* On the other hand, the scheme of [75] only requires coordination among nodes that are within a cell of area $1/n$. Note that the average number of nodes in such a cell is $\Theta(1)$. Proposition 5.7.1 then shows that, when coordination among a large number of nodes is prohibited, the scheme of [75] is close to optimal. Similarly, the scheme of [76] only requires single-hop transmissions from the mobile relays to the destinations. Proposition 5.7.2 shows that, when multi-hop transmissions are undesirable, the scheme of [76] is close to optimal. Therefore, the results reported in this chapter present a relatively complete picture of the achievable capacity-delay tradeoffs under different conditions.

5.8 The Random Way-point Mobility Model

The analyses in the previous sections have focused on the *i.i.d.* mobility model. In this section, we will extend our methodology to the random way-point (*RWP*) mobility model. In previous sections, we have shown that, at least for the *i.i.d.* mobility model, there is not a significant loss of generality by using cell-based schemes. Indeed, the capacity-achieving scheme in Section 5.6.2 belong to the class of cell-based schemes (see Example B in Section 5.3). By choosing the appropriate cell-partitioning, we have been able to use cell-based schemes to asymptotically achieve the maximum capacity under given delay constraints. Note that the key parameters of the capacity-achieving scheme in Section 5.6.2 are the number of sending cells $g_1(n)$ and the number of receiving cells $g_2(n)$. Hence, in this section, we will restrict our attention to cell-based schemes only, and our focus will be to find the optimal cell-partitioning (i.e., the values of $g_1(n)$ and $g_2(n)$) for the *RWP* mobility model. However, in the *RWP* mobility model, the nodes move continuously instead of in time slots. Hence, we need to modify the cell-based scheme as follows. We still divide the time into slots of unit length. After a bit b enters the system, it is broadcast to all other nodes in the same sending cell by the end of the next time slot. Let R_b be the number of mobile relays that receive the bit b. After certain delay D_b, one of the mobile relays (i.e., the *last mobile relay*) moves into the same receiving cell (of area A_b) as the destination node of bit b. The bit b is then forwarded from the last mobile relay to the destination by the end of the next time slot. Since the velocity of the nodes is $v(n) = \Theta(1/\sqrt{n})$, the distance any node can move within one time slot is of order $\Theta(1/\sqrt{n})$, which is small compared to the sizes of the sending cells and the receiving cells that we will choose later. Hence, the mobility of the nodes will not interfere much with both the replication of the bit at the very beginning and the multi-hop forwarding after capture. Let h_b be the number of hops that bit b takes from the last mobile relay to the destination.

With the above modification of the cell-based scheme, the Tradeoffs II, III, and IV can be readily extended to the *RWP* mobility model. However, the exact counterpart to Tradeoff I is quite difficult to obtain analytically. We instead use numerical methods to study the likely form of Tradeoff I under the *RWP* mobility model. In the cell-based scheme, the number of mobile relays for each bit b is determined at the beginning of the replication process, and all of these mobile relays were close to the source node of bit b when they received bit b. Hence, we can use the following simple simulation model to study the tradeoff between D_b, R_b and A_b. At time $t = 0$, we put one (destination) node at a random position uniformly distributed within the unit square. We put R_b mobile relays at another random position. Let the size of the receiving cell be $A_b = 1/g_2(n)$. We then let these nodes move according to the *RWP* mobility model and record the mean delay $\mathbf{E}[D_b]$ (averaged over simulation runs) before any one of the R_b mobile relays moves within the same receiving cell as the destination node. Varying R_b and A_b, we can thus obtain the relationship between $\mathbf{E}[D_b]$, R_b and A_b. However, note that we are only interested in the relationship when n is large. In order to extract the most useful information, we let $R_b = n^{d_1}, 0 < d_1 < 1$, and $A_b = n^{-d_2}, 0 < d_2 < 1$. With any fixed d_1 and d_2, we observe from our simulations that, when n is large, $\frac{\log \mathbf{E}[D_b]}{\log n}$ will converge to a number d, i.e., the delay is approximately n^d. In Fig. 5.7, we plot the relationship between d, d_1, and d_2 for the *RWP* mobility model. It is instructive to compare with the same plot obtained under the *i.i.d.* mobility model (Fig. 5.8). Note that each line in Fig. 5.8 can be expressed as

$$d = d_2 - d_1, \text{ for } d \geq 0,$$

which is consistent with (5.6) noting that $l_b = \Theta(\sqrt{A_b})$. On the other hand, each line in Fig. 5.7 can be expressed as

$$d = \frac{1 + d_2}{2} - d_1, \text{ for } 0.5 < d < 1,$$

which corresponds to

$$\mathbf{E}[D_b] \approx \Theta(\frac{n^{\frac{1}{2}}}{\mathbf{E}[l_b]\mathbf{E}[R_b]}). \tag{5.25}$$

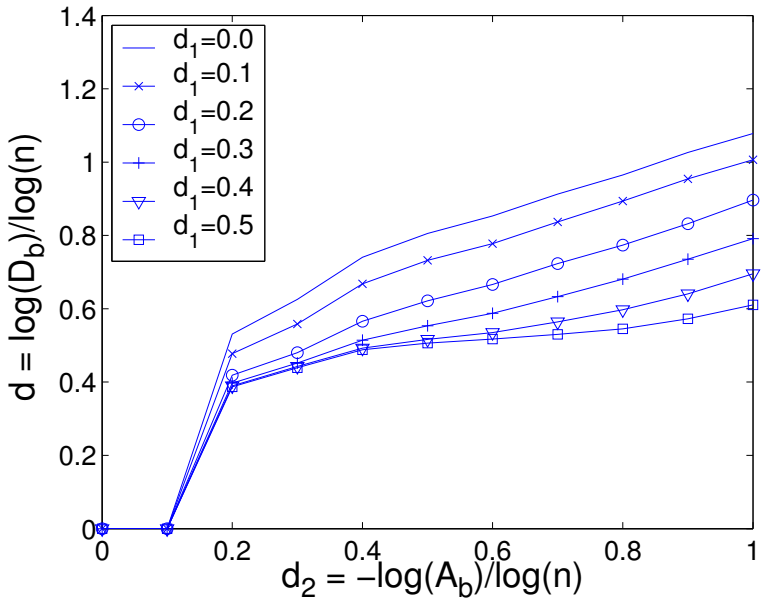


Fig. 5.7. Delay exponent d versus d_1 and d_2 for the random way-point mobility model.

This relationship between D_b, l_b and R_b under the *RWP* mobility model is consistent with the findings in [84]. When l_b and R_b are fixed, it has been shown in [84] that a given mobile relay can move within a distance l_b from the destination node during a single trip with probability $\Theta(l_b)$. Since odd trips are independent from each other, the expected number of trips for any of the R_b mobile relays to move within distance l_b from the destination node is $\Theta(\frac{1}{l_b R_b})$. Finally, as $v(n) = \Theta(1/\sqrt{n})$, each trip will take $\Theta(\sqrt{n})$ amount of time. We then obtain (5.25). However, this relationship only holds when $d \geq 0.5$. In fact, it has been shown in [84] that, in order to take advantage of mobility, the minimum amount of delay under the *RWP* mobility model is $\Theta(\sqrt{n})$.

5.8.1 Upper Bound on the Capacity-Delay Tradeoff

Combining relationship (5.25) with Tradeoffs II, III and IV, we can then compute the upper bound on the maximum capacity under given delay constraints as in

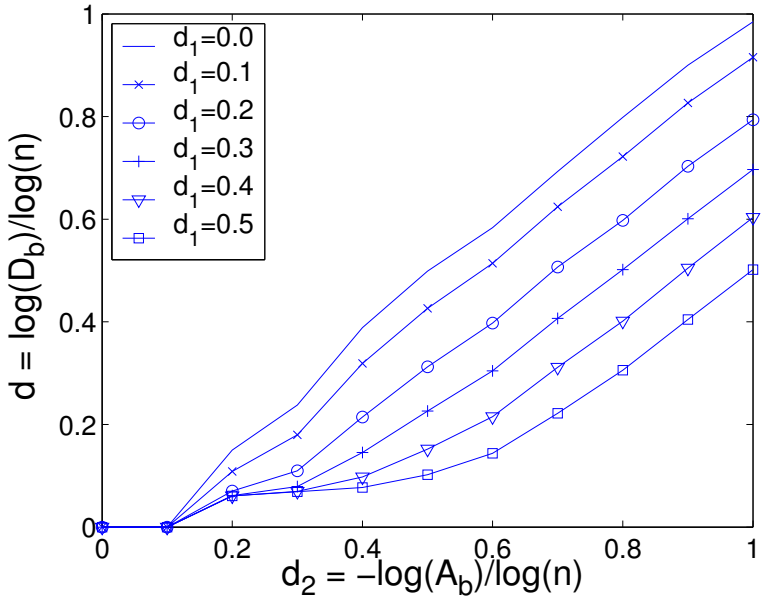


Fig. 5.8. Delay exponent d versus d_1 and d_2 for the *i.i.d.* mobility model.

Proposition 5.5.1. We can in fact assume a more general version of the estimate (5.25):

$$\mathbf{E}[D_b] \geq \frac{c_1 n^{\frac{1}{2}-\epsilon}}{\mathbf{E}[l_b]\mathbf{E}[R_b]}, \tag{5.26}$$

where $c_1 > 0$ and ϵ is a positive number close to 0. We then have

$$\sum_{b=1}^{\lambda nT} \mathbf{E}[R_b] \geq c_1 n^{\frac{1}{2}-\epsilon} \sum_{b=1}^{\lambda nT} \frac{1}{\mathbf{E}[l_b]\mathbf{E}[D_b]}. \tag{5.27}$$

Let

$$\bar{D} = \frac{\sum_{b=1}^{\lambda nT} \mathbf{E}[D_b]}{\sum_{b=1}^{\lambda nT} 1} = \frac{\sum_{b=1}^{\lambda nT} \mathbf{E}[D_b]}{\lambda nT}.$$

Using Jensen's Inequality and Hőlder's Inequality, we have,

$$\begin{aligned} \frac{1}{\left(\frac{\sum_{b=1}^{\lambda nT} \mathbf{E}[l_b]}{\sum_{b=1}^{\lambda nT} 1}\right)} &\leq \left(\frac{\sum_{b=1}^{\lambda nT} \frac{1}{\sqrt{\mathbf{E}[l_b]}}}{\sum_{b=1}^{\lambda nT} 1}\right)^2 \\ &\leq \frac{\sum_{b=1}^{\lambda nT} \frac{1}{\mathbf{E}[l_b]\mathbf{E}[D_b]}}{\sum_{b=1}^{\lambda nT} 1} \frac{\sum_{b=1}^{\lambda nT} \mathbf{E}[D_b]}{\sum_{b=1}^{\lambda nT} 1}. \end{aligned} \tag{5.28}$$

Equality holds when $\mathbf{E}[l_b]$ is the same for all b and $\mathbf{E}[D_b] = \bar{D}$ for all b. Substituting (5.28) in (5.27), we have

$$\sum_{b=1}^{\lambda nT} \mathbf{E}[R_b] \quad \geq \quad c_1 n^{\frac{1}{2}-\epsilon} \frac{\left(\sum_{b=1}^{\lambda nT} 1\right)^2}{\bar{D} \sum_{b=1}^{\lambda nT} \mathbf{E}[l_b]}. \tag{5.29}$$

Substituting (5.14) and (5.29) into Tradeoff III (5.9), we have

$$\begin{aligned} \frac{4c_3 WT \log n}{\Delta^2} &\geq \sum_{b=1}^{\lambda nT} \frac{\mathbf{E}[R_b] - 1}{n} + \pi \mathbf{E}[\sum_{b=1}^{\lambda nT} \sum_{h=1}^{h_b} (S_b^h)^2] \\ &\geq c_1 n^{-\frac{1}{2}-\epsilon} \frac{(\lambda nT)^2}{\bar{D} \sum_{b=1}^{\lambda nT} \mathbf{E}[l_b]} + \frac{2\pi}{WTn} \left(\sum_{b=1}^{\lambda nT} \mathbf{E}[l_b]\right)^2 - \lambda T. \end{aligned}$$

Using the inequality

$$a+b+c \geq 3\sqrt[3]{abc},$$

we have,

$$\begin{aligned}
\frac{4c_3WT\log n}{\Delta^2} &\geq 3\sqrt[3]{\left(\frac{c_1}{2}n^{-\frac{1}{2}-\epsilon}\frac{(\lambda nT)^2}{\bar{D}\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]}\right)^2\frac{2\pi}{WTn}\left(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2} - \lambda T \quad (5.30)\\
&= 3\sqrt[3]{\frac{c_1^2\pi}{2}\frac{\lambda^4 n^{2-2\epsilon}T^3}{\bar{D}^2W}} - \lambda T. \quad (5.31)
\end{aligned}$$

Therefore, either

$$\lambda \leq O(\frac{\bar{D}^2}{n^{2-2\epsilon}}), \quad (5.32)$$

or, if $\lambda = \omega(\frac{\bar{D}^2}{n^{2-2\epsilon}})$, then the first term in (5.31) dominates when n is large. In the latter case, for n large enough, we have,

$$\begin{aligned}
\frac{4c_3WT\log n}{\Delta^2} &\geq \frac{3}{2}\sqrt[3]{\frac{c_1^2\pi}{2}\frac{\lambda^4 n^{2-2\epsilon}T^3}{\bar{D}^2W}}\\
\frac{64c_3^3W^3T^3\log^3 n}{\Delta^6} &\geq \frac{27}{8}\frac{c_1^2\pi}{2}\frac{\lambda^4 n^{2-2\epsilon}T^3}{\bar{D}^2W}\\
\lambda^2 &\leq \sqrt{\frac{1024c_3^3W^4}{27\pi c_1^2\Delta^6}}\frac{\bar{D}}{n^{1-\epsilon}}\log^{\frac{3}{2}} n. \quad (5.33)
\end{aligned}$$

Compare (5.32) and (5.33). Since $\bar{D} = o(n^d), d < 1$, inequality (5.33) will eventually be the loosest for large n. Hence, the optimal capacity is upper bounded by

$$\lambda^2 \leq O(\frac{\bar{D}}{n^{1-\epsilon}}\log^{\frac{3}{2}} n).$$

We can now take $\epsilon \to 0$ and obtain the following upper bound on the capacity-delay tradeoff under the *RWP* mobility model:

$$\lambda^2 \leq O(\frac{\bar{D}}{n}\log^{\frac{3}{2}} n).$$

5.8.2 Optimal Values of the Key Scheduling Parameters

We can also identify the optimal values of the parameters R_b, A_b and h_b for achieving the above upper bound. Towards this end, we can infer as in Section 5.6.1

again that the scheduling policy should use the same parameters for all bits. Hence, we will assume that all key parameters (such as R_b, l_b, etc.) are indeed the same for all bits. Assume that the mean delay is bounded by $n^d, d < 1$. By earlier derivations, we have,

$$\lambda \leq \Theta(n^{\frac{d-1+\epsilon}{2}} \log^{\frac{3}{4}} n).$$

Note that equality holds in (5.30) if and only if

$$\frac{c_1}{2} n^{-\frac{1}{2}-\epsilon} \frac{(\lambda nT)^2}{\bar{D} \sum_{b=1}^{\lambda nT} \mathbf{E}[l_b]} = \frac{2\pi}{WTn} (\sum_{b=1}^{\lambda nT} \mathbf{E}[l_b])^2.$$

Substituting $\sum_{b=1}^{\lambda nT} \mathbf{E}[l_b] = \lambda nT l_b$, we can solve for l_b,

$$\begin{aligned} \frac{c_1}{2} n^{-\frac{1}{2}-\epsilon} \frac{\lambda nT}{\bar{D} l_b} &= \frac{2\pi}{WTn} (\lambda nT)^2 l_b^2 \\ l_b^3 &= \frac{c_1}{4\pi} \frac{W}{\bar{D} \lambda n^{\frac{1}{2}+\epsilon}}. \end{aligned}$$

Substituting $\lambda = \Theta(n^{(d-1+\epsilon)/2} \log^{3/4} n)$ and $\bar{D} = n^d$, we obtain the optimal value of l_b,

$$l_b = \Theta(n^{-\frac{d+\epsilon}{2}} \log^{-\frac{1}{4}} n).$$

A reasonable choice for the area of capture neighborhood, A_b, is then,

$$A_b = l_b^2 = \Theta(n^{-d-\epsilon}/\sqrt{\log n}).$$

By setting (5.11) of Tradeoff IV to equality, we have

$$\begin{aligned} \lambda nT h_b &= \frac{WTn}{2} \\ h_b &= \frac{W}{2\lambda} = \Theta(n^{\frac{1-d-\epsilon}{2}} / \log^{\frac{3}{4}} n). \end{aligned}$$

By setting (5.8) of Tradeoff II to equality, we have

$$S_b^h = \frac{l_b}{h_b} = \Theta(\sqrt{\frac{\log n}{n}}).$$

From (5.26), equality is attained when

$$R_b \;=\; \Theta(\frac{c_1 n^{\frac{1}{2}-\epsilon}}{l_b \bar{D}}) = \Theta(n^{\frac{1-d-\epsilon}{2}} \log^{\frac{1}{4}} n).$$

Taking $\epsilon \to 0$, we summarize the optimal values of the scheduling parameters below when the delay constraint is $D_b = O(n^d)$:

$$R_b = \Theta(n^{\frac{1-d}{2}} \log^{\frac{1}{4}} n),\ l_b = \Theta(n^{-\frac{d}{2}} / \log^{\frac{1}{4}} n),$$
$$h_b = \Theta(n^{\frac{1-d}{2}} / \log^{\frac{3}{4}} n),\ \text{and}\ S_b^h = \Theta(\sqrt{\frac{\log n}{n}}).$$

5.8.3 The Capacity-Achieving Scheme

We can now use these optimal values to construct the cell-based capacity-achieving scheme as in Section 5.6. According to the optimal values of R_b and l_b, the number of sending cells and receiving cells should be $g_1(n) = \Theta(\frac{n^{\frac{1+d}{2}}}{\log n})$ and $g_2(n) = \Theta(n^d)$, respectively. We have simulated this cell-based scheme under the *RWP* mobility model and found that it can achieve the following capacity-delay tradeoff:

$$\lambda^2 \geq \Theta(\frac{\bar{D}}{n} / \log^2 n) \text{ when } 0.5 < d < 1.$$

Note that this capacity-delay tradeoff is better than the tradeoff reported in earlier studies [84]. Analogous to Section 5.7, we can show that a restrictive choice of the receiving cell size is again the performance limiting factor of the scheme in [84].

5.9 The Brownian Motion Mobility Model

We now study the capacity-delay tradeoff under the Brownian Motion mobility model[9]. Interestingly, we will see that the results in this section are very different from those reported in earlier sections for the *i.i.d.* mobility model and the random

[9]We note that the results of this section can also be easily extended to other related mobility models such as the random walk mobility model [92] and the Markovian mobility model [75]. This is because the Brownian motion model can be viewed as a limiting case of these other mobility models.

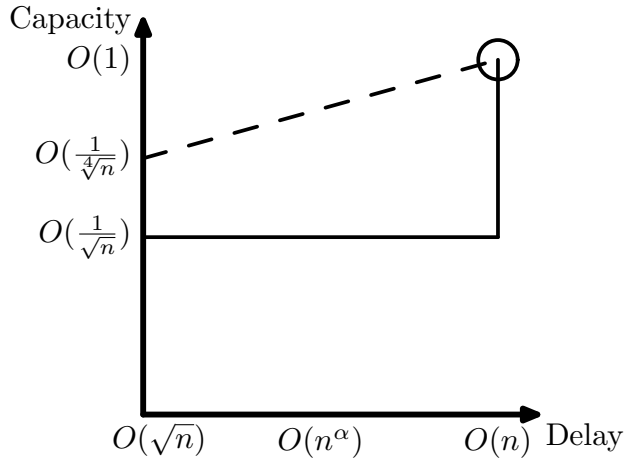


Fig. 5.9. The degenerate delay-capacity tradeoff under the Brownian motion model (the solid line) compared with the "smooth" delay-capacity tradeoff under the random way-point mobility model (the dashed line) reported in Section 5.8. We have chosen $\sigma_n^2 = 1/n$ and ignored all logarithmic terms in the figure.

way-point mobility model. We will show that there is *virtually no tradeoff between the capacity and delay under the Brownian motion model* (see Fig. 5.9). In particular, we show that under a large class of scheduling and relaying schemes, in order to achieve a delay of $\Theta(n^\alpha/\sigma_n^2)$ for any $\alpha < 0$, the per-node capacity must be $O(1/\sqrt{n})$. (Recall that σ_n^2 is the variance parameter of the Brownian motion. See the definition in Section 5.2.) Note that one can achieve $\Theta(1/\sqrt{n \log n})$ per-node capacity for static wireless networks using multi-hop transmission [71]. On the other hand, schemes have been developed that can achieve $\Theta(1)$ per-node capacity at $\Theta(\log n/\sigma_n^2)$ delay [83][10]. Thus, in order to achieve any significant capacity gains by exploiting mobility, one must be ready to tolerate huge delays, roughly on the order of $\Theta(1/\sigma_n^2)$, which is close to the delay at $\Theta(1)$ per-node capacity.

[10]We are referring to Scheme 2 in [83]. There is a missing $\log n$ factor in the delay result in [83] for Scheme 2. Independent of our work [27], the authors of [83] have also identified the missing $\log n$ factor (see [86]).

5.9.1 Basic Properties of Brownian Motion on a Sphere

Before we state the main result, we first summarize some basic properties of Brownian motion on a sphere, which will be used later on. Let us consider the motion of a single node. Let X_t denote its position at time t, which can be represented using the spherical coordinates (θ_t, ϕ_t). Recall the definition of Brownian motion on a sphere (see (5.4) and (5.5)). For analysis, it is useful to project each node's position onto the z-axis. Substituting $Y_t = \cos\theta_t$ into (5.4), and using Itô's Lemma, we obtain

$$dY_t = -\sigma_n^2 Y_t dt - \sigma_n \sqrt{1 - Y_t^2} dB_t. \tag{5.34}$$

Note that Y_t is a diffusion process with *drift coefficient* $-\sigma_n^2 Y_t$ and *diffusion coefficient* $\sigma_n^2(1 - Y_t^2)$.

We first cite the following result from [87] concerning the expected travel time of Y_t:

Lemma 5.9.1 *Let $-1 < a < x < 1$. Then, in traveling from x to a, Y_t takes an expected time, $V_a(x)$, given by:*

$$V_a(x) = \frac{2}{\sigma_n^2} \log\left(\frac{1+x}{1+a}\right). \tag{5.35}$$

The First Hitting Time

The first concept we study is the *first hitting time*. Let A be an arbitrary region on the sphere. We have the following definition:

Definition 5.9.1 *The first hitting time of A, denoted by T_A, is the first time instant at which X_t enters A; i.e., $T_A = \inf\{t \geq 0 : X_t \in A\}$.*

Let Π denote the uniform distribution on the unit sphere S, and denote by $\mathbf{E}_\Pi$ the expectation conditioned on X_0 being distributed according to Π. Let $A = \{x \in S : d_S(x, y) \leq a_n\}$, where y is an arbitrary point on S, d_S denotes the geodesic distance on the sphere, and $a_n > 0$. For $a_n \downarrow 0$, as $n \to \infty$, we have the following result:

Lemma 5.9.2 $\mathbf{E}_{\Pi}[T_A] = \Theta(\log(1/a_n)/\sigma_n^2)$.

Proof In view of the symmetry of the sphere, taking y to be the south pole (i.e., the bottom most point of S that corresponds to $\theta = \pi$) entails no loss of generality. Now, for $x \in A$, we have $\mathbf{E}[T_A|X_0 = x] = 0$. For $x \notin A$, let z_x denote its z co-ordinate. Note that the radius of S is $\frac{1}{2\sqrt{\pi}}$. The first time that X_t enters A is also the first time that Y_t travels from z_x to $-\cos(2a_n\sqrt{\pi})$. Using Lemma 5.9.1, we obtain

$$\mathbf{E}[T_A|X_0 = x] = \frac{2}{\sigma_n^2} \log\left(\frac{1 + z_x}{1 - \cos(2a_n\sqrt{\pi})}\right).$$

Integrating over all possible positions of the point x on S, and using the fact that x is uniformly distributed on S, we obtain

$$\mathbf{E}_{\Pi}[T_A] = \int_{\theta=2\sqrt{\pi}a_n}^{\pi} \frac{\sin\theta}{\sigma_n^2} \log\left(\frac{1 + \frac{\cos\theta}{2\sqrt{\pi}}}{1 - \cos(2a_n\sqrt{\pi})}\right) d\theta,$$

and the result follows after straightforward calculations. ■

Remark: If $a_n = n^{\alpha}, \alpha < 0$, then by Lemma 5.9.2, the first hitting time is always $\Theta(\log n/\sigma_n^2)$, regardless of the value of α. Even if we take $a_n = \sqrt{\pi}/4$, which means that the set A covers about half of the sphere, the first hitting time is still $\Theta(1/\sigma_n^2)$. Hence, the first hitting time changes very little when the size of the set A is increased. This result reveals the fundamental difference between the mobility pattern under the Brownian motion model and that under other mobility models (such as the *i.i.d* mobility model and the random way-point mobility model that we have studied in the previous sections). In these other models, the first hitting time for a set A decreases substantially when the size of the set A is increased [24, 84]. On the other hand, Lemma 5.9.2 is not completely surprising given the fact that, under the Brownian motion model, the node always wanders around like a "drunkard." Therefore, it is very difficult for the node to move towards any given destination.

The First Exit Time

The second concept that we study is the *first exit time*.

Definition 5.9.2 *Let $A = \{x \in S : d_S(x, y) \leq a_n\}$. The first exit time for the region A, denoted by τ_A, is the first instant of time at which the Brownian motion started at y (the center of A) exits A, i.e.,*

$$\tau_A = \inf\{t \geq 0 : X_0 = y, X_t \notin A\}.$$

Assuming $a_n \to 0$ as $n \to \infty$, we have the following result:

Lemma 5.9.3 $\mathbf{E}[\tau_A] = \Theta(a_n^2/\sigma_n^2)$.

Proof Using the symmetry of the sphere, we can set y to be the north pole of S (i.e., the top most point of S that corresponds to $\theta = 0$). It then follows that $\mathbf{E}[\tau_A]$ is the expected travel time of Y_t from 1 to $\cos(2a_n\sqrt{\pi})$. Applying Lemma 5.9.1 and performing some straightforward calculations, the result follows. ■

Remark: From the above discussion it is clear that under the Brownian motion model a node requires $\Theta(a_n^2/\sigma_n^2)$ time to move a radial distance of a_n. Thus the time a Brownian motion process spends in a region is in proportion to the area of the region. This also points to the well-known result that the path of Brownian motion is nowhere differentiable [88, p380]. Hence, it is inappropriate to define the "velocity" of a node that is moving in accordance with the Brownian motion model.

5.9.2 The Degenerate Capacity-Delay Tradeoff

In this section, we show that there is virtually no tradeoff between the delay and the capacity under the Brownian motion model. Specifically, we will show that whenever the delay constraint is $O(n^\alpha/\sigma_n^2)$ for any $\alpha < 0$, the per-node capacity is $O(1/\sqrt{n})$. In order to provide the readers with the main insight underlying this result, we use a slightly different network model in this section. We assume that the nodes are executing independent Brownian walks within a *unit square on a plane* (instead of on a *unit sphere*). This change simplifies the exposition substantially. Nonetheless, as we will see later, our results hold for a unit sphere as well.

Consider n nodes on a unit square centered at the origin, executing independent two-dimensional Brownian motions within the square. As will become clear soon, our result does not depend on how the boundary condition is handled: the Brownian motion could either be reflected at the boundary, or wrap around the boundary (like the 2-d torus model in [83]).

In order to prove the main result of this section, namely, that the delay-capacity tradeoff under the Brownian motion model is degenerate, we need some supporting results (Lemma 5.9.4 and Lemma 5.9.6 below). The main idea is as follows: if the delay is $O(n^\alpha/\sigma_n^2)$ for $\alpha < 0$, then we can show that the contribution due to node mobility in the packet delivery is likely very small. Hence, in order to deliver the packet to the destination, relaying over order $\Theta(1)$ distance is required. We can then show that the per-node capacity must be $O(1/\sqrt{n})$.

We start by showing that, if the delay is $O(n^\alpha/\sigma_n^2)$ for $\alpha < 0$, then the contribution due to node mobility in the packet delivery is likely very small. Let $\mathbf{SQ}(c_n)$ be the square centered at the origin with length c_n (see Fig. 5.10). Suppose there are $k_n \leq n$ nodes, starting at the origin at time 0. Each node then moves according to a two-dimensional Brownian motion with variance σ_n^2, which can be viewed as the composition of two independent one-dimensional Brownian motion along the x-axis and the y-axis, respectively, each having a variance of $\sigma_n^2/2$. Let $p_{k_n}(c_n, t_n)$ denote the probability of the event that one or more of the k_n nodes ever exit the square $\mathbf{SQ}(c_n)$ within time t_n. We have the following result concerning $p_{k_n}(c_n, t_n)$:

Lemma 5.9.4 *If there exists $N_0 < \infty$ such that*

$$\frac{c_n^2}{t_n} \geq 8\sigma_n^2 \log n, \ \textit{for } n \geq N_0, \tag{5.36}$$

then

$$\lim_{n\to\infty} p_{k_n}(c_n, t_n) = 0.$$

The following corollary is an immediate consequence of Lemma 5.9.4.

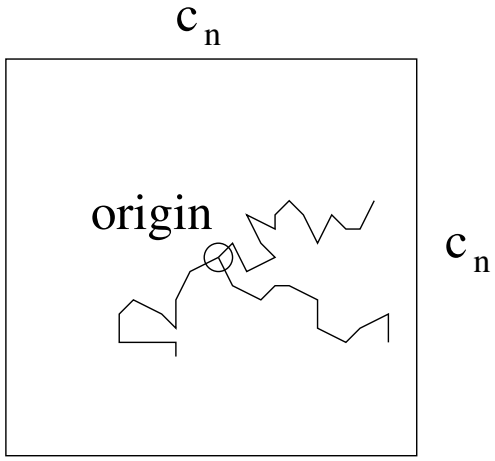


Fig. 5.10. k_n nodes at the origin

Corollary 5.9.5 *If*

$$\liminf_{n\to\infty} c_n \log n = c > 0$$
$$\limsup_{n\to\infty} \frac{\sigma_n^2 t_n}{n^\alpha} = c' < +\infty, \text{ for some } \alpha < 0 \ ,$$

then

$$\lim_{n\to\infty} p_{k_n}(c_n, t_n) = 0.$$

Remark: Corollary 5.9.5 shows that, within $O(n^\alpha/\sigma_n^2)$ time ($\alpha < 0$), none of the k_n nodes can possibly travel a $\Theta(1/\log n)$ distance in *any* direction.

Proof **[of Lemma 5.9.4]** Consider an arbitrary node. Let X_t be its position at time t. Let B_t^x and B_t^y denote its x-coordinate and y-coordinate, respectively. Then B_t^x and B_t^y are independent one-dimensional Brownian motions with variance $\sigma_n^2/2$. Let $p(c_n, t_n)$ be the probability that this particular node ever exits the square $\mathbf{SQ}(c_n)$ within time t_n. Let

$$\tau_x^+ \triangleq \inf\{t \geq 0 : B_t^x = c_n/2\},$$
$$\tau_x^- \triangleq \inf\{t \geq 0 : B_t^x = -c_n/2\},$$

and let τ_y^+, τ_y^- be similarly defined with B_t^y in place of B_t^x. Using the union bound, and appealing to the symmetry of the two-dimensional Brownian motion, we obtain

$$\begin{aligned} p(c_n, t_n) &\leq \mathbf{P}\{\tau_x^+ \leq t_n \text{ or } \tau_x^- \leq t_n \text{ or } \tau_y^+ \leq t_n \text{ or } \tau_y^- \leq t_n\} \\ &\leq 4\mathbf{P}\{\tau_x^+ \leq t_n\}. \end{aligned}$$

Further, using the Reflection Principle for one-dimensional Brownian motion [88, p394], we have

$$\mathbf{P}\{\tau_x^+ \leq t_n\} = 2\mathbf{P}\{B_{t_n}^x \geq c_n/2\}.$$

Since the distribution of $B_{t_n}^x$ is Gaussian with zero mean and variance $\sigma_n^2 t_n/2$, we have,

$$\mathbf{P}\{\tau_x^+ \leq t_n\} = 2\int_{\frac{c_n}{\sqrt{2}\sigma_n\sqrt{t_n}}}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du.$$

Using the inequality,

$$\begin{aligned} \int_x^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du &\leq \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \frac{u}{x} \exp\left(-\frac{u^2}{2}\right) du \\ &= \frac{1}{\sqrt{2\pi}x} \exp\left(-\frac{x^2}{2}\right), \end{aligned}$$

we have,

$$\mathbf{P}\{\tau_x^+ \leq t_n\} \leq 2\sqrt{\frac{\sigma_n^2 t_n}{\pi c_n^2}} \exp\left[-\frac{c_n^2}{4\sigma_n^2 t_n}\right].$$

Using (5.36), we have

$$\mathbf{P}\{\tau_x^+ \leq t_n\} \leq \frac{1}{\sqrt{2\pi \log n}} \exp(-2\log n) = \frac{1}{n^2\sqrt{2\pi \log n}}.$$

Hence,

$$p(c_n, t_n) \leq \frac{4}{n^2\sqrt{2\pi \log n}}.$$

Finally, since there are k_n nodes, each of them moves according to a two-dimensional Brownian Motion, we have

$$p_{k_n}(c_n, t_n) \leq k_n p(c_n, t_n) \leq \frac{4k_n}{n^2\sqrt{2\pi \log n}}.$$

Noting that $k_n \leq n$, the result then follows. ■

By Corollary 5.9.5, if the delay is $O(n^{\alpha}/\sigma_n^2)$ for $\alpha < 0$, then the contribution due to node mobility to the packet delivery is likely very small ($O(1/\log n)$). Hence, in order to deliver the packet to the destination, relaying over order $\Theta(1)$ distance is required. We now show that if each packet is relayed over $\Theta(1)$ distance (on an average), then the per-node capacity must be $O(1/\sqrt{n})$. Consider a large enough time interval $\mathcal{T}$. The total number of packets communicated end-to-end between all source-destination pairs during the interval is $c_p\lambda n\mathcal{T}$, where $1/c_p$ is the number of bits per packet. Let h_p be the number of times the packet p is relayed, and let l_p^h, for $h = 1, ..., h_b$, denote the transmission range for the h-th relaying. We have the following result:

Lemma 5.9.6 *Suppose that there exists a constant $c > 0$, such that on average each packet is relayed over a total distance no less than c, i.e.,*

$$\frac{\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b} l_p^h}{c_p\lambda n\mathcal{T}} \geq c, \tag{5.37}$$

then

$$\lambda \leq O(1/\sqrt{n}).$$

Proof We use the idea in Section 5.4.4 again that disks of radius $\frac{\Delta}{2}$ times the transmission range centered at the transmitter are disjoint from each other. We can therefore measure the radio resources that each transmission consumes by the areas of these disjoint disks. Note that the total area of the square is 1; for each of these disks, at least 1/4 of it must lie inside the unit square; and each relaying of a packet lasts $\frac{1}{c_pW}$ amount of time. Thus,

$$\frac{1}{4}\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b}\pi\left[\frac{\Delta}{2}l_p^h\right]^2 \leq c_pW\mathcal{T}. \tag{5.38}$$

By Cauchy-Schwarz Inequality,

$$\left[\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b} l_p^h\right]^2 \leq \left[\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b}(l_p^h)^2\right]\left[\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b}1\right]. \tag{5.39}$$

Further, since there are at most n simultaneous transmissions at any given time in the network, we have

$$\sum_{p=1}^{c_p\lambda nT} h_p \leq c_p W\mathcal{T}n. \tag{5.40}$$

Therefore,

$$\begin{aligned}
\frac{16c_pW\mathcal{T}}{\pi\Delta^2} &\geq \sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b}(l_p^h)^2 \qquad \text{(using (5.38))} \\
&\geq \frac{\left[\sum_{p=1}^{c_p\lambda nT}\sum_{h=1}^{h_b} l_p^h\right]^2}{\left[\sum_{p=1}^{c_p\lambda nT} h_p\right]} \qquad \text{(using (5.39))} \\
&\geq \frac{(c_p\lambda n\mathcal{T}c)^2}{c_pW\mathcal{T}n} \qquad \text{(using (5.37) and (5.40)).}
\end{aligned}$$

Hence,

$$\lambda \leq \sqrt{\frac{16W^2}{\pi\Delta^2c^2}}\frac{1}{\sqrt{n}}.$$

■

We are now ready to prove the main result of this section. We will consider the general class of scheduling policies outlined in Section 5.3, except that now we impose an additional restriction as follows. Recall the notion of replication and capture in Section 5.3. We will restrict our study to the class of scheduling policies that satisfy the following assumption:

Assumption A:

- Only the source of a packet is allowed to replicate the packet. That is, relay nodes holding a packet are *not* allowed to replicate it further.

Remark: Note that almost all scheduling schemes that have been proposed in the literature satisfy Assumption A [23, 24, 75, 76, 83–85].

It is worthwhile to elaborate on Assumption A a little bit, since it may seem restrictive at first sight. First, observe that the notions of *replication* and *relaying* are different, even though both involve forwarding packets to other relay nodes. For example, when node i decides to *replicate* the packet p to node j, node i can either

transmit the packet directly to node j, or use multi-hop transmission; i.e., node i can forward the packet to another node k, and let node k *forward* the packet to node j. (Node k may also keep the copy of the packet p, in which case, both nodes k and j are considered to receive the packet due to the *same* replication decision initiated by node i.) In this example, although both nodes i and k forward the packet p to other nodes, their roles are different. Node i is the one who *initiates* the replication, while node k is just *passively following* the instruction of node i to *relay* the packet to node j. Thus, we see that Assumption A only prohibits relay nodes to *initiate* replication. Hence, multi-hop relaying is still allowed under Assumption A. (Multi-hop relaying is also allowed for the relay-to-destination communication, i.e., capture.)

If we attempt to develop *distributed* scheduling policies where nodes make replication decisions and capture decisions without any knowledge of the decisions at other nodes, then restricting the replication decisions to the source node is a natural way to *control the number of copies of a packet in the system*. Note that excessive redundancy would reduce the system throughput substantially. The source node of a packet p is in the best position to control both the total number of replications for the packet and the number of relay nodes getting the packet for each replication. If the relay nodes were allowed to replicate, then additional cooperation among the relay nodes would most likely be required in order to limit the number of replicas of a packet (see, for example, the scheme in [79], where the relay nodes know the location of the static destination, and also have some knowledge of the future direction of other nodes' movement, based on which they can *cooperate* to make selective and more efficient replication toward the destination).

We can prove the following main result:

Proposition 5.9.1 *Let $\bar{D}$ denote the expected delay averaged over all packets and all source-destination pairs, and let λ denote the throughput of each source-destination pair. For any scheduling policy that satisfies Assumption A, if*

$$\bar{D} \leq O(n^{\alpha}/\sigma_n^2), \alpha < 0,$$

then

$$\lambda \leq O(1/\sqrt{n}).$$

Proof Consider squares A and B of length 1/4, centered at $(-1/4, 1/4)$ and $(1/4, -1/4)$, respectively (see Fig. 5.11). Since the packet arrivals are independent of the positions of the mobile nodes, there will be a constant fraction f_0 of the packets that have their source nodes in square A and destinations in square B, at the time of arrival. (If the stationary distribution of the positions of the nodes are uniform, then $f_0 = \left(\frac{1}{4}\right)^4 = 1/256$. Otherwise, f_0 is still a positive constant independent of n.) Let Φ_{AB} denote this set of packets. In order to ensure that $\bar{D} \leq O(n^\alpha/\sigma_n^2)$, the delay for the packets in Φ_{AB} has to be $O(n^\alpha/\sigma_n^2)$. Precisely, since $\bar{D} \leq O(n^\alpha/\sigma_n^2)$, there exists some $N_0 > 0$ and $c_1 > 0$, such that

$$\bar{D} \leq c_1 n^\alpha/\sigma_n^2, \text{ when } n \geq N_0. \tag{5.41}$$

Therefore, the delay of at least half of the packets in Φ_{AB} must be no greater than

$$t_n = \frac{2c_1}{f_0}\frac{n^\alpha}{\sigma_n^2}.$$

(Otherwise, the delay of the other half of the packets in Φ_{AB} must be greater than t_n. Because this other half contributes to at least $f_0/2$ fraction of all packets, the condition (5.41) will be violated.) Let Φ_{AB}^0 denote the set of packets in Φ_{AB} whose delay is no greater than t_n. Consider an arbitrary packet p which is in Φ_{AB}^0. Let S_p and D_p denote its source node and destination node, respectively. Fig. 5.12 shows a typical packet delivery. The source nodes S_p moves from position S_0 to U_1, and replicates the packet p to a relay node, say r_1, at position V_1, possibly using multi-hop transmission. The node r_1 then moves independently of S_p. The source node moves on to position U_2, where it replicates the packet p to one more relay node, say r_2, positioned at V_2, and so on. It is also possible to replicate the packet to more than one relay node at the same time (for example, we can take $U_1 = U_2$ if the source node replicates the packet to r_1 and r_2 at the same time). At time $t \leq t_n$, a successful capture occurs, as one of the relay nodes holding the packet p (node r_2

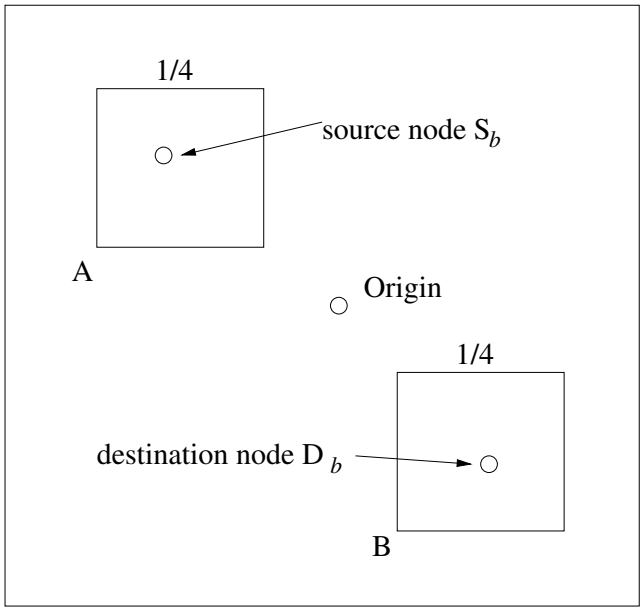


Fig. 5.11. There exists a constant fraction of packets that originate from nodes in A and are destined to nodes in B.

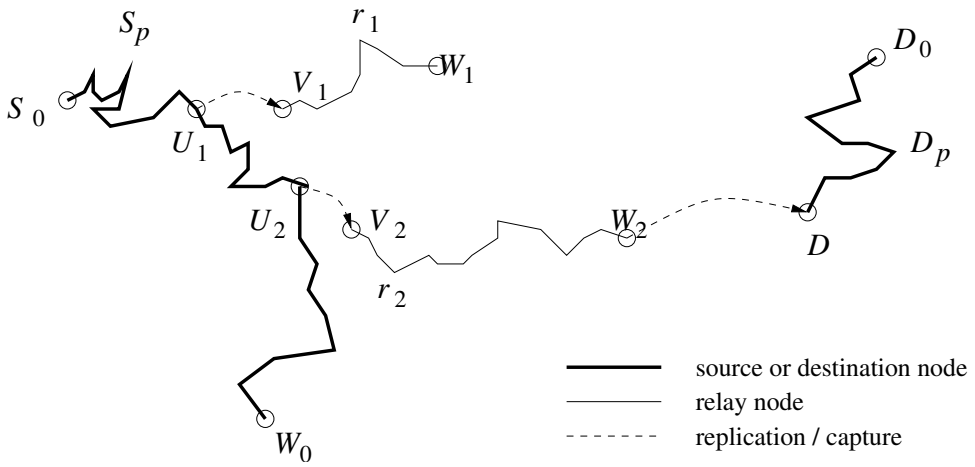


Fig. 5.12. How a typical packet p is delivered.

in the case shown in Fig. 5.12) decides to forward the packet to its destination node D_p, which has moved from its initial position D_0 to the position D, at time t. Let k_n denote the total number of relay nodes that hold packet p in this process, and let r_k, for $k = 1, 2, ..., k_n$, denote the k-th relay. Let U_k and V_k denote the position of the source node S_p and the position of the relay node r_k, respectively, at time of *replication*. Let W_k denote the position of the relay node r_k at the time of *capture* (see Fig. 5.13). Let $d(x, y)$ denote the Euclidean distance between positions x and y within the unit square. Since the direct straight-line path is always the shortest path connecting any two points, we have, for any k,

$$d(S_0, U_k) + d(U_k, V_k) + d(V_k, W_k) + d(W_k, D) + d(D_0, D) \geq d(S_0, D_0).$$

Hence,

$$d(U_k, V_k) + d(W_k, D) \geq d(S_0, D_0) - d(D_0, D) - d(S_0, U_k) - d(V_k, W_k). \quad (5.42)$$

Since S_0 and D_0 are in the squares A and B, respectively,

$$d(S_0, D_0) \geq \frac{\sqrt{2}}{4}.$$

Further, each of the terms $d(D_0, D)$, $d(S_0, U_k)$, and $d(V_k, W_k)$, corresponds to the movement of a different node. There are at most n nodes involved in this process. By

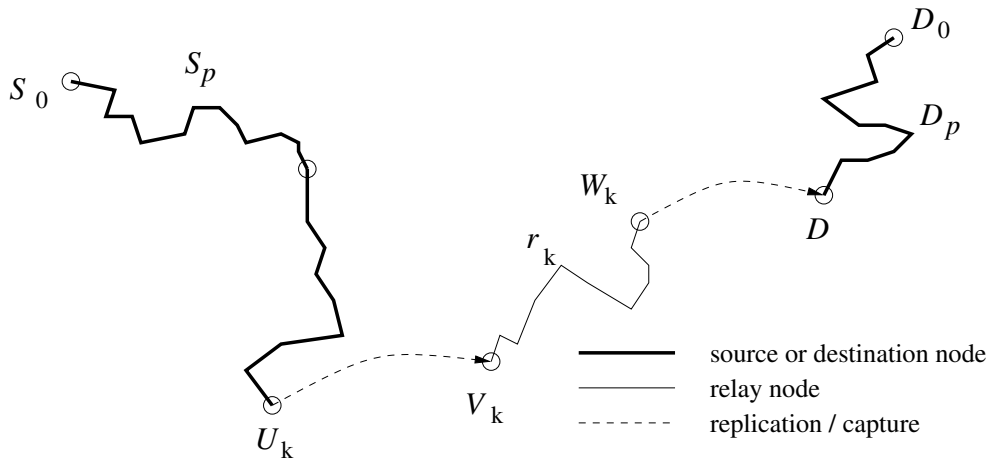


Fig. 5.13. The relay node r_k

setting $c_n = 1/\log n$ in Corollary 5.9.5, we can see that, with probability approaching 1 as $n \to \infty$, all of the last three terms in (5.42) are no greater than $\sqrt{2}/\log n$, for all k. Therefore,

$$d(U_k, V_k) + d(W_k, D) \geq \frac{\sqrt{2}}{4} - 3\frac{\sqrt{2}}{\log n} \geq 1/4$$

for large enough n. Finally, let W_0 denote the position of the source node at time t. Then using a similar argument,

$$d(W_0, D) \quad \geq d(S_0, D_0) - d(D, D_0) - d(S_0, W_0) \geq 1/4.$$

This shows that for each packet p in Φ^0_{AB}, the total distance that the packet p has to be relayed is at least $1/4$. Since Φ^0_{AB} contributes to at least $f_0/2$ fraction of all packets, on an average each packet must be relayed over a distance no less than $f_0/8 > 0$. Hence, by Lemma 5.9.6, the per-node throughput λ must be no larger than $O(1/\sqrt{n})$. ■

Remark: For the ease of exposition, we have shown the above results for Brownian motion on a plane. However, it is not difficult to see that the argument in Proposition 5.9.1 applies to Brownian motion on a unit sphere as well. In particular, in Lemma

5.9.4, if we choose $c_n = c/\log n$, the size of the square $\mathbf{SQ}(c_n)$ diminishes to zero as $n \to \infty$. Hence, the difference between such a square on a plane and that on a unit sphere vanishes. Therefore, both Corollary 5.9.5 and Proposition 5.9.1 hold for Brownian motion on a unit sphere as well.

The Degenerate Tradeoff: Proposition 5.9.1 shows that the *capacity-delay tradeoff under the Brownian motion model is degenerate*: For delay less than $O(n^\alpha/\sigma_n^2), \alpha < 0$, the per-node capacity is at most $O(1/\sqrt{n})$. Since one can achieve $\Theta(1/\sqrt{n \log n})$ per-node capacity for static wireless networks using multi-hop transmission [71], our result shows that whenever the delay is constrained to be less than $O(n^\alpha/\sigma_n^2), \alpha < 1$, Brownian mobility cannot improve the capacity by more than a logarithmic factor. Further, since the packet transmissions are usually carried out at a much faster time-scale than the node mobility, one could view the delay under the multi-hop scheduling (see [71]) as being *almost zero*. Earlier studies have shown that it is possible to achieve $\Theta(1)$ per-node capacity at roughly $\Theta(1/\sigma_n^2)$ delay under the Brownian motion model [83]. Obviously, $\Theta(1)$ is an upper bound on the per-node capacity (under our network model). Hence, if we ignore the logarithmic terms, the capacity-delay tradeoff under the Brownian motion model degenerates into two points: one can either achieve $\Theta(1/\sqrt{n \log n})$ per-capacity at almost no delay, or $\Theta(1)$ per-node capacity at roughly $\Theta(1/\sigma_n^2)$ delay, but nothing in between! Finally, although Proposition 5.9.1 is shown under the Brownian motion model, it is not difficult to see that the result also applies to the random walk mobility model [92], or the Markovian mobility model in [75]. This is because as $n \to \infty$, the difference between these mobility models vanishes.

The result of Proposition 5.9.1 is in sharp contrast to the results reported in the existing works [83,85], where it is claimed that certain schemes can provide a smooth tradeoff between the capacity and the delay under the Brownian motion model. For a discussion of the reasons for these discrepancies, please refer to [27].

5.10 Conclusion and Future Work

In this chapter, we have studied the fundamental capacity-delay tradeoff in mobile wireless networks. We have developed a systematic methodology to study this problem. Unlike previous works that start from some specific scheduling schemes and try to prove that the particular scheme achieves the optimal capacity-delay tradeoff, we assume the most general class of scheduling schemes and start our study by investigating the inherent tradeoffs in the system among the capacity, the delay and various key scheduling parameters. Based on these inherent tradeoffs (in the form of inequalities), we can not only derive the true upper bound on the per-node capacity of a large mobile wireless network under given delay constraints, but also identify the the optimal values of the key scheduling parameters and construct the corresponding capacity-achieving schemes. Our methodology also allows us to identify the limiting factors in existing schemes. In Table 5.2, we have compared the results in this chapter with those in previous works. The previous works often put various implicit restrictions on the key scheduling parameters (i.e., R_b, l_b and h_b) in their model. As we have seen in the results of this chapter, such restrictions can significantly limit the achievable performance of the network. Even if these restrictions may have practical significance, it is still important to understand their implications, which can only be revealed through a systematic methodology as we have gone through in this chapter.

The results in this chapter can be generalized in several directions. For example, the *i.i.d.* mobility model can be extended to incorporate "pause times." Specifically, at each time slot, each node may independently choose to stay in its old position with probability p, and to move to a new random position with probability $1 - p$. This model may approximate scenarios where nodes move at a fast speed and then stay for a relatively long period of time. Tradeoff I in Section 5.4 will hold for this extension of the *i.i.d.* mobility model, and hence our main results in Sections 5.5 and 5.6 will hold as well. Our result for the Brownian motion model can also be extended to

Table 5.2
Restrictions on the parameters R_b, l_b and h_b may limit the capacity-delay tradeoff

Scheme	Mobility Model	R_b	l_b	h_b	Capacity-Delay Tradeoff Achieved
Neely & Modiano [75]	*i.i.d.*	Vary	$O(1/\sqrt{n})$	Vary	$\lambda \leq O(\frac{D}{n})$
Toumpis & Goldsmith [76]	*i.i.d.*	Vary	Vary	1	$\lambda^2 = \Theta(\frac{D}{n}/\log^3 n)$
Our work [24–26]	*i.i.d.*	Vary	Vary	Vary	$\lambda^3 = \Theta(\frac{D}{n}/\log^{9/2} n)$
Sharma & Mazumdar [84]	Random Way-Point	Vary	$O(1/\sqrt{n})$	Vary	$\lambda \leq O(\frac{D}{n})$
Our work [25]	Random Way-Point	Vary	Vary	Vary	$\lambda^2 = \Theta(\frac{D}{n}/\log^2 n)$
Gamal *et al* [83]	Brownian Motion	2	Vary	Vary	
Sharma & Mazumdar [85]	Brownian Motion	Vary	$O(1/\sqrt{n})$	Vary	
Our work [27]	Brownian Motion	Vary	Vary	Vary	Degenerate

the other related models, such as the random walk model [86] and the Markovian Model [75].

An interesting observation of the results in this chapter is that the delay-capacity tradeoff in mobile wireless networks critically depends on the underlying mobility model. As we have seen in Section 5.9, the tradeoff is *degenerate* under the Brownian motion model. On the other hand, under the *i.i.d.* mobility model, we have shown in Section 5.5 that the following tradeoff between the per-node capacity λ and the delay D can be achieved:

$$\lambda = \Theta\left(\sqrt[3]{\frac{D}{n}}/\log^{3/2} n\right)$$

for $\Theta(1) \leq D \leq \Theta(n)$. Further, under the random way-point mobility model, our scheme in Section 5.8 can achieve the following delay-capacity tradeoff:

$$\lambda = \Theta\left(\sqrt[2]{\frac{D}{n}}/\log n\right)$$

for $\Theta(\sqrt{n}) \leq D \leq \Theta(n)$, when the speed of the nodes is $v(n) = 1/\sqrt{n}$. In Fig. 5.14, we illustrate the difference in the above three delay-capacity tradeoffs. In this figure, we have chosen $v(n) = 1/\sqrt{n}$ and $\sigma_n = 1/\sqrt{n}$. The reason for such choice of $v(n)$ and σ_n is to ensure that the *contact time* (i.e., the time for two nodes to remain neighbors of each other) is $\Theta(1)$ under all three mobility models. As we can see, a smooth tradeoff exists for any value of delay for the *i.i.d.* mobility model, while a smooth tradeoff only exists for delay between $\Theta(\sqrt{n})$ and $\Theta(n)$ under the random way-point mobility model, and the tradeoff degenerates to only two points under the Brownian motion model.

Looking at these results, a natural question to ask is: what will the tradeoff be for a *real* mobile wireless network? Will the tradeoff in real networks be one of these three types? Or will it be a combination of these three? A closely related question is whether these tradeoffs (along with their respective mobility models) represent three *distinct* cases, or are part of a continuous range of delay-capacity tradeoffs. Thus, it is important to understand the reasons behind these different tradeoffs, and

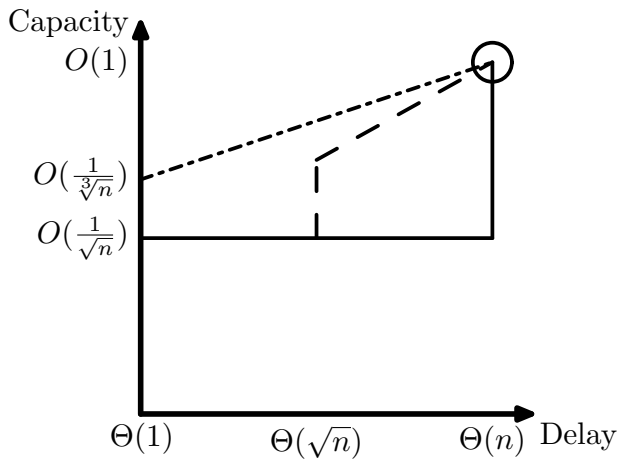


Fig. 5.14. The delay-capacity tradeoffs under the Brownian motion model (the solid line), the random way-point mobility model (the dashed line), and the *i.i.d.* mobility model (the dash-dotted line). We have chosen $v(n) = 1/\sqrt{n}$ and $\sigma_n = 1/\sqrt{n}$ and ignored all logarithmic terms in the figure.

to investigate whether a unified framework can be developed to understand these results. We plan to address these questions in our future work.

6. A LOOSE-COUPLING APPROACH TO CROSS-LAYER DESIGN IN MULTIHOP WIRELESS NETWORKS

6.1 Introduction

In Chapters 2-5, we have shown how one can obtain simple and efficient control solutions for large networks by exploiting the largeness of the system. The accuracy of such an approach varies according to the particular problem. For example, for wire-line networks where the capacity is large, we can obtain simple control solutions that are asymptotically exact in the sense that the gap between the simple solution and the optimal solution goes to zero as the scale of the system increases to infinity (see Chapters 2 and 3). However, for wireless networks where the number of nodes is large, our approach that exploits largeness has only allowed us to obtain results that are order-accurate (e.g., $\Theta(1/\sqrt[3]{n})$ versus $\Theta(1/\sqrt{n})$, see Chapter 5). Note that due to the scarcity of network resources in wireless systems, the constants before the order terms are often important in many practical scenarios (e.g., $1/n$ versus $10/n$). Thus, other approaches are required if we want to obtain tighter simplicity results for wireless networks.

In this chapter, we will demonstrate that simplification of network control can also be obtained through appropriately designing the control architecture. In particular, we will study the cross-layer design problem in multihop wireless networks. Here, simplicity is obtained in the sense that the complex cross-layer interactions in multihop wireless networks can be structured into layers that are only weakly dependent on each other through a judiciously chosen set of control parameters. We refer to our solution the *loose-coupling* approach to cross-layer design. By loose-

coupling, we mean that the cross-layer solution only requires a minimal amount of interaction between the layers, and is robust to imperfect decisions at each layer. We will demonstrate how cross-layer control solutions with this loose-coupling property can be developed for the cross-layer congestion control and scheduling problem in multihop wireless networks. These results allow us to use imperfect, but simpler and potentially distributed, algorithms for cross-layer control of large wireless networks. In fact, we will develop one such fully distributed algorithm for certain interference model by taking advantage of the loose-coupling approach.

6.1.1 Cross-Layer Design in Wireless Networks

Traditionally, communication networks have been engineered according to the layered architecture [28]. The functionality of the network is divided into layers. For example, the OSI reference model has seven layers: the physical layer, the data link layer, the network layer, the transport layer, the session layer, the presentation layer, and the application layer. The layering concept essentially treats each layer as a *black box*: the higher layer only needs to know the interface to the lower layer, but not the details of how the interface is implemented. Clearly, the layered architecture provides *modularity*, which contributes to *simplicity* and *scalability* of the entire system. Thus, the layered architecture has been a key contributing factor to the success of many network systems, including the Internet.

While such a layered approach has been very successful for wire-line networks, it has turned out to be increasingly inadequate for wireless networks. In wireless networks, there exists a natural coupling between different layers. For example, in a wireless system, the capacity of each radio link depends on the signal and interference levels, and thus depends on the power and transmission schedules at the other links. To make things more complicated, even if the power assignment and schedule at each link are known, the wireless channel exhibits variations due to fading [93–96]. These characteristics of wireless networks are in contrast to wire-line networks where the

capacity of each link is fixed and known. Hence, the set of end-to-end data rates that can be supported by a multihop wireless network (and that can be used by the high-layer protocol layers) is usually in a complex form that critically depends on the way in which resources at the underlying physical (PHY) and MAC layers (such as modulation, coding, and power control) are scheduled [72–74, 93, 94, 97].

Due to this tight coupling between layers, researchers have increasingly adopted a "cross-layer" approach for the design and control of multihop wireless networks [29, 72–74, 93–95, 97–104]. The rational is very simple, if one can optimize control variables at two layers together, a better performance can be achieved than the case where control variables at each layer are optimized independently. However, one needs to be very careful in developing such cross-layer solutions. In fact, one of the main arguments against cross-layer design is that it destroys modularity [105]. The argument is as follows: Although cross-layer solutions could potentially increase the performance of the system, the resultant tight interaction between the layers could make the overall system much more complex and fragile. The entire system may become sensitive to changes at each layer, i.e., changes at one layer could potentially lock the overall system into a substantially inefficient operating point, if the impact of such changes on the overall system behavior are not carefully accounted for. This is like an elaborate three dimensional puzzle: a beautiful replica emerges only when the pieces are placed in their correct positions. However, by removing one piece, the whole replica may fall apart.

Therefore, there is a fundamental tradeoff between *efficiency* and *modularity* in cross-layer control of wireless networks [105]. In this chapter, we address this tradeoff by proposing a *loose-coupling* approach to cross-layer design. By *loose-coupling*, we mean that the cross-layer control solution only requires a minimal amount of information shared across layers, and the solution should be robust to imperfect information or imperfect actions taken at different layers. In other words, the complex cross-layer interactions in the system will then be structured into layers that are only weakly dependent on each other through a judiciously chosen set of control

parameters. Clearly, if the solution has the loose-coupling property, it will achieve both efficiency and modularity, which is very desirable for wireless system design.

In the rest of the chapter, we will demonstrate how such a *loose-coupling* solution can be obtained for the cross-layer congestion control and scheduling problem in multihop wireless networks. The loose-coupling approach then provides us with a framework to design simple, efficient, and potentially distributed cross-layer control solutions for multihop wireless networks [29, 30]. In particular, we will demonstrate how to use the insights drawn from our analyses to design a simple and fully distributed cross-layer congestion control and scheduling algorithm under a particular interference model.

The rest of the chapter is organized as follows. In Section 6.2, we formulate the cross-layer congestion control and scheduling problem. In Section 6.3, we present the optimal solution for this problem, which demonstrates the loose-coupling property. In Section 6.4 and 6.5, we further investigate the loose-coupling property by studying the impact of imperfect scheduling on the cross-layer solution. In Section 6.6, we present a fully distributed cross-layered congestion control and scheduling algorithm. Then we conclude.

6.2 Problem Formulation

In this work, we investigate the problem of how to *fairly and efficiently* manage resources over multiple layers so as to maximize the *end-to-end* data rates that can be supported by the network without incurring excessive overload, delays, or packet losses. This is in fact the *congestion control problem*, i.e., determining the fair end-to-end rate allocation at which users should transmit information into the network. Congestion control is a key functionality in modern communication networks to avoid congestion and to ensure fairness among the users. In the standard OSI layering architecture [28], congestion control is categorized as functionality at the transport

layer. For the Internet, it is the functionality of the Transport Control Protocol (TCP).

Although congestion control has been studied extensively for *wireline* networks (see [61,106] for good references), these results cannot be applied directly to multihop wireless networks. In wireline networks, the *capacity region* (i.e., the set of feasible data rates) is of a simple form, i.e., the sum of the data rates at each link should be less than the link capacity, which is known and fixed. In multihop wireless networks, the capacity of each radio link depends on the signal and interference levels, and thus depends on the power and transmission schedule at other links. Hence, the capacity region is usually of a complex form that critically depends on the way in which resources at the underlying physical and MAC layers are scheduled. Therefore, the congestion control problem in wireless networks needs to be investigated as a cross-layer control problem jointly with control at the underlying MAC and physical layers.

Consider the following network model. There are N nodes in a multihop wireless network. Let $\mathcal{L}$ denote the set of node pairs (i,j) (i.e., links) such that direct transmission from node i to node j is allowed. The links are assumed to be directional. Due to the shared nature of the wireless media, the data rate r_{ij} of a link (i,j) depends not only on its own modulation/coding scheme and power assignment P_{ij}, but also on the interference due to the power assignments on other links. Let $\vec{P} = [P_{ij}, (i,j) \in \mathcal{L}]$ denote the vector of global power assignments and let $\vec{r} = [r_{ij}, (i,j) \in \mathcal{L}]$ denote the vector of data rates. We assume that $\vec{r} = u(\vec{P})$, i.e., the data rates are completely determined by the global power assignment[1]. The function $u(\cdot)$ is called the *rate-power function* of the system. Note that the global power assignment $\vec{P}$ and the rate-power function $u(\cdot)$ summarize the *cross-layer* control capability of the network at both the physical layer and the MAC layer. Precisely, the global power assignment determines the Signal-to-Interference-Ratio

[1]Although we have not considered channel variation, e.g., due to fading, our main results may be generalized to those cases.

(SIR) at each link. Given the SIR, each link can choose appropriate modulation and coding schemes to achieve the data rate specified by $u(\vec{P})$. Finally, the network can schedule different sets of links to be active (and to use different power assignments) at different time to achieve maximum capacity [73, 94, 97]. There may be constraints on the feasible power assignment. For example, if each node has a total power constraint $P_{i,\max}$, then $\sum_{j:(i,j)\in\mathcal{L}} P_{ij} \leq P_{i,\max}$. Let Π denote the set of feasible power assignments, and let $\mathcal{R} = \{u(\vec{P}), \vec{P} \in \Pi\}$. We assume that $\text{Co}(\mathcal{R})$, the convex hull of $\mathcal{R}$, is closed and bounded. We assume that time is divided into slots and the power assignment vector $\vec{P}(t)$ is fixed during each time slot t. We will refer to $\vec{r}(t) = u(\vec{P}(t))$ as the *schedule* at time slot t.

In the rest of the chapter, it is usually more convenient to index the links numerically (e.g., links $1, 2, ..., L$) rather than as node-pairs (e.g., link (i, j)). The power assignment vector and the rate vector should then be written as $\vec{P} = [P_1, ..., P_L]$ and $\vec{r} = [r_1, ..., r_L]$, respectively.

There are S users and each user $s = 1, ..., S$ has one path through the network[2]. Let $H = [H_s^l]$ denote the routing matrix, i.e., $H_s^l = 1$, if the path of user s uses link l, and $H_s^l = 0$, otherwise. Let x_s be the rate with which user s injects data into the network. Each user is associated with a utility function $U_s(x_s)$, which reflects the level of "satisfaction" of user s when its data rate is x_s. As is typically assumed in the congestion control literature, we assume that each user s has a maximum data rate M_s and the utility function $U_s(\cdot)$ is strictly concave, non-decreasing and twice continuously differentiable on $(0, M_s]$.

We now define the *capacity region* of the system. We say that a system is *stable* if the queue lengths at all links remain finite. We say that a user rate vector $\vec{x} = [x_1, ..., x_s]$ is *feasible* if there exists a scheduling policy that can *stabilize* the system under user rates $\vec{x}$. We define the *capacity region* to be the set of *feasible* rates $\vec{x}$. It

[2]Extensions to the case with multipath routing are also possible (see [29]).

has been shown in [72–74] that the *optimal capacity region* Λ is a convex set and is given by

$$\Lambda = \left\{ \vec{x} \left| [\sum_{s=1}^{S} H_s^l x_s] \in \mathrm{Co}(\mathcal{R}) \right. \right\}, \tag{6.1}$$

where $\sum_{s=1}^{S} H_s^l x_s$ can be interpreted as the total data rate on link l. The convex hull operator $\mathrm{Co}(\cdot)$ is due to a standard time-averaging argument [72–74]. Λ is optimal in the sense that no vector $\vec{x}$ outside Λ is feasible for any scheduling policy.

We now formulate a cross-layered congestion control and scheduling problem as follows.

The Cross-Layered Congestion Control and Scheduling Problem:

- Find the end-to-end data rates x_s of the users within the capacity region of the system such that the total system utility is maximized;

$$\max_{x_s} \quad \sum_{s=1}^{S} U_s(x_s) \tag{6.2}$$

$$\text{subject to} \quad [x_s] \in \Lambda. \tag{6.3}$$

- Find the associated scheduling policy that *stabilizes* the system.

The above problem is indeed a cross-layer control problem. The first part of the problem determines the rates with which users inject data into the network (i.e., an end-to-end problem). The second part of the problem determines when and at what rate each link in the network should transmit (i.e., a link-by-link problem). The utility maximization formulation is standard in wireline systems for studying fair rate-allocations [14–16, 61, 106, 107]. Maximizing the total system utility as in (6.2) has been shown to be equivalent to various *fairness* objectives when the utility functions are appropriately chosen [108]. For example, utility functions of the form

$$U_s(x_s) = w_s \log x_s \tag{6.4}$$

correspond to *weighted proportional fairness*, where $w_s, s = 1, ..., S$ are the weights. A more general form of utility function is

$$U_s(x_s) = w_s \frac{x_s^{1-\beta}}{1-\beta}, \beta > 0. \tag{6.5}$$

Maximizing the total utility corresponds to *maximizing the weighted throughput* as $\beta \to 0$, *weighted proportional fairness* as $\beta \to 1$, *minimizing weighted potential delay* as $\beta \to 2$, and *max-min fainess* as $\beta \to \infty$.

The above cross-layer congestion control and scheduling problem is difficult to solve because the capacity region Λ is in a highly complex form. Note that Λ is a convex combination of potentially a large number of points (see (6.1)). In the next section, we will present an optimal solution to the cross-layer congestion control and scheduling problem that do not require exact knowledge of the set Λ.

6.3 Cross-Layer Congestion Control with Perfect Scheduling

We now take a duality approach to solve problem (6.2). We can rewrite the constraint (6.3) in an equivalent form as:

$$\begin{aligned} \text{subject to} \quad & \sum_{s=1}^{S} H_s^l x_s \le r_l, \\ & [r_l] \in \text{Co}(\mathcal{R}). \end{aligned} \tag{6.6}$$

We then associate a Lagrange multiplier q^l for each constraint in (6.6). The Lagrangian is then:

$$\begin{aligned} & L(\vec{x}, \vec{r}, \vec{q}) \\ = & \sum_{s=1}^{S} U_s(x_s) - \sum_{l=1}^{L} q^l \left[\sum_{s=1}^{S} H_s^l x_s - r_l \right] \\ = & \sum_{s=1}^{S} \left[U_s(x_s) - \sum_{l=1}^{L} H_s^l q^l x_s \right] + \sum_{l=1}^{L} q^l r_l. \end{aligned}$$

The objective function of the dual of problem (6.2) is then:

$$\begin{aligned} D(\vec{q}) &= \max_{0 \le x_s \le M_s, s=1,\ldots,S, \vec{r} \in \text{Co}(\mathcal{R})} L(\vec{x}, \vec{r}, \vec{q}) \\ &= \sum_{s=1}^{S} B_s(\vec{q}) + V(\vec{q}), \end{aligned}$$

where

$$B_s(\vec{q}) = \max_{0 \le x_s M_s} \left[U_s(x_s) - \sum_{l=1}^{L} H_s^l q^l x_s \right], \tag{6.7}$$

and

$$V(\vec{q}) \quad = \quad \max_{\vec{r}\in \mathrm{Co}(\mathcal{R})} \sum_{l=1}^{L} q^l r_l. \tag{6.8}$$

Further, because the objective function in (6.8) is a linear function of $\vec{r}$, the optimal point must lie in the set $\mathcal{R}$, i.e.,

$$V(\vec{q}) = \max_{\vec{r}\in\mathcal{R}} \sum_{l=1}^{L} q^l r_l = \max_{\vec{r}=u(\vec{P}),\vec{P}\in\Pi} \sum_{l=1}^{L} q^l r_l. \tag{6.9}$$

The dual approach thus results in an elegant decomposition of the original problem. Given the Lagrange multipliers q^l, the congestion control problem $B_s(\vec{q})$ and the scheduling problem $V(\vec{q})$ are decomposed. Both of these two subproblems react on $\vec{q}$ *independently*. Note that $V(\vec{q})$ also appears as the optimal scheduling policy in [73, 74].

The dual problem of (6.2) is then

$$\min_{\vec{q}\geq 0} D(\vec{q}). \tag{6.10}$$

The dual objective function $D(\vec{q})$ is convex. We can show that its subgradient is given by,

$$\frac{\partial D}{\partial q^l} = -\left(\sum_{s=1}^{S} H_s^l x_s - r_l\right).$$

where $\vec{x} = [x_s]$ and $\vec{r} = [r_l]$ solve (6.7) and (6.9), respectively. We can then use the subgradient method to solve the dual problem [109]. The solution to the optimal cross-layered congestion control problem can be summarized as follows:

The Optimal Cross-Layered Congestion Control Algorithm:

At each iteration t:

- The data rates of the users are determined by

$$x_s(t) = \operatorname*{argmax}_{0\leq x_s\leq M_s} \left[U_s(x_s) - \sum_{l=1}^{L} H_s^l q^l(t) x_s\right]. \tag{6.11}$$

- The schedule is determined by

$$\vec{r}(t) = \operatorname*{argmax}_{\vec{r}\in\mathcal{R}} \sum_{l=1}^{L} q^l(t) r_l = \operatorname*{argmax}_{\vec{r}=u(\vec{P}),\vec{P}\in\Pi} \sum_{l=1}^{L} q^l(t) r_l. \tag{6.12}$$

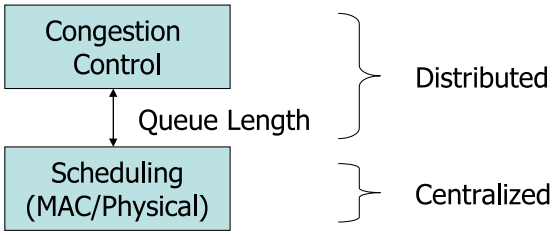


Fig. 6.1. The optimal solution retains a certain degree of modularity with only a loose coupling between the congestion control component and the scheduling component via the queue length.

- The Lagrange multipliers are updated by

$$q^l(t+1) = \left[q^l(t) + \alpha_l \left(\sum_{s=1}^{S} H_s^l x_s(t) - r_l(t) \right) \right]^+ . \tag{6.13}$$

Hence, as illustrated in Fig. 6.1, we observe that the congestion control component and the scheduling component are only *loosely coupled* by the Lagrange multipliers $\vec{q}$. We will see later that $\vec{q}$ is also closely associated with the queue length at each link. Note that the above set of equations make perfect *economic* sense. We can interpret the Lagrange multipler q^l as the price (or implicit cost) of the resource at link l. So $\sum_{l=1}^{L} H_s^l q^l$ can be viewed as the price of the resource on the path of user s. The congestion control component (6.11) simply says that *each user should maximize its own net utility*. Because $U_s(\cdot)$ is strictly concave, if the price increases, the rate of the user will decrease. On the other hand, the scheduling component (6.12) means that *the network should choose the schedule that maximizes the total value of the data transmitted*. Finally, the price is updated based on the principle of balancing the supply with the demand: if the demand ($\sum_{s=1}^{S} H_s^l x_s$) exceeds the supply (r_l), the price increases, and vice versa.

The following proposition shows that this simple set of control rules will converge to an arbitrarily small neighborhood of the optimal rate allocation, provided that the stepsize α_l is sufficiently small[3]. The details of the proof is in Appendix E.1.

[3]If instead the stepsizes are time-varying and diminishing to zero, i.e., they are chosen such that $\alpha_l(t) = h_t \alpha_l^0$, $h_t \to 0$ as $t \to \infty$ and $\sum_{t=1}^{+\infty} h_t = +\infty$, then $\vec{x}(t) \to \vec{x}^*$ as $t \to \infty$.

Proposition 6.3.1 *a) There is no duality gap, i.e., the minimal value of (6.10) coincides with the optimal value of (6.2).*

b) Let Φ be the set of $\vec{q}$ that minimizes $D(\vec{q})$. For any $\vec{q} \in \Phi$, let $\vec{x}$ solve (6.11), then $\vec{x}$ is the unique optimal solution $\vec{x}^*$ of (6.2).

c) Assume that $\alpha_l = h\alpha_l^0$. Define $||\vec{q}||_A = \sum_{l=1}^{L} \frac{(q^l)^2}{\alpha_l^0}$ and let $d(\vec{q}, \Phi) = \min_{\vec{p}\in\Phi} \sqrt{||\vec{q} - \vec{p}||_A}$. For any $\epsilon > 0$, there exists some $h_0 > 0$ such that, for any $h \leq h_0$ and any initial implicit costs $\vec{q}(0)$, there exists a time T_0 such that for all $t \geq T_0$,

$$d(\vec{q}(t), \Phi) < \epsilon \text{ and } ||\vec{x}(t) - \vec{x}^*|| < \epsilon.$$

The optimal cross-layered congestion control algorithm (6.11)-(6.13) not only computes the optimal rate allocation, but also generates the stabilizing scheduling policy by solving (6.12) at each time slot t. In fact, let Q^l denote the queue size at link l. Then Q^l evolves approximately as[4]:

$$Q^l(t+1) \approx \left[Q^l(t) + \left(\sum_{s=1}^{S} H_s^l x_s(t) - r_l(t) \right) \right]^+. \tag{6.14}$$

Comparing (6.14) with (6.13), we can see that $Q^l(t) \approx q^l(t)/\alpha_l$. From here we can infer that $Q^l(t)$ is bounded.

Proposition 6.3.2 *If the stepsizes α_l are sufficiently small, then using the schedules determined by solving (6.12) at each time slot, we have,*

$$\sup_t Q^l(t) < +\infty \text{ for all } l \in \mathcal{L}.$$

We give the proof in Appendix E.2. Combining Propositions 6.3.1 and 6.3.2, we conclude that, by choosing the stepsizes α_l sufficiently small, we can obtain user rate allocation $\vec{x}$ as close to $\vec{x}^*$ as we want, and we can obtain the joint stabilizing scheduling policy at the same time.

[4]Note, (6.14) is an approximation because not all links are active at the same time. Hence, data injected to the network by each user at time t may take several time slots to reach downstream links.

Remark: The duality approach that we used here (and in [29]) shares some similarities to the approach in [98,100,101]. However, there are also some major differences. The network models in [98] and [101] assume a restrictive set of rate-power functions. They either assume that the data rate at each link is a concave function of its *own* power assignment, or assume a special form of rate-power functions that are concave after a change of variables. In this chapter, we impose no such restrictions. Further, a consequence of the assumption in [101] is that, at their optimal solution, all links will be transmitting at the same time. In the more general network model of this chapter, it usually requires different sets of links to transmit at different time in order to achieve optimality. In [100], the authors propose a column generation approach for solving (6.2). This approach appears to be more suitable for *offline* computation as it requires solving a sequence of approximate problems to (6.2), each of which requires an iterative solution by itself. In contrast, in this chapter we are more interested in solutions suitable for *on-line* implementation. Finally, these previous works have not addressed the joint stabilizing scheduling policy as we did in Proposition 6.3.2.

6.4 The Impact of Imperfect Scheduling on Cross-Layered Congestion Control: The Static Case

Note that in our optimal cross-layer congestion control algorithm in Section 6.3, the congestion control component (6.11) is fully distributed: each user only needs to know the implicit cost at the links that it traverses. Also distributed is the implicit cost update (6.13) at each link, which only requires the information about the offered load and capacity at the local link. However, the scheduling component (6.12) in the optimal solution is a global optimization problem that is generally difficult to solve. Although the power constraint Π in (6.12) may sometimes be a convex set, the function $u(\cdot)$ is typically neither concave nor convex (e.g., consider the case when $u(\cdot)$ is determined by the Shannon bound on capacity). Hence, the scheduling problem

is usually not a convex programming problem, and is usually very difficult to solve. In some cases, this optimization problem does not even have a polynomial-time solution. Therefore, solving (6.12) exactly at every time slot is too time-consuming. The complexity of the scheduling component is the main difficulty in implementing our cross-layer solution.

In large multihop wireless networks, it is usually preferred that the control algorithm be implemented in a distributed fashion. However, the scheduling subproblem (6.12) is difficult to solve even by a centralized algorithm, let alone in a distributed fashion. Again, the complexity of the scheduling component is the main obstacle in developing fully distributed solutions.

Therefore, in order to develop simple and potentially distributed solutions for the cross-layer congestion control and scheduling problem, we have to consider the likely scenario with *imperfect scheduling*, i.e., the scheduling component does not compute the optimal schedule in (6.12) at each time. Next, we will study how imperfect scheduling impacts the optimality of cross-layered congestion control. Our objective is to find some imperfect scheduling policies that are easy to implement and that, when properly designed with congestion control, result in good overall performance.

In this chapter, we will particularly be interested in the following class of imperfect scheduling policies:

Imperfect Scheduling Policy S_γ:

Fix $\gamma \in (0, 1]$. At each time slot t, compute a schedule $\vec{r}(t) \in \mathcal{R}$ that satisfies:

$$\sum_{l=1}^{L} r_l(t) q^l(t) \geq \gamma \max_{\vec{r} \in \mathcal{R}} \sum_{l=1}^{L} r_l q^l(t). \tag{6.15}$$

With an imperfect scheduling policy S_γ, the dynamics of cross-layered congestion control are summarized by the following set of equations:

$$x_s(t) = \operatorname*{argmax}_{0 \leq x_s \leq M_s} \left[U_s(x_s) - \sum_{l=1}^{L} H_s^l q^l(t) x_s \right], \tag{6.16}$$

$$\vec{r}(t)^T \vec{q}(t) \geq \gamma \max_{\vec{r} \in \mathcal{R}} \vec{r}^{\,T} \vec{q}(t), \quad \vec{r}(t) \in \mathrm{Co}(\mathcal{R}), \tag{6.17}$$

$$q^l(t+1) = \left[q^l(t) + \alpha_l \left(\sum_{s=1}^{S} H_s^l x_s(t) - r_l(t) \right) \right]^+ . \tag{6.18}$$

The parameter γ in (6.15) can be viewed as a tuning parameter indicating the degree of precision of the imperfect schedule. The complexity of finding a schedule $\vec{r}(t)$ satisfying (6.15) usually decreases as γ is reduced. When $\gamma = 1$, the dynamics (6.16)-(6.18) reduce to the case with perfect scheduling (as in Section 6.3).

Loose Coupling Property Revisited: We now get to the heart of the issue of modularity versus efficiency in cross-layer design. Our solution (6.11)-(6.13) is optimal when all components run perfectly. In particular, it is optimal only when optimal schedules are computed at each time. When an imperfect scheduler S_γ is used by the scheduling component, we no longer have a handle on the efficiency of the rate allocation. The rate allocation computed by the dynamics (6.16)-(6.18) will likely be sub-optimal, but we do not know how bad it can be. In fact, due to the interplay between the congestion control component and the scheduling component, there is a possibility that the entire system may get stuck into some local sub-optimal regions where the performance is very poor. For our proposed cross-layer control solution to be successful, we need to be able to quantify the impact of these imperfect scheduling policies, and ensure that the performance of the system degrades gracefully. Recall that we would like to maintain a loose coupling between the layers: i.e., even when one layer changes, the system should still result in satisfactory performance.

We next investigate the impact of imperfect scheduling policies on cross-layer congestion control when the number of users in the system is fixed (i.e., the static setting). Let $\vec{x}^{\,*,0}$ denote the solution to the original optimal cross-layered congestion control and scheduling problem (6.2). The solution to the following problem turns out to be a good reference point for studying the dynamics (6.16)-(6.18) when $\gamma < 1$:

The γ-Reduced Problem:

$$\begin{aligned} \max_{0 \le x_s \le M_s} \quad & \sum_{s=1}^{S} U_s(x_s) \\ \text{subject to} \quad & \vec{x} \in \gamma\Lambda. \end{aligned} \tag{6.19}$$

Let $\vec{x}^{*,\gamma}$ denote the solution to the γ-reduced problem. The choice of $\gamma\Lambda$ in the constraint of the γ-reduced problem is motivated by the following proposition, which shows that an imperfect scheduling policy S_γ at most reduces the capacity region by a factor of γ. The proof is given in Appendix E.3.

Proposition 6.4.1 *If the user rates $\vec{x}$ lie strictly inside $\gamma\Lambda$, then any imperfect scheduling policy S_γ can stabilize the system.*

Motivated by Proposition 6.4.1, we would expect that the rate allocation computed by the dynamics (6.16)-(6.18) will be "no worse than" $\vec{x}^{*,\gamma}$. However, this assertion is not quite true. As we will see soon, the interaction between cross-layered congestion control and imperfect scheduling is much more complicated. As the data rates of the users are reacting to the same implicit costs as the scheduling component is, there is a possibility that the system gets stuck into local sub-optimal areas. We will construct examples where, for a subset of the users, their data rates determined by the dynamics (6.16)-(6.18) can be much smaller than the corresponding rate allocation computed by the γ-reduced problem. Nonetheless, we will be able to show certain weak but desirable results on the fairness and convergence properties of cross-layer congestion control with imperfect scheduling.

6.4.1 Dominance

Our first hope is that the rate allocation computed by the dynamics (6.16)-(6.18) will dominate $\vec{x}^{*,\gamma}$, which is the rate allocation computed by the γ-reduced problem (6.19). (Note: a vector $[x_1, ..., x_S]$ dominates another vector $[y_1, ..., y_S]$ if $x_i \geq y_i$ for all $i = 1, ..., S$.) However, we will soon see that this is not true in general. We begin our analysis by studying whether $\vec{x}^{*,0}$, the rate allocation computed by the optimal cross-layer solution, will dominate $\vec{x}^{*,\gamma}$. Note that the perfect scheduling policy is automatically an S_γ policy for any $\gamma < 1$. Hence, if such dominance does not hold, then the hope that an arbitrary S_γ policy will compute a rate allocation dominating $\vec{x}^{*,\gamma}$ will not hold either. The following proposition shows that $\vec{x}^{*,0}$ will dominate

$\vec{x}^{\,*,\gamma}$ if the utility function is logarithmic. (Recall that logarithmic utility functions are of the form

$$U_s(x_s) = w_s \log x_s \text{ for all user } s,$$

where w_s is the weight for user s. In this case, the rate allocation computed by the original problem (6.2) is *weighted proportionally fair* [108].)

Proposition 6.4.2 *Assume that the utility function is logarithmic. Let $\vec{x}^{\,*,0}$ be the solution to the original problem (6.2). Then the solution to the γ-reduced problem is*

$$\vec{x}^{\,*,\gamma} = \gamma \vec{x}^{\,*,0}.$$

Proof In the γ-reduced problem (6.19), do a change of variables $\vec{x}' = \vec{x}/\gamma$. Using the fact that

$$U_s(x_s) = w_s \log x'_s + w_s \log \gamma,$$

one can show that the γ-reduced problem becomes equivalent to the original problem (6.2). Hence, $\vec{x}^{\,*,\gamma} = \gamma \vec{x}^{\,*,0}$. ■

However, as shown in the following example, if the utility function is not logarithmic, dominance will not hold in general.

Example 1: Consider the following wireline network (note that a wireline network can be viewed as a special case of our network model where the capacity of each link is fixed). There are two links, whose capacities are 2 and 7, respectively. There are three users. The first user uses both links, the second user uses only the first link, and the third user uses only the second link. Their utility functions are

$$\begin{aligned} U_1(x) &= \log x + 6x \\ U_2(x) &= \log x \\ U_3(x) &= 36 \log x. \end{aligned}$$

The γ-reduced problem is then

$$\begin{aligned} \max_{x_1,x_2,x_3 \geq 0} \quad & (\log x_1 + 6x_1) + \log x_2 + 36 \log x_3 \\ \text{subject to} \quad & x_1 + x_2 \leq 2\gamma \\ & x_1 + x_3 \leq 7\gamma. \end{aligned}$$

When $\gamma = 1$, the solution is $\vec{x}^{\,*,0} = [1 \quad 1 \quad 6]^T$. When $\gamma = 0.95$, the solution becomes $\vec{x}^{\,*,\gamma} = [0.8551 \quad 1.0449 \quad 5.7949]^T$. Note that the rate of the second user *increases* as γ is reduced. This example shows that $\vec{x}^{\,*,0}$ does not dominate $\vec{x}^{\,*,\gamma}$ in general.

6.4.2 A Weak Fairness Property

For the rest of the chapter, we will focus on logarithmic utility functions, although most of the results that follow can also be extended to utility functions of other forms (as in (6.5)). Note that even though $\vec{x}^{\,*,0}$ dominates $\vec{x}^{\,*,\gamma}$ when the utility function is logarithmic (as shown in Proposition 6.4.2), it does not imply that the rate allocation computed by the cross-layered congestion control algorithm with an *arbitrary* imperfect scheduling policy S_γ will dominate $\vec{x}^{\,*,\gamma}$.

In the following proposition, we characterize the likely rate allocation under imperfect scheduling *provided that the dynamics (6.16)-(6.18) converges*. The proof is given in Appendix E.4.

Proposition 6.4.3 *Assume that the utility function is logarithmic (i.e., of the form in (6.4)). If the dynamics (6.16)-(6.18) converges, i.e.,* $\vec{x}(t) \to \vec{x}^{\,*,I}$ *and* $\vec{q}(t) \to \vec{q}_I^{\,*}$ *as* $t \to \infty$, *then*

$$\vec{x}^{\,*,I} \in \Lambda \text{ and } \sum_{s=1}^{S} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}} \leq \sum_{s=1}^{S} w_s. \tag{6.20}$$

Proposition 6.4.3 can be generalized to other forms of utility functions (as in (6.5)). This result can be viewed as a *weak fairness property*. It shows that, if the dynamics (6.16)-(6.18) converge, the rate allocation of the users will lie in a strip defined by (6.20) (see Fig. 6.2). Hence, even though the rates of the users may not dominate $\vec{x}^{\,*,\gamma}$, they are unlikely to be too unfair compared to $\vec{x}^{\,*,\gamma}$. In particular, if $w_s = 1$ for all s, then by (6.20), $x_s^{*,I}$ will be no smaller than $x_s^{*,\gamma}/S$. On the negative side, the rates of some users can still be substantially smaller than their rates computed by the γ-reduced problem, which indicates that cross-layered congestion control with imperfect scheduling may indeed get stuck into local sub-optimal regions.

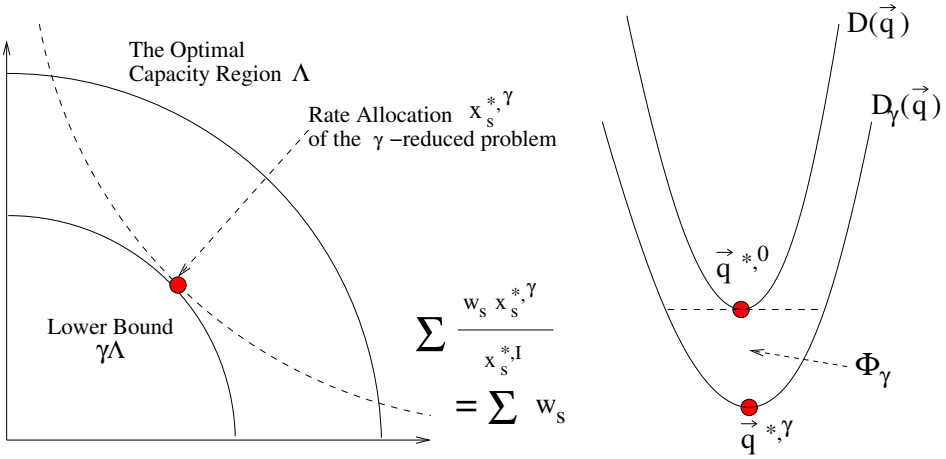


Fig. 6.2. The weak fairness property (left) and the set Φ_γ (right).

6.4.3 Convergence

We next study the question whether the dynamics (6.16)-(6.18) converge in the first place. Using a duality approach analogous to that in Section 6.3, we can define the dual of the γ-reduced problem as

$$D_\gamma(\vec{q}) \;\; = \;\; \sum_{s=1}^{S} B_s(\vec{q}) + \gamma V(\vec{q}),$$

where $B_s(\vec{q})$ and $V(\vec{q})$ are still defined as in (6.7) and (6.8), respectively. Note that both $D(\vec{q})$ and $D_\gamma(\vec{q})$ are convex functions and $D(\vec{q}) \geq D_\gamma(\vec{q})$.

Let $\vec{q}^{\,*,0}$ denote a minimizer of $D(\vec{q})$ and $\vec{q}^{\,*,\gamma}$ denote a minimizer of $D_\gamma(\vec{q})$. Further, let

$$\Phi_\gamma = \{\vec{q} : D_\gamma(\vec{q}) \leq D(\vec{q}^{\,*,0})\}.$$

Proposition 6.4.4 *Assume that $\alpha_l = h\alpha_l^0$. Let $||\vec{q}||_A = \sum_{l=1}^{L} \frac{(q^l)^2}{\alpha_l^0}$. For any $\epsilon > 0$, there exists some $h_0 > 0$ such that, for any $h \leq h_0$ and any initial implicit costs $\vec{q}(0)$, there exists a time T_0 such that for all $t \geq T_0$,*

$$\sqrt{||\vec{q}(t) - \vec{q}^{\,*,0}||_A} < \max_{\vec{p} \in \Phi_\gamma} \sqrt{||\vec{p} - \vec{q}^{\,*,0}||_A} + \epsilon.$$

The proof is provided in Appendix E.5. Proposition 6.4.4 shows that, if the stepsizes α_l are sufficiently small, the dynamics (6.16)-(6.18) will eventually enter a neighborhood of the set Φ_γ. Note that both $\vec{q}^{\,*,0}$ and $\vec{q}^{\,*,\gamma}$ belong to the set Φ_γ (see Fig. 6.2). Hence, in a weak sense, the dynamics of the system are moving in the right direction. However, in general the set Φ_γ is quite large and does not provide much further insights on the eventual rate allocation. We next present two examples illustrating the possible behaviors of the dynamics.

Example 2:

We will first show that, for any vectors $\vec{q}_I^{\,*}$ and $\vec{x}^{\,*,I}$ that satisfy

$$x_s^{*,I} = \frac{w_s}{\sum_{l=1}^{L} H_s^l q_I^{l,*}} < M_s \text{ for all } s, \; \vec{x}^{\,*,I} \in \Lambda, \text{ and}$$

$$\sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} > \gamma \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \sum_{l=1}^{L} q_I^{l,*} r_l, \tag{6.21}$$

there exists an imperfect scheduling policy S_γ such that the dynamics (6.16)-(6.18) converge to $\vec{q}_I^{\,*}$ and $\vec{x}^{\,*,I}$. Note that the above set of conditions implies (6.20). In fact, since

$$\left[\sum_{s=1}^{S} H_s^l x_s^{*,\gamma}, l \in \mathcal{L}\right] \in \gamma \mathrm{Co}(\mathcal{R}),$$

we have,

$$\begin{aligned}
\sum_{s=1}^{S} w_s &= \sum_{s=1}^{S} x_s^{*,I} \sum_{l=1}^{L} H_s^l q_I^{l,*} = \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} \\
&\geq \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} = \sum_{s=1}^{S} x_s^{*,\gamma} \sum_{l=1}^{L} H_s^l q_I^{l,*} \\
&= \sum_{s=1}^{S} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}}.
\end{aligned}$$

We now show how a suitable imperfect scheduling policy S_γ can be constructed. It is easy to verify that $\vec{x}^{\,*,I}$ is the solution to the following optimization problem and $\vec{q}_I^{\,*}$ is the corresponding Lagrange multipliers.

$$\begin{aligned}
\max_{\vec{x} \geq 0} \quad & \sum_{s=1}^{S} w_s \log x_s \\
\text{subject to} \quad & \sum_{s=1}^{S} H_s^l x_s \leq \sum_{s=1}^{S} H_s^l x_s^{*,I}.
\end{aligned} \tag{6.22}$$

Hence, if we let

$$r_l(t) = \sum_{s=1}^{S} H_s^l x_s^{*,I} \text{ for all } l \text{ and all } t, \tag{6.23}$$

then, using a standard gradient descent argument for the dual problem of (6.22), we can show that the dynamics (6.16)-(6.18) will converge to $\vec{q}_I^{\,*}$ and $\vec{x}^{\,*,I}$ as $t \to \infty$. It remains to be verified whether the schedule in (6.23) belongs to the class of imperfect scheduling policies S_γ. To see this, note that if we pick the initial implicit cost vector $\vec{q}(0)$ to be sufficiently close to $\vec{q}_I^{\,*}$, then $\vec{q}(t) \approx \vec{q}_I^{\,*}$ for all t. Hence,

$$\begin{aligned}
\sum_{l=1}^{L} q^l(t) r_l(t) &\approx \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} \\
> \ \gamma \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \sum_{l=1}^{L} q_I^{l,*} r_l &\approx \gamma \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \sum_{l=1}^{L} q^l(t) r_l,
\end{aligned}$$

i.e., the schedule in (6.23) indeed belongs to S_γ if the initial implicit cost vector $\vec{q}(0)$ is sufficiently close to $\vec{q}_I^*$.

Example 3:

We next give another example in which the dynamics (6.16)-(6.18) *never converge to any point*. Consider the following simple wireline network with two users, each of which uses one different link. The capacity is c for both links. The solution to the γ-reduced problem is simply $x_1^{*,\gamma} = x_2^{*,\gamma} = \gamma c$. Assume that the vectors $\vec{q}_I^*$ and $\vec{x}^{\,*,I}$ satisfy the conditions in (6.21) of Example 2. At any time t, define

$$\Delta(t) = \vec{q}(t) - \vec{q}_I^*.$$

Let ϵ be a small positive number. We now use the following scheduling policy:

$$\vec{r}(t) = [r_1 \quad r_2]^T = \begin{cases} \begin{bmatrix} x_1^{*,I} & x_2^{*,I} \end{bmatrix}^T - \epsilon \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \frac{\Delta(t)}{||\Delta(t)||}, & \text{if } \Delta(t) \le \epsilon, \\ \begin{bmatrix} x_1^{*,I} & x_2^{*,I} \end{bmatrix}^T - \epsilon \frac{\Delta(t)}{||\Delta(t)||}, & \text{if } \epsilon \le \Delta(t) \le 2\epsilon, \\ c, & \text{otherwise.} \end{cases}$$

With this choice of the schedule $\vec{r}(t)$, the update of the implicit cost $\vec{q}(t)$ will be around a circle when $||\vec{q}(t) - \vec{q}_I^*|| \le \epsilon$, and it will be towards $\vec{q}_I^*$ when $\epsilon < ||\vec{q}(t) - \vec{q}_I^*|| \le 2\epsilon$ (see Fig. 6.3). Provided that the initial $\vec{q}(0)$ satisfies $||\vec{q}(0) - \vec{q}_I^*|| \le 2\epsilon$ and the stepsizes are sufficiently small, the dynamics (6.16)-(6.18) will eventually follow the circle $||\vec{q}(t) - \vec{q}_I^*|| = \epsilon$, and hence will never converge. We can verify as in Example 2 that the schedule $\vec{r}(t)$ does belong to the class S_γ when the stepsizes and ϵ are sufficiently small.

To conclude this section, we have studied the impact of imperfect scheduling on the dynamics of cross-layered congestion control *when the number of users in the system is fixed*. We have presented several examples that illustrate the difficulty in characterizing the dynamics precisely. We have shown that the system may not even converge in the first place, or, it may converge to any rate allocation within a fairly large set that does not possess any desirable dominance property. These examples indicate that the interaction between cross-layered congestion control and imperfect

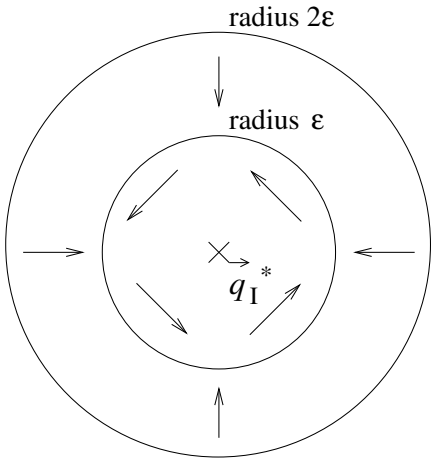


Fig. 6.3. The direction of the update of the implicit costs

scheduling are quite complicated, and the system may indeed get stuck into local sub-optimal regions. Nonetheless, we do show two desirable, but weak, results on the fairness and convergence properties of the system. In Proposition 6.4.4, we are able to show that the dynamics (6.16)-(6.18) appear to move in the right direction globally. In Proposition 6.4.3, we show that those local sub-optimal regions are probably "not too bad." In the next section, we will turn to the case when users dynamically arrive and depart the network, and surprisingly, we will be able to show far stronger results on the performance of the system there.

6.5 Stability Region of Cross-Layered Congestion Control with Imperfect Scheduling

In this section, we turn to the case when the number of users in the system is itself a stochastic process. We will study how imperfect scheduling impacts the *stability region* of the system employing cross-layer congestion control. Here, by *stability*, we mean that the number of users in the system and the queue lengths at all links in the network remain finite. The *stability region* of the system is the set of offered loads under which the system is stable. Previous works for wireline networks have shown that, by allocating data rates to the users according to some fairness criteria, the *largest possible* stability region can be achieved [108, 110–112]. This result is important as it tells us that fairness is not just an *aesthetic* property, but it actually has a strong global *performance* implication, i.e., in achieving the *largest possible* stability region. In this section, we will show that similar but stronger results can be shown for our cross-layered congestion control scheme with imperfect scheduling.

To be precise, instead of using the notation s for user s, we now use s to denote a class of users with the same utility function and the same path. We assume that users of class s arrive according to a Poisson process with rate λ_s and that each user brings with it a file for transfer whose size is exponentially distributed with mean $1/\mu_s$. The load brought by users of class s is then $\rho_s = \lambda_s/\mu_s$. Let $\vec{\rho} = [\rho_1, ..., \rho_S]$.

Let $n_s(t)$ denote the number of users of class s that are in the system at time t, and let $\vec{n}(t) = [n_1(t), ..., n_S(t)]$. We assume that the rate allocations for users of the same class are identical. Let $x_s(t)$ denote the rate of each user of class s at time t. In the rate assignment model that follows, the evolution of $\vec{n}(t)$ will be governed by a Markov process. Its transition rates are given by:

$$\begin{aligned} n_s(t) &\to n_s(t)+1, && \text{with rate } \lambda_s, \\ n_s(t) &\to n_s(t)-1, && \text{with rate } \mu_s x_s(t) n_s(t) \\ & && \quad \text{if } n_s(t) > 0. \end{aligned}$$

As in [113], We say that the above system is *stable* if

$$\limsup_{t\to\infty} \frac{1}{t} \int_0^t \mathbf{I}_{\{\sum_{s=1}^{S} n_s(t) + \sum_{l=1}^{L} q^l(t) > M\}} dt \to 0,$$

as $M \to \infty$. This means that the fraction of time that the amount of "unfinished work" in the system exceeds a certain level M can be made arbitrary small as $M \to \infty$. The *stability region* Θ of the system under a given congestion control and scheduling policy is the set of offered loads $\vec{\rho}$ such that the system is stable.

We next describe the rate assignment and implicit cost update policy. We assume that time is divided into slots of length T, and the schedules and implicit costs are only updated at the end of each time slot. However, users may arrive and depart in the middle of a time slot. Let $\vec{q}(kT)$ denote the implicit cost at time slot k. The data rates of the users are determined by the current implicit costs as in (6.11). For simplicity, we assume that the utility function is logarithmic (the result can be readily generalized to utility functions of other forms in (6.5)). Further, let M_s denote the maximum data rate for users of class s. The rate of each user of class s is then given by

$$x_s(t) = x_s(kT) = \min \left\{ \frac{w_s}{\sum_{l=1}^{L} H_s^l q^l(kT)}, M_s \right\} \tag{6.24}$$

for $kT \leq t < (k+1)T$. The schedule $\vec{r}(kT)$ at time slot k is computed according to an imperfect scheduling policy S_γ based on the current implicit cost $\vec{q}(kT)$. Finally, at the end of each time slot, the implicit costs are updated as

$$q^l((k+1)T) = \left[q^l(kT) + \alpha_l \left(\sum_{s=1}^{S} H_s^l \int_{kT}^{(k+1)T} n_s(t) x_s(kT) dt - r_l(kT)T \right) \right]^+ . \tag{6.25}$$

The following proposition shows that, using the above cross-layered congestion control algorithm with imperfect scheduling policy S_γ, the stability region of the system is no smaller than $\gamma\Lambda$.

Proposition 6.5.1 *If*

$$\max_{l \in \mathcal{L}} \alpha_l \leq \frac{1}{T\bar{S}\bar{L}} \min_s \frac{w_s}{4\rho_s M_s}, \tag{6.26}$$

where $\bar{S} = \max_{l \in \mathcal{L}} \sum_{s=1}^{S} H_s^l$ *is the maximum number of classes using any link, and* $\bar{L} = \max_s \sum_{l=1}^{L} H_s^l$ *is the maximum number of links used by any class, then for any offered load* $\vec{\rho}$ *that resides strictly inside* $\gamma\Lambda$*, the system described by the Markov process* $[\vec{n}(kT), \vec{q}(kT)]$ *is stable.*

Several remarks are in order: Firstly, Proposition 6.5.1 shows that, when imperfect schedules are used, the stability region of the system employing cross-layer congestion control is no worse than the capacity region shown in Proposition 6.4.1 (and used by the γ-reduced problem). This result is interesting (and somewhat surprising) given the fact that, when the number of users in the system is fixed, the dynamics of cross-layered congestion control with imperfect scheduling can form loops or get stuck into local sub-optimal regions. *Nonetheless, Proposition 6.5.1 shows that these potential local sub-optimums are inconsequential when the arrivals and departures of the users are taken into account.*

Secondly, we do not need the rates of any users to converge. Previous results on the stability region of congestion control typically adopt a *time-scale separation assumption* [108, 110–112], which assumes that the rate allocation $\vec{x}(t)$ *perfectly solves*

(6.2) at each time instant t. Such an approach is of little value for the model in this chapter because the dynamics (6.16)-(6.18) with imperfect scheduling do not even converge in the first place! Further, the time-scale separation assumption is rarely realistic in practice: as the number of users in the system is constantly changing, the rate allocation may never have the time to converge. In Proposition 6.5.1, we establish the stability region of the system without requiring such a time-scale separation assumption. This result is of independent value. For the special case when $\gamma = 1$, it can be viewed as a stronger version of previous results in the literature (including those for wireline networks, e.g., Theorem 1 in [108]).

Finally, a simple stepsize rule is provided in (6.26). Note that when the number of users in the system is fixed, we typically require the stepsizes to be driven to zero for convergence to occur (see Proposition 6.3.1). However, in (6.26) the stepsizes can be chosen bounded away from zero. In fact, as the set $\gamma\Lambda$ is bounded, the stepsizes can be chosen independently of the offered load. The simplicity in the stepsize rule is another benefit we obtain by studying the dynamic arrivals and departures of the users.

6.5.1 The Main Idea of the Proof of Proposition 6.5.1

We now sketch the main idea of the proof for Proposition 6.5.1 so that the reader can gain some insight on the dynamics of the system. Define the following Lyapunov function,

$$\mathcal{V}(\vec{n}, \vec{q}) = V_n(\vec{n}) + V_q(\vec{q}),$$

where $V_n(\vec{n}) = \sum_{s=1}^{S} \frac{w_s n_s^2}{2\lambda_s}$, and $V_q(\vec{q}) = \sum_{l=1}^{L} \frac{(q^l)^2}{2\alpha_l}$. We shall show below that $\mathcal{V}(\vec{n}, \vec{q})$ has a negative drift. As a crude first-order approximation, assume that users arrive and depart only at the end of each time slot. Thus, $n_s(t) = n_s(kT)$ during the k-th time slot. We can show that (see Appendix E.6 for the details),

$$\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)]$$

$$\leq \quad T\sum_{s=1}^{S}\left[\frac{w_s}{x_s(kT)}\right][\rho_s - n_s(kT)x_s(kT)] + E_1(k),$$

where $E_1(k)$ is an error term that is roughly on the order of $|\rho_s - n_s(kT)x_s(kT)|$. Since the rate allocation is determined by (6.24), we have (ignoring the maximum data rate M_s),

$$\begin{aligned}
&\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq \quad & T\sum_{s=1}^{S}\left[\sum_{l=1}^{L} H_s^l q^l(kT)\right][\rho_s - n_s(kT)x_s(kT)] \\
& +E_1(k).
\end{aligned} \tag{6.27}$$

We can also show that

$$\begin{aligned}
&\mathbf{E}[V_q(\vec{q}((k+1)T) - V_q(\vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq \quad & T\sum_{l=1}^{L} q^l(kT)\left[\sum_{s=1}^{S} H_s^l n_s(kT)x_s(kT) - r_l(kT)\right] \\
& +E_2(k),
\end{aligned} \tag{6.28}$$

where $E_2(k)$ is an error term that is roughly on the order of $\left[\sum_{s=1}^{S} H_s^l n_s(kT)x_s(kT) - r_l(kT)\right]^2$. Hence, by adding (6.27) and (6.28), and by changing the order of the summation, we have

$$\begin{aligned}
&\mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) \\
&\qquad -\mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq \quad & T\sum_{l=1}^{L} q^l(kT)\left[\sum_{s=1}^{S} H_s^l \rho_s - r_l(kT)\right] \\
& +E_1(k) + E_2(k).
\end{aligned} \tag{6.29}$$

By assumption, $\vec{\rho}$ lies strictly inside $\gamma\Lambda$. Hence, there exists some $\epsilon > 0$ such that

$$[(1+\epsilon)\sum_{s=1}^{S} H_s^l \rho_s] \in \gamma \mathrm{Co}(\mathcal{R}).$$

By the definition of the imperfect scheduling policy S_γ,

$$\sum_{l=1}^{L} q^l(kT) r_l(kT) \geq (1+\epsilon)\sum_{l=1}^{L} q^l(kT)\sum_{s=1}^{S} H_s^l \rho_s.$$

Substituting into (6.29), we have,

$$\begin{aligned} & \mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) \\ & \qquad -\mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\ \leq \; & -T\epsilon \sum_{l=1}^{L} q^l(kT) \sum_{s=1}^{S} H_s^l \rho_s + E_1(k) + E_2(k). \end{aligned} \tag{6.30}$$

This shows that $\mathcal{V}(\cdot, \cdot)$ would drift towards zero when $||\vec{q}(kT)||$ is large and when the error terms $E_1(k)$ and $E_2(k)$ are bounded. We would then apply Theorem 2 of [113] to establish the stability of the system.

To complete the proof, however, we have to address several difficulties:

- In order to apply Theorem 2 of [113], a stronger negative drift is required. Instead of (6.30), we need,

$$\begin{aligned} & \mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) \\ & \qquad -\mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\ \leq \; & -\epsilon'(||\vec{n}(kT)|| + ||\vec{q}(kT)||) + E_0 \end{aligned}$$

  for some positive constants ϵ' and E_0.

- Further, in order to apply Theorem 2 of [113], the error terms $E_1(k)$ and $E_2(k)$ have to be bounded, which is not true in (6.30) since they both can become large as $n_s(kT)$ increases.

- Finally, users could arrive and depart at any time (not only at the end of a time slot).

The complete proof that addresses these difficulties is given in Appendix E.6. There, we use a slightly modified Lyapunov function, and we use the stepsize condition (6.26) and the assumption on the maximum data rate M_s to obtain upper bounds on the error terms $E_1(k)$ and $E_2(k)$. For details, please refer to Appendix E.6.

We now give two examples showing how efficient cross-layer congestion control schemes can be constructed by applying Proposition 6.5.1 to different network settings.

6.5.2 The Node Exclusive Interference Model

Proposition 6.5.1 is most useful when an imperfect schedule that satisfies (6.15) can be easily computed for some reasonable value of γ. This is the case under the following node exclusive interference model.

The Node Exclusive Interference Model:

- The data rate of each link is fixed at c_l.
- Each node can only send to or receive from one other node at any time.

This interference model has been used in earlier studies of congestion control in multihop wireless networks [114, 115]. Under this model, the *perfect* schedule (according to (6.12)) at each time slot corresponds to the **Maximum Weighted Matching (MWM)**, where the weight of each link is $q^l c_l$. (A *matching* is a subset of the links such that no two links share the same node. The *weight* of a matching is the total weight over all links belonging to the matching. A *maximum-weighted-matching (MWM)* is the matching with the maximum weight.) An $O(N^3)$-complexity algorithm for MWM can be found in [116], where N is the number of nodes. On the other hand, the following much simpler **Greedy Maximal Matching (GMM)** algorithm can be used to compute an imperfect schedule with $\gamma = 1/2$. Start from an empty schedule. From all possible links $l \in \mathcal{L}$, pick the link with the largest $q^l c_l$. Add this link to the schedule. Remove all links that are incident with either the sending node or the receiving node of link l. Pick the link with the largest $q^l c_l$ from the *remaining* links, and add to the schedule. Continue until there are no links left. The GMM algorithm has only $O(L \log L)$-complexity (where L is the number of links), and is much easier to implement than MWM. Using the technique in Theorem 10 of [117], we can show that the weight of the schedule computed by the GMM algorithm is at least 1/2 of the weight of the *maximum-weighted-matching*. According to Proposition 6.5.1, the stability region will be at least $\Lambda/2$ using our cross-layered congestion control scheme with the GMM scheduling policy.

For the node-exclusive interference model, a *layered approach* to congestion control is also possible, which considers *separately* the dynamics of congestion control and scheduling [114, 115]. In the layered approach, the network designer will choose a *rate region* within the capacity region, which has a simpler set of constraints similar to that of wireline networks, and compute the rate allocation within this simpler rate region [114, 115, 118]. For the node-exclusive interference model, such a simple rate region can be found. It has been shown that the optimal capacity region Λ in the node-exclusive interference model is bounded by $\frac{2}{3}\Psi_0 \subseteq \Lambda \subseteq \Psi_0$, where

$$\Psi_0 = \left\{ \vec{x} \left| \sum_{l:b(l)=i \text{ or } e(l)=i} \frac{1}{c_l} \sum_{s=1}^{S} H_s^l x_s \leq 1 \text{ for all } i \right. \right\}.$$

and $b(l)$ and $e(l)$ are the sending node and the receiving node, respectively, of link l. The layered approach then chooses the lower bound $\frac{2}{3}\Psi_0$ as the *rate region* for computing the rate allocation [114,115]. On the other hand, when an imperfect GMM scheduling policy is used, the capacity region can be reduced by half in the worst case (according to Proposition 6.4.1). Hence, the layered approach then needs to use $\Psi_0/3 (\subseteq \Lambda/2)$ as the *rate region*. Note that for the layered approach with GMM scheduling, $\Psi_0/3$ is an *upper bound* for its stability region, which is smaller than the *lower bound* of the stability region of the corresponding cross-layered approach (which is $\Lambda/2$ according to Proposition 6.5.1). Hence, due to its *conservative* nature, the layered approach always suffers from *worst case* inefficiencies. In Section 6.7, we will use simulations to show that our cross-layered congestion control scheme can in practice substantially outperform the layered approach.

6.5.3 General Interference Models

Under general interference models, it may still be time-consuming to compute a schedule that satisfies (6.15) for a given value of γ. We now use Proposition 6.5.1 to develop a scheduling policy that can cut down the *frequency* of such computation, and hence effectively reduce the computation overhead. This idea is motivated by

the observation that implicit costs, being updated by (6.18), cannot change abruptly. Hence, there is a high chance that a schedule computed earlier can be *reused* in subsequent time-slots. To see this, assume that we know a schedule $\vec{r}^{\,0}$ that satisfies (6.15) for an inefficiency factor $\gamma_0 > \gamma$ when the implicit cost vector is $\vec{q}^{\,0}$, i.e.,

$$\sum_{l=1}^{L} r_l^0 q_0^l \geq \gamma_0 \max_{\vec{r}\in\mathcal{R}} \sum_{l=1}^{L} r_l q_0^l. \tag{6.31}$$

Let the implicit cost vector at the current time slot be $\vec{q}$, and let $\vec{r}^{\,*}$ denote the corresponding (but unknown) perfect schedule. We can normalize $\vec{q}^{\,0}$ and $\vec{q}$ to be of unit length since the corresponding schedules will remain the same. We have,

$$\begin{aligned}\sum_{l=1}^{L} q^l r_l^* &= \sum_{l=1}^{L}(q^l - q_0^l) r_l^* + \sum_{l=1}^{L} q_0^l r_l^* \\ &\leq \sum_{l=1}^{L}[q^l - q_0^l]^+ r_l^{\max} + \frac{\sum_{l=1}^{L} q_0^l r_l^0}{\gamma_0},\end{aligned}$$

where $r_l^{\max}$ is the maximum rate of link l. Hence, if

$$\sum_{l=1}^{L} q^l r_l^0 \geq \gamma \left\{ \sum_{l=1}^{L}[q^l - q_0^l]^+ r_l^{\max} + \frac{\sum_{l=1}^{L} q_0^l r_l^0}{\gamma_0} \right\},$$

we can still use $\vec{r}^{\,0}$ as the imperfect schedule for $\vec{q}$. This approach is even more powerful when the network can remember multiple schedules from the past. Let $\vec{r}^{\,k} = [r_1^k, ..., r_L^k]$ and $\vec{q}^{\,k} = [q_1^1, ..., q_k^L]$, $k = 1, ..., K$. Assume that the schedules $\vec{r}^{\,1}, \vec{r}^{\,2}, ..., \vec{r}^{\,K}$ correspond to $\vec{q}^{\,1}, \vec{q}^{\,2}, ..., \vec{q}^{\,K}$, respectively, and each pair satisfies (6.31). Then, as long as

$$\begin{aligned}&\max_{k=1,..,K} \sum_{l=1}^{L} q^l r_l^k \\ \geq\ &\min_{k=1,...,K} \gamma \left\{ \sum_{l=1}^{L}[q^l - q_k^l]^+ r_l^{\max} + \frac{\sum_{l=1}^{L} q_k^l r_l^k}{\gamma_0} \right\},\end{aligned} \tag{6.32}$$

we do not need to compute a new schedule. Instead, we can use the schedule that maximizes the left hand side of (6.32). By Proposition 6.5.1, the stability region of the system using the above scheduling policy is no smaller than $\gamma\Lambda$. In Section 6.7, we will use simulations to show that such a simple policy can perform very well in practice.

6.6 A Fully Distributed Cross-Layered Rate Control and Scheduling Algorithm

Proposition 6.5.1 opens a new avenue for studying cross-layer design for congestion control in multihop wireless networks. Instead of restricting our attention to the rate allocation at each snapshot of the system (as we did in Section 6.4 where the results tend to be weaker), we can now study the entire time horizon by focusing on the stability region of such a cross-layer-designed system. Motivated by Proposition 6.5.1, we now present a *fully distributed* cross-layered congestion control and scheduling algorithm for the node-exclusive interference model in Section 6.5.2. (In contrast, the GMM algorithm in Section 6.5.2 still requires centralized implementation.) This new algorithm can be shown to achieve a stability region no smaller than $\Lambda/2$.

The new algorithm uses **Maximal Matching (MM)** to compute the schedule at each time [117, 119, 120]. A *maximal matching* is a matching such that no more links can be added without violating the node-exclusive interference constraint. To be precise, let q_{ij} denote the implicit cost at link (i, j). (For convenience, in this section we will index a link by a node pair (i, j).) A maximal matching $\mathcal{M}$ is a subset of $\mathcal{L}$ such that $q_{ij} \geq 1$ for all $(i, j) \in \mathcal{M}$, and, for each $(i, j) \in \mathcal{L}$, one of the following holds:

$$\begin{aligned} &q_{ij} < 1, \text{ or} \\ &(i, k) \in \mathcal{M} \text{ for some link } (i, k) \in \mathcal{L}, \text{ or} \\ &(k, i) \in \mathcal{M} \text{ for some link } (k, i) \in \mathcal{L}, \text{ or} \end{aligned} \tag{6.33}$$

$$(j,h) \in \mathcal{M} \text{ for some link } (j,h) \in \mathcal{L}, \text{ or}$$
$$(h,j) \in \mathcal{M} \text{ for some link } (h,j) \in \mathcal{L}.$$

Note that a maximal matching can be computed in a distributed fashion as follows. When a link (i,j) is added to the matching, we say that both node i and node j are *matched*. For each node i, if it has already been matched, no further action is required. Otherwise, node i scans its neighboring nodes. If there exists a neighboring node j such that node j has not been matched, node i sends a matching request to node j. It is possible that a matching request conflicts with other matching requests. In this case, the nodes involved in the conflict can use some randomization and local coordination to pick any non-conflicting subset of the matching requests. For those nodes whose matching requests are declined, they can repeat the above procedure until every node in the network is either matched or has no neighbors that are not matched.

Let

$$Q_i = \sum_{j:(i,j)\in\mathcal{L}} q_{ij} + \sum_{j:(j,i)\in\mathcal{L}} q_{ji}$$

denote the total cost of the links that either start from, or end at node i. Our new cross-layered congestion control and scheduling algorithm then proceeds as follows.

The Distributed Cross-Layered Congestion Control Algorithm:

At each time slot $[kT,(k+1)T)$:

- A maximal matching $\mathcal{M}(kT)$ is computed based on the implicit costs $\vec{q}(kT)$.
- The data rate of each user of class s is determined by

$$x_s(t) = x_s(kT) = \max\left\{\frac{w_s}{2\sum_{(i,j)\in\mathcal{L}} H_s^{ij}\frac{Q_i(kT)+Q_j(kT)}{c_{ij}}}, M_s\right\} \tag{6.34}$$

  where c_{ij} is the capacity of link (i,j), and H_s^{ij} is defined as H_s^l, i.e., $H_s^{ij}=1$, if users of class s use link (i,j); and $H_s^{ij}=0$, otherwise.

- The implicit costs are updated by:

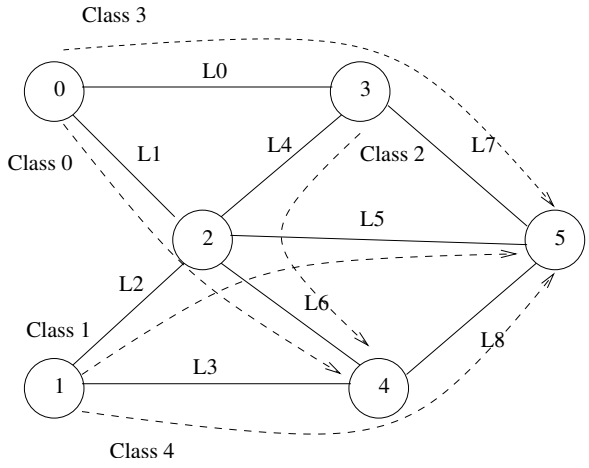
$$q_{ij}((k+1)T) = \Big[q_{ij}(kT) + \alpha\left(\sum_{s=1}^{S} H_s^{ij}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ij}}dt\ -T\mathbf{I}_{\{(i,j)\in\mathcal{M}(kT)\}}\right)\Big]^+. \tag{6.35}$$

This new cross-layered congestion control and scheduling algorithm is similar to the algorithms of Section 6.4 and 6.5 in many aspects:

- A user reacts to congestion by reducing its data rate when the implicit costs along its path increase.
- The implicit cost at each link (i, j) is updated based on the difference between the offered load and the schedule of the link.

However, there is a critical difference. When the maximal matching is computed, we do not care about the precise value of the implicit costs (see (6.33), where the maximal matching only depends on whether the implicit costs q_{ij} are larger than a chosen threshold). Hence, the maximal matching typically does not satisfy the requirement of the imperfect scheduling policy S_γ, and Proposition 6.5.1 does not apply either. Further, the congestion control part (6.34) is also different from that in the earlier sections. *It has been chosen specifically for the maximal matching scheduling policy.* Nonetheless, using similar techniques as in Section 6.5, we can show the following result on the stability region of the system. The details are given in Appendix E.7.

Proposition 6.6.1 *If the stepsize α is sufficiently small, then for any offered load $\vec{\rho}$ that resides strictly inside $\Lambda/2$, the system with the above distributed cross-layered congestion control algorithm is stable.*



Node	Position
0	(0.0, 0.0)
1	(0.0, 2.0)
2	(1.0, 1.0)
3	(2.2, 0.0)
4	(2.5, 2.0)
5	(3.5, 1.0)

Fig. 6.4. The network topology

6.7 Numerical Results

We now use simulations to verify the results in this chapter. We use the network in Fig. 6.4. There are 5 classes of users, whose paths are shown in Fig. 6.4. Their utility functions are all given by $U_s(x_s) = \log x_s$. We first use the following interference model. The path loss $G(i,j)$ from a node i to a node j is given by $G(i,j) = d_{ij}^{-4}$ where d_{ij} is the distance from node i to node j (the positions of the nodes are also given in Fig. 6.4). We assume that the data rate r_{ij} at link $(i,j) \in \mathcal{L}$ is proportional to the SIR, i.e.,

$$r_{ij} \;=\; W \frac{G(i,j)P_{ij}}{N_0 + \sum_{(k,h)\in\mathcal{L},(k,h)\neq(i,j)} G(k,j)P_{kh}},$$

where N_0 is the background noise and W is the bandwidth of the system. This assumption is suitable for CDMA systems with a moderate processing gain [74]. Each node i has a power constraint $P_{i,\max}$, i.e., the power allocation must satisfy $\sum_{j:(i,j)\in\mathcal{L}} P_{ij} \leq P_{i,\max}$ for all i.

We first simulate the case when there is one user for each class. The top figure in Fig. 6.5 shows the evolution of the data rates for all five users when the network computes the perfect schedule according to (6.12) at every time slot. We have chosen

$W = 10$, $N_0 = 1.0$, $P_{i,\max} = 1.0$ for all node i and $\alpha_l = 0.1$ for all link l. Note that the scheduling subproblem (6.12) for this interference model is a complex non-convex global optimization problem. In [29], we have given an $O(2^N)$ algorithm for solving the perfect schedule, where N is the number of nodes. Executing such an algorithm at every time-slot is extremely time-consuming.

We then simulate the imperfect scheduling policy outlined in Section 6.5.3 for general interference models. Such an imperfect scheduling policy attempts to *reuse* schedules that have already been computed in the past. In our simulation, we have chosen $\gamma_0 = 1.0$ in (6.31), i.e., each of these past schedules are perfect schedules. The computational complexity could have been further reduced if we had chosen $\gamma_0 < 1$. However, we leave this for future work. Instead, in this section we focus on how the imperfect scheduling policy can *reduce the number of times that new perfect schedules have to be computed*. The system that we simulate can store at most 10 past schedules. If there are already 10 past schedules and a new perfect schedule is computed, the new schedule will replace the old one that has the smallest weighted-sum $\sum_{l=1}^{L} q^l r_l$. In the bottom figure of Fig. 6.5, we show the evolution of the data rates when $\gamma = 0.5$. Note that the rate allocation eventually converges to values close to that with perfect scheduling. We also record the number of times that perfect schedules are computed. When $\gamma = 0.5$, perfect schedules are computed in only 7 iterations among the entire 2000 iterations of the simulation, and most of these perfect schedules are computed at the initial stage of the simulation. We have simulated other values of γ and find similar results. In fact, by just reducing γ from 1.0 to 0.9, the number of times that perfect schedules have to be computed is reduced to 34 (over 2000 iterations of simulation). These results indicate that our cross-layered congestion control scheme with the imperfect scheduling policy in Section 6.5.3 can substantially reduce the computation overhead and still maintain good performance.

We then simulate the case when there are dynamic arrivals and departures of the users as in Section 6.5. Users of each class arrive to the network according to

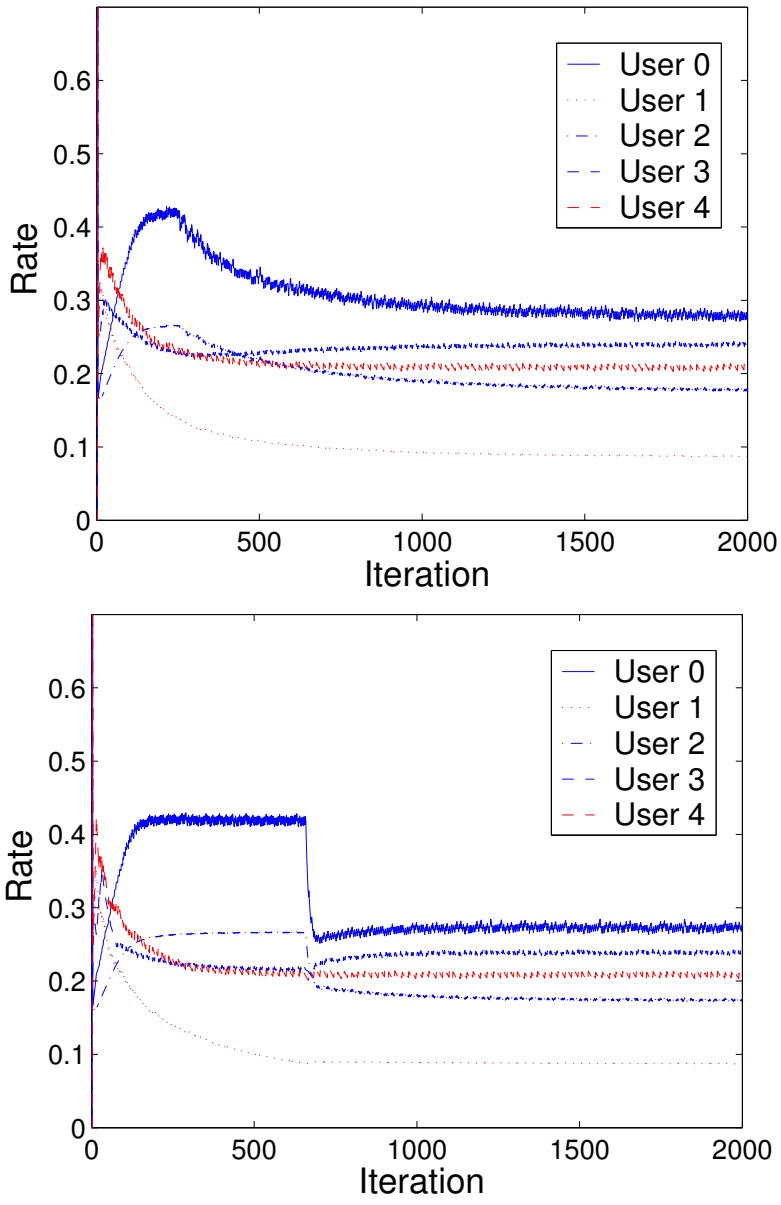


Fig. 6.5. The evolution of the data rates for all users with perfect scheduling (top) and with imperfect scheduling (bottom, $\gamma = 0.5$).

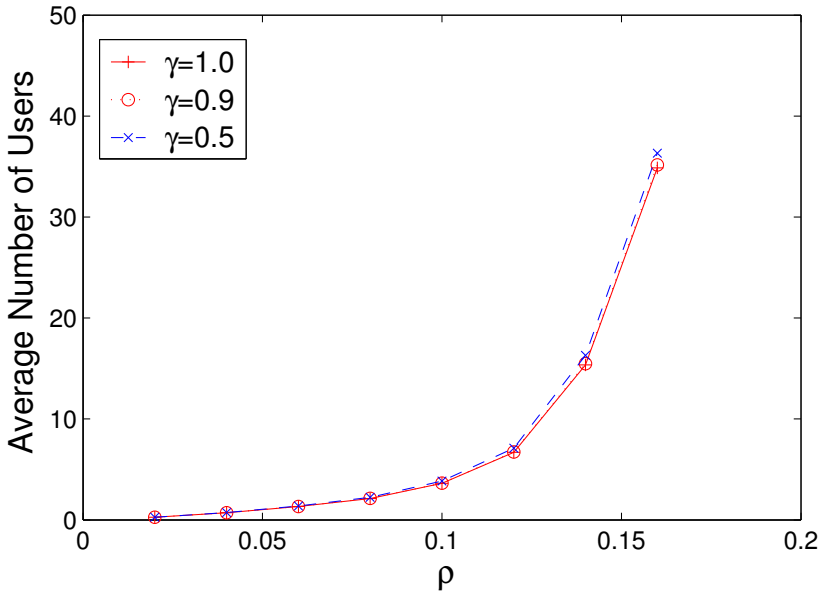


Fig. 6.6. The average number of users in the system versus load.

a Poisson process with rate λ. Each user brings with it a file to transfer whose size is exponentially distributed with mean $1/\mu = 100$ unit. We vary the arrival rate λ (and hence the load $\rho = \lambda/\mu$) and record in Fig. 6.6 the average number of users in the system at any time for different choice of γ. Given γ, the average number of users in the system will increase to infinity as the offered load ρ approaches a certain limit. This limit can then be viewed as the capacity of the system. From Fig. 6.6, we observe that the capacity of the system is not significantly affected when γ is reduced from 1.0 to 0.5. On the other hand, the number of time-slots that new perfect schedules have to be computed is reduced to less than 1% of the total number of time-slots when $\gamma = 0.9$, and to less than 0.05% when $\gamma = 0.5$. These results confirm again the effectiveness of our cross-layered congestion control scheme with the imperfect scheduling policy in Section 6.5.3, in reducing the computation overhead and achieving good overall performance.

We next turn to the node-exclusive interference model in Section 6.5.2, where we can draw a comparison with the layered approach to congestion control [114, 115].

We still use the network topology in Fig. 6.4. The capacity of each link is now fixed at 10 units. We only report the result for the case when there are dynamic arrivals and departures of the users. Fig. 6.7 demonstrates the average number of users in the system versus load with different congestion control and scheduling schemes. We label each curve with the congestion control scheme (we use "Joint" to denote the cross-layered congestion control scheme and use "Layered" to denote the layered approach in [115]), followed by the scheduling policy. (Note that the curve for the cross-layered congestion control scheme with GMM scheduling, labeled as "Joint-GMM," in fact overlaps with the curve for the optimal cross-layered congestion control scheme with perfect MWM scheduling, which is the right most curve labeled as "Joint-MWM.") From Fig. 6.7, we observe that, regardless of the scheduling policy used (either MWM, GMM, or MM), the layered approach always performs *much poorer* than the corresponding cross-layered approach. The performance gap widens even more when an imperfect scheduling policy (such as GMM) is used. In particular, the fully distributed joint congestion control and scheduling algorithm in Section 6.6 (with *imperfect* maximal matching scheduling, labeled "Joint-MM"), actually performs even better than the layered approach with the *perfect* (and more complex) MWM scheduling (labeled "Layered-MWM"). These results demonstrate that the conservative nature of the layered approach indeed hurts the overall performance of the system, and an appropriately designed cross-layered congestion control scheme can perform very well in practice even with imperfect scheduling.

6.8 Conclusion

In this chapter, we study issues in cross-layer design of multihop wireless networks. We propose a loose-coupling approach to cross-layer design, which achieves both modularity and efficiency. We demonstrate how a cross-layer solution with such a loose-coupling property can be developed for the cross-layer congestion control and scheduling problem. Our cross-layer solution only requires minimal amount of inter-

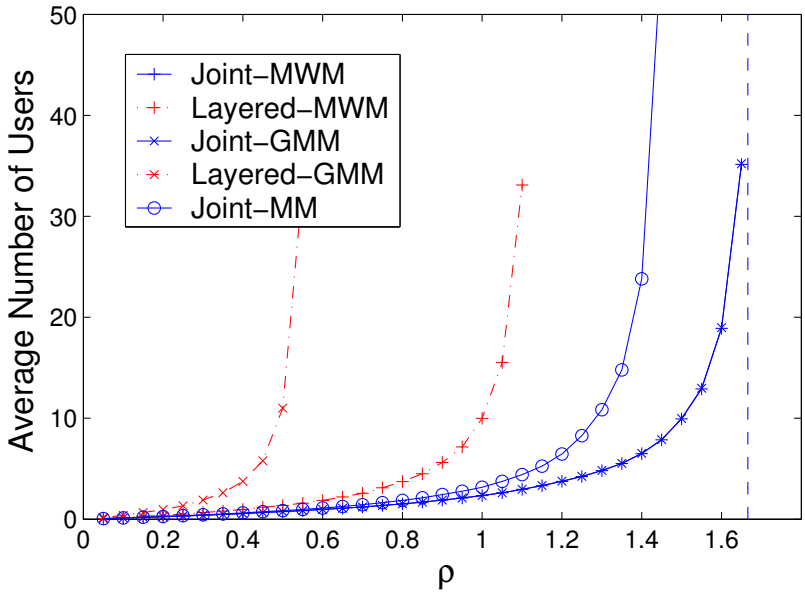


Fig. 6.7. The average number of users in the system versus load: the node-exclusive interference model

action among the protocol layers, and is robust to imperfect decisions made at the scheduling component. We also demonstrate how fully distributed solutions may be built by taking advantage of the loose-coupling property.

These results constitute an important step towards designing fully distributed cross-layered congestion control schemes for multihop wireless networks. Several directions for future work are possible. For example, Proposition 6.5.1 may be combined with a clustering scheme to design distributed cross-layered congestion-control solutions for large networks. We can also use similar techniques as in [29] to combine cross-layered congestion control with multipath routing. Our main result (Proposition 6.5.1) can also be extended to the case with random fading. It would also be important to study the impact of feedback delays, to address the effect of node mobility, and to extend our results to hybrid wireless-wireline networks.

7. SUMMARY

7.1 Summary of Contributions

In this dissertation, we have studied how to simplify the network dynamics and control in large communication networks. We have taken two orthogonal approaches to this problem. Next, we summarize our main contributions through each approach.

7.1.1 Exploiting the Largeness of the System to Simplify Control

In the first approach, we seek simplicity through exploiting the largeness of the network. We first studied the pricing-based network control problem and the Quality-of-Service routing problem in large-capacity wire-line networks. We showed that simple static control policies can approach the performance of the optimal (but complex) dynamic control policy when the capacity of the system is large. Further, the near-optimal static control parameters can be determined from a simple non-linear programming problem that depends only on the average statistics of the network. We have established this result under very general network settings, first for a non-Markovian network with fixed topology and routing, then in the case when the network supports dynamic routing, and also in the case where the topology of the network becomes increasingly complex as its capacity grows.

These results indicates that significant simplicity in control can be achieved in large-capacity networks. Compared with the optimal dynamic scheme, the static scheme has several desirable features. The static schemes are much easier to obtain because of their simple structures. They are also much easier to execute since they do not require the collection of instantaneous load information. Hence, they introduce less computation and communication overhead, and they are less sensitive to

feedback delay. These advantages make the static scheme an attractive alternative for controlling large networks.

For large networks, it is also imperative that the control algorithm can be implemented on-line in a distributive fashion. In the second part of the dissertation, we developed distributed control algorithms based on these static control policies. Our distributed algorithm can adaptively track the optimal static control parameters based on on-line measurements. We rigorously established the convergence of the distributed algorithm to the optimal control parameters, without requiring an unrealistic two-level convergence structure typically in standard results. We also provided guidelines on how to choose the parameters of the algorithm to ensure efficient control. This algorithm can then be applied to a number of networking problems, such as multi-path flow control, QoS routing, and network pricing. For example, when applied to the QoS routing problem in large-capacity networks, our proposed algorithm not only achieves near-optimal routing performance, but also substantially alleviates the computation and communication overhead of QoS routing without sacrificing the performance.

In the third part of the dissertation, we turned to wireless networks and we investigated the fundamental tradeoff between the capacity and the delay in large mobile wireless networks. By exploiting the largeness in the number of nodes in these networks, we obtained simple scaling laws that determine the optimal achievable capacity given delay constraints. We have developed a systematic methodology both for finding the optimal capacity-delay tradeoff and for designing the capacity-achieving scheme. Our methodology can be applied to a number of mobility models, such as the *i.i.d.* mobility model, the random way-point mobility model, and the Brownian motion mobility model. In each case, we have identified the limitations of existing works, obtained sharper results under more general settings, and provided new insights on the fundamental capacity-delay tradeoffs. In particular, under the *i.i.d.* mobility model, our study allows us to develop a scheme that can exploit mobility and achieve a provably larger per-node capacity than that of the static

networks *even with delay that does not grow with the number of nodes.* This is the first such result of its kind in the literature.

7.1.2 Designing Appropriate System Architecture to Simplify Control

In the second approach, we seek simplicity by designing an appropriate control architecture such that complex interactions within the system can be structured into layers that are only weakly dependent on each other through a judiciously chosen set of control parameters. In particular, we investigated the cross-layer congestion control and scheduling problem in multi-hop wireless networks. We have developed a loose-coupling approach to this problem. By loose-coupling, we mean that the cross-layer solution only requires a minimal amount of interaction between the layers, and is robust to imperfect decisions at each layer. We showed that the optimal solution to the cross-layer congestion control and scheduling problem can be decomposed into a congestion control component and a scheduling component. Both components act independently on the queue lengths of the system. They are then coupled by the update of the queue length at each link. Further, if one replaces the scheduling component by an imperfect scheduling policy that only computes suboptimal schedules at each time, we can still quantify the impact on the overall system performance fairly easily for a large class of imperfect scheduling policies. These results allow us to use imperfect, but simpler and potentially distributed, algorithms for cross-layer control of large wireless networks. Under the node-exclusive interference model, we have successfully developed the first fully distributed cross-layer congestion control and scheduling algorithm in the literature.

7.2 Suggestions for Future Research

In this dissertation, we have presented a number of scenarios where simplicity arises in large communication networks. These scenarios are certainly not the only

ones where simplicity can be obtained. Next, we give a few examples that illustrate the multitude of possible future research directions.

7.2.1 Dynamics of TCP in Large-Capacity Networks

In Chapters 2 and 3, we have mainly studied the simplification of network dynamics in large-capacity networks where the data rates of the users are pre-chosen. Even though we have considered users with elastic data rates in Section 2.4 of Chapter 2, there we still assume that the amount of time that the user will remain in the system is independent of the data rate. While these assumptions are suitable for video/audio streaming traffic with fixed or adaptive data rates, they are not for typical data traffic in the Internet, such as file transfers and HTTP traffic. For these types of data traffic in the current Internet, the data rates of the flows are regulated by TCP, and a flow will terminate as soon as a fixed amount of data is transferred through the network. Hence, the amount of time that the flow remains in the system is a function of its data rate.

Therefore, it remains an interesting problem to study the dynamics of TCP in large-capacity networks. There are two paradigms we can consider. In the first paradigm, we can consider a network with a large but fixed number of TCP users. This has been the model taken in [121–123]. The authors there show that the dynamics of the system will converge to that of a fluid system when the number of users is large. A key assumption in these works is that the buffer size at the bottle neck router also grows proportionally to the number of users. Hence, the per-user buffer size remains fixed. Lately, it has been shown in [36] that, as long as the capacity and the number of users in the system is large, one can in fact reduce the per-user buffer size aggressively, while still maintaining high link-utilization and low packet losses. Note that more aggressive buffer provisioning could translate into huge savings in the cost of high-speed buffers in backbone routers in the Internet. These result illustrates the possibility of exploiting new dynamics of TCP in a network with

large capacity. The work in [36] has assumed that the system can be modeled as a doubly-stochastic processes. It would be interesting to study whether the results hold under more general assumptions.

In the second paradigm, the number of TCP users in the system can also change according to certain stochastic processes. Previous works in [108, 124] have shown that some of the conclusions drawn from such a dynamic settings can be very different from those draw from a static setting (where the number of users is fixed). Thus, it would be interesting to study whether the results in [36] also hold for networks with a dynamic set of TCP users.

7.2.2 Scaling Laws in Wireless Networks

The techniques in Chapter 5 can be very powerful in obtaining first-order insights regarding various quantities of interest in wireless networks. Other than capacity and delay, another metric that is of great concern in many wireless systems (including sensor networks) is energy. Problems of interest include: What is the most energy-efficient way that the network should operate given capacity and delay constraints? What is the difference between the most energy-efficient *centralized* solution compared with the most energy-efficient *distributed* solution? Using the techniques in Chapter 5, one may find simple answers to these questions, which will greatly enhance our understanding for the design of energy-efficient wireless networks.

7.2.3 TCP for Wireless Networks

In Chapter 6, we have provided a framework for cross-layer congestion control and scheduling in multihop wireless networks. Note that the congestion control component in our solution is not TCP yet. Nonetheless, there is a striking similarity between our congestion control component and the modern understanding of TCP based on the optimization framework [14–16, 41, 61, 106]. In the latter group of work, a TCP flow is also viewed as carrying out a local net-utility maximization.

Thus, the framework we provided in Chapter 6 offers great hope for the design of practical congestion control protocols in wireless networks. We may be able to use the insights obtained there to make minimal changes to TCP and make it work efficiently in wireless networks.

7.2.4 Combining Scaling Laws with Loose-Coupling

In Chapter 6, we have developed a fully distributed cross-layer control solution for the node-exclusive interference model. For more general interference models, fully distributed solution remains an open problem. The main difficulty still lies upon the complexity of the scheduling component. It would be interesting to study whether we can use the insights from scaling laws (as in Chapter 5) to further simplify the scheduling component, and eventually develop efficient distributed solutions for more general interference models.

LIST OF REFERENCES

LIST OF REFERENCES

[1] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Mass., 2001.

[2] D. D. Botvich and N. G. Duffield. Large Deviations, the Shape of the Loss Curve, and Economies of Scale in Large Multiplexers. *Queueing Systems*, 20:293–320, 1995.

[3] C. Courcoubetis and R. Weber. Buffer Overflow Asymptotics for a Buffer Handling Many Traffic Sources. *Journal of Applied Probability*, 33:886–903, 1996.

[4] N. Likhanov and R. R. Mazumdar. Cell-Loss Asymptotics in Buffers Fed with a Large Number of Independent Stationary Sources. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, 1998.

[5] M. Montgomery and G. De Veciana. On the Relevance of Time Scales in Performance Oriented Traffic Characterization. In *Proceedings of IEEE INFOCOM*, pages 513–520, San Francisco, CA, 1996.

[6] R. G. Addie and M. Zukerman. An Approximation for Performance Evaluation of Stationary Single Server Queues. *IEEE Transactions on Communications*, 42(12):3150–3160, Dec. 1994.

[7] J. Choe and N. B. Shroff. A Central Limit Theorem Based Approach for Analyzing Queue Behavior in High-Speed Networks. *IEEE/ACM Transactions on Networking*, 6(5):659–671, Oct. 1998.

[8] J. Choe and N. B. Shroff. Use of the Supremum Distribution of Gaussian Processes in Queueing Analysis with Long-Range Dependence and Self-Similarity. *Comm. Statist. Stochastic Models*, 16(2), Feb. 2000.

[9] J. Choe and N. B. Shroff. Queueing Analysis of High-Speed Multiplexers including Long-Range Dependent Arrival Processes. In *Proceedings of IEEE INFOCOM*, New York, NY, Mar. 1999.

[10] D. Wischik. The Output of a Switch, or, Effective Bandwidths for Networks. *Queueing Systems*, 32:383–396, 1999.

[11] D. Y. Eun and N. B. Shroff. Simplification of Network Analysis in Large-Bandwidth Systems. In *Proceedings of IEEE INFOCOM*, San Francisco, April 2003.

[12] J. K. Mackie-Mason and H. R. Varian. Pricing the Internet. In *Public Access to the Internet (B. Kahin and J. Keller, Eds.)*, pages 269–314. The MIT Press, Cambridge, MA, 1995.

[13] A. A. Lazar and N. Semret. The Progressive Second Price Auction Mechanism for Network Resource Sharing. In *8th International Symposium on Dynamic Games and Applications*, pages 359–365, Maastricht, the Netherlands, July 1998.

[14] F. P. Kelly, A. Maulloo, and D. Tan. Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[15] S. H. Low and D. E. Lapsley. Optimization Flow Control–I: Basic Algorithm and Convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999.

[16] H. Yaiche, R. Mazumdar, and C. Rosenberg. A Game Theoretic Framework for Bandwidth Allocation and Pricing in Broadband Networks. *IEEE/ACM Transactions on Networking*, 8(5):667–678, Oct. 2000.

[17] I. Ch. Paschalidis and J. N. Tsitsiklis. Congestion-Dependent Pricing of Network Services. *IEEE/ACM Transactions on Networking*, 8(2):171–184, April 2000.

[18] X. Lin and N. B. Shroff. Pricing-Based Control of Large Networks. In *Evolutionary Trends of the Internet, Proceedings of Tyrrhenian International Workshop on Digital Communications (IWDC 2001)*, pages 212–231, Taormina, Italy, September 2001.

[19] X. Lin and N. B. Shroff. Simplification of Network Dynamics in Large Systems. In *Tenth International Workshop on Quality of Service(IWQoS 2002)*, Miami Beach, FL, May 2002.

[20] X. Lin and N. B. Shroff. Simplification of Network Dynamics in Large Systems. *to appear in IEEE/ACM Transactions on Networking, October 2005, also available at http://min.ecn.purdue.edu/∼linx/.*

[21] X. Lin and N. B. Shroff. An Optimization Based Approach for Quality of Service Routing in High-Bandwidth Networks. In *Proceedings of IEEE INFOCOM*, Hong Kong, China, March 2004.

[22] X. Lin and N. B. Shroff. The Multi-Path Utility Maximization Problem. In *41st Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, October 2003.

[23] M. Grossglauser and D. Tse. Mobility Increases the Capacity of Ad Hoc Wireless Networks. *IEEE/ACM Transactions on Networking*, 10(4), August 2002.

[24] X. Lin and N. B. Shroff. The Fundamental Capacity-Delay Tradeoff in Large Mobile Ad Hoc Networks. In *Third Annual Mediterranean Ad Hoc Networking Workshop*, Bodrum, Turkey, June 2004.

[25] X. Lin and N. B. Shroff. Towards Achieving the Maximum Capacity in Large Mobile Wireless Networks Under Delay Constraints. *Invited for publication in the Journal of Communications and Networks, Special Issue on Mobile Ad Hoc Wireless Networks*, 2004.

[26] X. Lin and N. B. Shroff. On the Fundamental Relationship Between the Achievable Capacity and Delay in Mobile Wireless Networks. *Advances in Pervasive Computing and Networking. B. K. Szymanski and B. Yener (eds), Kluwer Academic Publishers*, 2004.

[27] X. Lin, G. Sharma, R. R. Mazumdar, and N. B. Shroff. Degenerate Delay-Capacity Trade-offs in Ad Hoc Networks with Brownian Mobility. *submitted to the Joint Special Issue of IEEE Transactions on Information Theory and IEEE/ACM Transactions on Networking, Special Issue on Networking and Information Theory, available at http://min.ecn.purdue.edu/∼linx/*, 2005.

[28] M. Schwartz. *Telecommunication Networks*. Addison-Wesley, Menlo Park, CA, 1987.

[29] X. Lin and N. B. Shroff. Joint Rate Control and Scheduling in Multihop Wireless Networks. In *Proceedings of the IEEE Conference on Decision and Control*, Paradise Island, Bahamas, December 2004.

[30] X. Lin and N. B. Shroff. The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Multihop Wireless Networks. In *Proceedings of IEEE INFOCOM*, Miama, FL, March 2005.

[31] I. Ch. Paschalidis and Y. Liu. Pricing in Multiservice Loss Networks: Static Pricing, Asymptotic Optimality, and Demand Substitution Effects. *IEEE/ACM Transactions on Networking*, 10(3):425–438, June 2002.

[32] R. J. McEliece and K. N. Sivarajan. Maximizing Marginal Revenue in Generalized Blocking Service Networks. In *Proc. 30th Annual Allerton Conference on Communication, Control, and Computing*, pages 455–464, 1992.

[33] P. B. Key. Optimal Control and Trunk Reservation in Loss Networks. *Probability in the Engineering and Informational Sciences*, 4:203–242, 1990.

[34] T. Basar and R. Srikant. Revenue-Maximizing Pricing and Capacity Expansion in a Many-Users Regime. In *Proceedings of IEEE INFOCOM*, pages 1556–1563, New York, New York, June 2002.

[35] P. Tinnakornsrisuphap and A. M. Makowski. TCP Traffic Modeling via Limit Theorems. *Technical Report, Institute for Systems Research, University of Maryland*, 2002.

[36] D. Y. Eun and X. Wang. Stationary Behavior of TCP/AQM with Many Flows Under Aggressive Packet Marking. In *Proceedings of IEEE ICC 2005*, May 2005.

[37] S. Jin and A. Bestavros. Small-World Internet Topologies: Possible Causes and Implications on Scalability of End-System Multicast. *Technical Report BUCS-TR-2002-004, Boston University, Boston, MA*, 2002.

[38] A. Barabasi and R. Albert. Emergence of Scaling in Random Networks. *Science*, 286:509–512, 1999.

[39] F. P. Kelly. Loss Networks. *Annals of Applied Probability*, 1:319–378, 1991.

[40] A. Medina, A. Lakhina, I. Matta, and J. Byers. Brite: Universal Topology Generation From a User's Perspective. In *Proceedings of the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS '01)*, Cincinnati, Ohio, August 2001.

[41] S. Kunniyur and R. Srikant. End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks. In *Proceedings of IEEE INFOCOM*, Tel-Aviv, Israel, March 2000.

[42] R. Rejaie, M. Handley, and D. Estrin. RAP: an End-to-End Rate-Based Congestion Control Mechanism for Realtime Streams in the Internet. In *Proceedings of IEEE INFOCOM*, New York, NY, March 1999.

[43] B. Girod. Psychovisual Aspects of Image Communications. *Signal Processing*, 28(3):239–251, September 1992.

[44] S. Floyd, M. Handley, and J. Padhye. A Comparison of Equation-Based and AIMD Congestion Control. In *http://www.aciri.org/tfrc/aimd.ps*, 2000.

[45] D. Bansal and H. Balakrishnan. Binomial Congestion Control Algorithms. In *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, April 2001.

[46] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis. Call Admission Control and Routing in Integrated Service Networks Using Neuro-Dynamic Programming. *IEEE Journal on Selected Areas in Communications*, 18(2):197–208, February 2000.

[47] P. Whittle. Approximation in Large-Scale Circuit-Switched Networks. *Prob. Engrn. Inf. Sci.*, 2:279–291, 1988.

[48] S. Chen and K. Nahrstedt. An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions. *IEEE Network, Special Issue on Transmission and Distribution of Digital Video*, 12(6):64–79, November/December 1998.

[49] G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi. Quality of Service Based Routing: A Performance Perspective. In *Proceedings of ACM SIGCOMM*, pages 17–28, Vancouver, Canada, September 1998.

[50] Q. Ma and P. Steenkiste. On Path Selection for Traffic with Bandwidth Guarantees. In *IEEE ICNP*, 1997.

[51] A. Shaikh, J. Rexford, and K. Shin. Efficient Precomputation of Quality-of-Service Routes. In *Proceedings of Workshop on Network and Operating Systems Support for Digital Audio and Video*, Cambridge, United Kingdom, July 1998.

[52] A. Shaikh, J. Rexford, and K. Shin. Evaluating the Impact of Stale Link State on Quality-of-Service Routing. *IEEE/ACM Transactions on Networking*, 9(2):162–176, April 2001.

[53] F. P. Kelly. Routing in Circuit Switched Networks: Optimization, Shadow Prices and Decentralization. *Advances in Applied Probability*, 20:112–144, 1988.

[54] F. P. Kelly. Routing and Capacity Allocation in Networks with Trunk Reservation. *Mathematics of Operations Research*, 15(4):771–793, 1990.

[55] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. ATM Network Design and Optimization: a Multirate Loss Network Framework. *IEEE/ACM Transactions on Networking*, 4(4):531–543, August 1996.

[56] S. Nelakuditi, Z.-L. Zhang, R. P. Tsang, and D. H. C. Du. Adaptive Proportional Routing: a Localized QoS Routing Approach. *IEEE/ACM Transactions on Networking*, 10(6):790–804, December 2002.

[57] S. Nelakuditi, S. Varadarajan, and Z.-L. Zhang. On Localized Control in QoS Routing. *IEEE Transactions on Automatic Control*, 47(6):1026–1032, June 2002.

[58] W. H. Wang, M. Palaniswami, and S. H. Low. Optimal Flow Control and Routing in Multi-Path Networks. *Performance Evaluation*, 52(2-3):119–132, April 2003.

[59] K. Kar, S. Sarkar, and L. Tassiulas. Optimization Based Rate Control for Multipath Sessions. *Technical Report No. 2001-1, Institute for Systems Research, University of Maryland*, 2001.

[60] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley. Overlay TCP for Multi-Path Routing and Congestion Control. *Presented at the ENS-INRIA ARC-TCP Workshop, Paris, France, Nov. 2003 and at the IMA Workshop on Measurements and Modeling of the Internet, January 2004, avilalable at http://tesla.csl.uiuc.edu/∼srikant/pub.html*.

[61] S. H. Low and R. Srikant. A Mathematical Framework for Designing a Low-Loss Low-Delay Internet. *Network and Spatial Economics*, 4(1):75–102, March 2004.

[62] K. J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in Linear and Nonlinear Programming*. Stanford University Press, Stanford, CA, 1958.

[63] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. *RFC 3031*, January 2001.

[64] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, New Jersey, 1989.

[65] R. T. Rockafellar. Monotone Operators and the Proximal Point Algorithm. *SIAM J. Control and Optimization*, 14:877–898, August 1976.

[66] H. J. Kushner and G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, New York, 1997.

[67] H. J. Kushner and J. Yang. Analysis of Adaptive Step-Size SA Algorithms for Parameter Tracking. *IEEE Transactions on Automatic Control*, 40(8):1403–1410, August 1995.

[68] S. Kunniyur and R. Srikant. An Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. *IEEE/ACM Transactions on Networking*, 12(2):286–299, April 2004.

[69] J. Eckstein. *Splitting Methods for Monotone Operators With Applications to Prallel Optimization*. PhD thesis, Massachusetts Institute of Technology, Department of Civil Engineering, 1989.

[70] H. K. Khalil. *Nonlinear Systems*. Prentice-Hall, Upper Saddle River, New Jersey, second edition, 1996.

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

[72] S. Toumpis and A. J. Goldsmith. Capacity Regions for Wireless Ad Hoc Networks. *IEEE Transactions on Wireless Communications*, 2(4):736–748, July 2003.

[73] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic Power Allocation and Routing for Time Varying Wireless Networks. In *Proceedings of IEEE INFOCOM*, San Francisco, April 2003.

[74] R. L. Cruz and A. V. Santhanam. Optimal Routing, Link Scheduling and Power Control in Multi-hop Wireless Networks. In *Proceedings of IEEE INFOCOM*, San Francisco, April 2003.

[75] M. J. Neely and E. Modiano. Capacity and Delay Tradeoffs for Ad-Hoc Mobile Networks. *To appear in the IEEE Transactions on Information Theory, available at http://www-rcf.usc.edu/∼mjneely/*, 2003.

[76] S. Toumpis and A. J. Goldsmith. Large Wireless Networks under Fading, Mobility, and Delay Constraints. In *Proceedings of IEEE INFOCOM*, Hong Kong, China, March 2004.

[77] P. Gupta and P. R. Kumar. Critical Power for Asymptotic Connectivity in Wireless Networks. *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming. W. M. McEneany, G. Yin, and Q. Zhang (eds), Birkhauser, Boston*, 1998.

[78] S. Diggavi, M. Grossglauser, and D. Tse. Even One-Dimensional Mobility Increases Ad Hoc Wireless Capacity. In *ISIT 02*, Lausanne, Switzerland, June 2002.

[79] N. Bansal and Z. Liu. Capacity, Delay and Mobility in Wireless Ad-Hoc Networks. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, April 2003.

[80] E. Perevalov and R. Blum. Delay Limited Capacity of Ad hoc Networks: Asymptotically Optimal Transmission and Relaying Strategy. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, April 2003.

[81] P. Gupta and P. R. Kumar. Towards an Information Theory of Large Networks: An Achievable Rate Region. *IEEE Transactions on Information Theory*, 49(8):1877–1894, August 2003.

[82] M. Gastpar and M. Vetterli. On the Capacity of Wireless Networks: The Relay Case. In *Proceedings of IEEE INFOCOM*, New York, June 2002.

[83] A. E. Gamal, J. Mammen, B. Prabhakar, and D. Shah. Throughput-Delay Trade-off in Wireless Networks. In *Proceedings of IEEE INFOCOM*, Hong Kong, China, March 2004.

[84] G. Sharma and R. R. Mazumdar. Delay and Capacity Tradeoffs for Wireless Ad Hoc Networks with Random Mobility. *preprint available at http://www.ece.purdue.edu/∼mazum/*, October 2003.

[85] G. Sharma and R. R. Mazumdar. On Achievable Delay/Capacity Trade-offs in Mobile Ad Hoc Networks . In *Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WIOPT)*, Cambridge, UK, March 2004.

[86] A. El Gamal, J. Mammen, B. Prabhakar, and D. Shah. Throughput-Delay Trade-off in Wireless Networks – Part II: Constant-Size Packets. *preprint available at http://simula.stanford.edu/trade-off.html*, 2005.

[87] D. R. Brillinger. A Particle Migrating Randomly on a Sphere. *J. Theoret. Probab.*, 10(2):429–443, April 1997.

[88] R. Durrett. *Probability : Theory and Examples*. Duxbury Press, Belmont, CA, second edition, 1996.

[89] M. E. Woodward. *Communication and Computer Networks: Modelling with Discrete-Time Queues*. IEEE Computer Society Press, Los Alamitos, CA, 1994.

[90] I. Rubin and Z. Zhang. Message Delay and Queue-Size Analysis for Circuit-Switched TDMA Systems. *IEEE Transactions on Communications*, 39(6):905–914, June 1991.

[91] Z. Zhang. Tighter Upper Bounds for the Average Message Delay in a $Geom^{[X]}/Geom/N$ Queueing System. *IEEE Transactions on Communications*, 42(2/3/4):846–847, April 1994.

[92] T. Camp, J. Boleng, and V. Davies. A Survey of Mobility Models for Ad Hoc Network Research. In *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research,Trends and Applications*, 2002.

[93] X. Liu, E. K. P. Chong, and N. B. Shroff. A Framework for Opportunistic Scheduling in Wireless Networks. *Computer Networks*, 41(4):451–474, March 2003.

[94] A. Eryilmaz, R. Srikant, and J. Perkins. Stable Scheduling Policies for Fading Wireless Channels. *to appear in the IEEE/ACM Transactions on Networking, earlier versions presented at the IEEE International Symposium on Information Theory, 2002, also available at http://comm.csl.uiuc.edu/∼srikant/pub.html*.

[95] A. Eryilmaz and R. Srikant. Fair Resource Allocation in Wireless Networks Using Queue-length-based Scheduling and Congestion Control. In *Proceedings of IEEE INFOCOM*, Miami, FL, March 2005.

[96] M. Schwartz. *Mobile Wireless Communications*. Cambridge University Press, Cambridge, UK, 2005.

[97] L. Tassiulas and A. Ephremides. Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, December 1992.

[98] L. Xiao, M. Johansson, and S. Boyd. Simultaneous Routing and Resource Allocation via Dual Decomposition. In *Proceedings of 4th Asian Control Conference*, pages 29–34, Singapore, September 2002.

[99] M. Johansson, L. Xiao, and S. Boyd. Simultaneous Routing and Power Allocation in CDMA Wireless Data Networks. In *Proceedings of IEEE International Conference on Communications*, pages 51–55, Anchorage, Alaska, May 2003.

[100] M. Johansson and L. Xiao. Scheduling, Routing and Power Allocation for Fairness in Wireless Networks. In *IEEE VTC - Spring*, Milan, Italy, May 2004.

[101] M. Chiang. To Layer or Not to Layer: Balancing Transport and Physical Layers in Wireless Multihop Networks. In *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[102] M. J. Neely, E. Modiano, and C. Li. Fairness and Optimal Stochastic Control for Heterogeneous Networks. In *Proceedings of IEEE INFOCOM*, Miami, FL, March 2005.

[103] 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*, Miami, FL, March 2005.

[104] I. Paschalidis, W. Lai, and D. Starobinski. Asymptotically Optimal Transmission Policies for Low-Power Wireless Sensor Networks. In *Proceedings of IEEE INFOCOM*, Miami, FL, March 2005.

[105] V. Kawadia and P. R. Kumar. A Cautionary Perspective on Cross Layer Design. *IEEE Wireless Communications Magazine*. To appear.

[106] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhauser, 2004.

[107] S. Kunniyur and R. Srikant. End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks. *IEEE/ACM Transactions on Networking*, 11(5):689–702, October 2003.

[108] T. Bonald and L. Massoulie. Impact of Fairness on Internet Performance. In *Proceedings of ACM Sigmetrics*, pages 82–91, Cambridge, MA, June 2001.

[109] N. Z. Shor. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, Berlin, 1985.

[110] G. De Veciana, T. J. Lee, and T. Konstantopoulos. Stability and Performance Analysis of Networks Supporting Elastic Services. *IEEE/ACM Transactions on Networking*, 9(1):2–14, February 2001.

[111] G. Fayolle, A. L. Fortelle, J. M. Lasgouttes, L. Massoulie, and J. Roberts. Best Effort Networks: Modeling and Performance Analysis via Large Network Asymptotics. In *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, April 2001.

[112] H. Q. Ye. Stability of Data Networks Under an Optimization-Based Bandwidth Allocation. *IEEE Transactions on Automatic Control*, 48(7):1238–1242, July 2003.

[113] M. J. Neely, E. Modiano, and C. E. Rohrs. Power Allocation and Routing in Multibeam Satellites with Time-Varying Channels. *IEEE/ACM Transactions on Networking*, 11(1):138–152, February 2003.

[114] S. Sarkar and L. Tassiulas. End-to-end Bandwidth Guarantees Through Fair Local Spectrum Share in Wireless Ad-hoc Networks. In *Proceedings of the IEEE Conference on Decision and Control*, Maui, Hawaii, December 2003.

[115] Y. Yi and S. Shakkottai. Hop-by-hop Congestion Control over a Wireless Multi-hop Network. In *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[116] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[117] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan. On the Stability of Input-Queued Switches with Speed-Up. *IEEE/ACM Transactions on Networking*, 9(1):104–118, February 2001.

[118] Y. Xue, B. Li, and K. Nahrstedt. Price-based Resource Allocation in Wireless Ad Hoc Networks. In *Proceedings of the Eleventh International Workshop on Quality of Service (IWQoS 2003), also Lecture Notes in Computer Science, ACM Springer-Verlag*, volume 2707, pages 79–96, Monterey, CA, June 2003.

[119] J. G. Dai and B. Prabhakar. The Throughput of Data Switches with and without Speedup. In *Proceedings of IEEE INFOCOM*, pages 556–564, Tel Aviv, Israel, March 2000.

[120] T. Weller and B. Hajek. Scheduling Nonuniform Traffic in a Packet-switching System with Small Propagation Delay. *IEEE/ACM Transactions on Networking*, 5(6):813–823, December 1997.

[121] S. Shakkottai and R. Srikant. How Good are Deterministic Fluid Models of Internet Congestion Control? In *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.

[122] S. Deb, S. Shakkottai, and R. Srikant. Stability and Convergence of TCP-like Congestion Controllers in a Many-Flows Regime. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, April 2003.

[123] P. Tinnakornsrisuphap and A. M. Makowski. Limit Behavior of ECN/RED Gateways Under a Large Number of TCP Flows. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, April 2003.

[124] X. Lin and N. B. Shroff. On the Stability Region of Congestion Control. In *42st Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, October 2004.

[125] A. A. Borovkov. *Stochastic Processes in Queueing Theory, translated by K. Wickwire*. Springer-Verlag, New York, 1976.

[126] A. A. Borovkov. Ergodicity and Stability Theorems for a Class of Stochastic Equations and Their Applications. *Theory of Probability and Its Applications*, 23(2):227–247, June 1978.

[127] L. Breiman. *Probability*. Addison-Wesley, Reading, Mass., 1968.

[128] K. Sigman. *Stationary Marked Point Processes, An Intuitive Approach*. Chapman & Hall, New York, 1995.

[129] D. Y. Burman, J. P. Lehoczky, and Y. Lim. Insensitivity of Blocking Probabilities in a Circuit-Switching Network. *Journal of Applied Probability*, 21:850–859, 1984.

[130] F. Theberge, A. Simonian, and R. R. Mazumdar. Upper Bounds for Blocking Probabilities in Large Multi-rate Loss Networks. *Telecommunication Systems*, 9:23–39, 1998.

[131] L. Ljung, G. Pflug, and H. Walk. *Stochastic Approximation and Optimization of Random Systems*. Birkhauser, Basel, 1992.

[132] J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Springer, New York, 2000.

[133] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic Power Allocation and Routing for Time Varying Wireless Networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad-Hoc Networks*, 23(1):89–103, January 2005.

APPENDICES

APPENDICES

Appendix A: Supporting Results for Chapter 2

A.1 Proof of Proposition 2.2.1

Let us first focus on the case of a single link with users coming from a single class i. To develop the result, we need to take a different but equivalent view of the original model in Section 2.2. In the original system, the arrival rate is a function of the current price. In the new but equivalent system, the arrival rate is constant but the arrivals are "thinned" by a probability as a function of the price. Specifically, in the new model, the arrivals are Poisson with constant rate λ_0. Each arrival now carries a value v that is independently distributed, with distribution function $\mathbf{P}\{v \geq a\} = \lambda_i(a)/\lambda_0$ for all $a \geq 0$. The value v of each arrival is independent of the arrival process and the service times. If $v < u$, where u is the current price charged to the incoming call at the time of arrival, the call will not enter the system. It is easy to verify that the model is equivalent to the original model since the arrivals at price u (after "thinning") is still Poisson with rate $\lambda_0 \mathbf{P}\{v \geq u\} = \lambda_i(u)$.

In order to show stationarity and ergodicity, we will construct a *regenerative event* A_l as follows. Let d time units denote the length of the finite amount of past history used in the prediction ($d = 0$ if no prediction is performed). Let τ_l^e, τ_l^s, and v_l be the l-th arrival's interarrival time, service time, and value, respectively, $-\infty < l < \infty$ (note this is the arrival of the Poisson process before "thinning"). Define "epoch l" to be the time of the l-th arrival. Let

$$\begin{aligned} Q_l &= \mathbf{1}_{\{\tau_{l-1}^s \leq \tau_l^e - d\}} + \mathbf{1}_{\{\tau_{l-2}^s \leq \tau_l^e + \tau_{l-1}^e - d\}} \\ &\quad + ... + \mathbf{1}_{\{\tau_{l-k}^s \leq \sum_{j=0}^{k-1} \tau_{l-j}^e - d\}} + ..., \end{aligned}$$

and let $A_l = \{Q_l = 0\}$. Note that A_l can be interpreted as the event that "all potential arrivals (i.e., those before 'thinning') have cleared the system d time units before epoch l." The event A_l is a *regenerative event*, that is, if event A_l occurs, then after epoch l, the system will evolve independently from the past (this is true because we assume that the price is only dependent on the current state of the network, or a finite amount of past history with length d). The events A_l are stationary. Precisely, we can define $\mathbf{T}$ as the shift operator [125, p13] such that $\mathbf{T}\{\{\tau^e_{l_i}, \tau^s_{l_i}, v_{l_i}\} \in B_i, i = 1...k\} = \{\{\tau^e_{l_i+1}, \tau^s_{l_i+1}, v_{l_i+1}\} \in B_i, i = 1...k\}$ for an arbitrary collection of indices $l_1, ..., l_k$ and Borel sets $B_1, ..., B_k$. Then $A_l = \mathbf{T}^l A_0$, and $\mathbf{P}\{A_l\} = \mathbf{P}\{A_0\}$. Now to proceed with the proof, we need the following lemma.

Lemma A.1 *Let the sequence of service times τ^s be* i.i.d.*, and $\mathbf{E}[\tau^s] < \infty$, then $\mathbf{P}\{A_0\} > 0$.*

Proof We follow [125, p205]. For any $a > 0, m \geq 1$, we have,

$$\begin{aligned}
\mathbf{P}\{A_0\} &\geq \mathbf{P}\left\{\{\tau^e_0 \geq a+d\} \bigcap_{k=1}^{k=m} \left\{\tau^s_{-k} \leq a\right\} \right. \\
&\qquad \left. \bigcap_{k=m+1}^{\infty} \left\{\tau^s_{-k} \leq \sum_{j=-k+1}^{-1} \tau^e_j\right\}\right\} \\
&= \mathbf{P}\{\tau^e_0 \geq a+d\} \times \prod_{k=1}^{m} \mathbf{P}\{\tau^s_{-k} \leq a\} \\
&\qquad \times \mathbf{P}\left\{\bigcap_{k=m+1}^{\infty} \left\{\tau^s_{-k} \leq \sum_{j=-k+1}^{-1} \tau^e_j\right\}\right\}.
\end{aligned}$$

The above relationship can be interpreted as follows: A_0 is the event that $\{Q_0 = 0\}$, i.e., all potential arrivals have cleared the system d time units before epoch 0. The event on the right hand side of the inequality above says that, of all potential arrivals before epoch 0, the last arrival arrives before a time interval of $a+d$ ($\tau^e_0 \geq a+d$); the last m arrivals all have service time less than a; and finally, the rest of the arrivals leave the system before epoch -1. Obviously this is a smaller event than A_0.

We shall now focus on this smaller event. Choose a such that $\mathbf{P}\{\tau^s_{-k} \le a\} = q > 0$. Since the interarrival times are exponential, we also have $p \triangleq \mathbf{P}\{\tau^e_l \ge a + d\} > 0$. Thus,

$$\mathbf{P}\{A_0\} \ge pq^m \mathbf{P}\left\{ \bigcap_{k=m+1}^{\infty} \left\{ \tau^s_{-k} \le \sum_{j=-k+1}^{-1} \tau^e_j \right\} \right\}.$$

Let B denote the event inside the outer bracket on the right hand side. Choose $b < \mathbf{E}\{\tau^e_l\}$. We have

$$\begin{aligned}
\mathbf{P}\{B^c\} &= \mathbf{P}\left\{ \bigcup_{k=m+1}^{\infty} \left\{ \tau^s_{-k} > \sum_{j=-k+1}^{-1} \tau^e_j \right\} \right\} \\
&\le \mathbf{P}\left\{ \bigcup_{k=m+1}^{\infty} \left\{ \sum_{j=-k+1}^{-1} \tau^e_j < b(k-1) \right\} \right. \\
&\qquad \left. \bigcup_{k=m+1}^{\infty} \left\{ \tau^s_{-k} \ge b(k-1) \right\} \right\} \\
&\le \mathbf{P}\left\{ \bigcup_{k=m+1}^{\infty} \left\{ \sum_{j=-k+1}^{-1} \tau^e_j < b(k-1) \right\} \right\} \\
&\qquad + \mathbf{P}\left\{ \bigcup_{k=m+1}^{\infty} \{ \tau^s_{-k} \ge b(k-1) \} \right\}.
\end{aligned}$$

As $m \to \infty$, the first term goes to

$$\mathbf{P}\left\{ \sum_{j=-k+1}^{-1} \tau^e_j < b(k-1) \text{ infinitely often} \right\} = 0$$

by the Strong Law of Large Numbers (since $b < \mathbf{E}\{\tau^e_l\}$). On the other hand, as $m \to \infty$, the second term is bounded by

$$\sum_{k=m+1}^{\infty} \mathbf{P}\{\tau^s_{-k} \ge b(k-1)\} \to 0$$

since $\mathbf{E}\{\tau^s_l\} < \infty$. Therefore, we can choose m large enough such that $\mathbf{P}\{B^c\} < 1/2$, and thus $\mathbf{P}\{A_0\} \ge pq^m \mathbf{P}\{B\} > 0$. ■

Now we invoke Borovkov's Ergodic Theorem [126]. Since by Lemma A.1 the regenerative events A_l occur with positive probability, the distribution of the stochastic

process $\vec{n}(t)$ (i.e., the vector of the number of flows of each class in the system) converges as $t \to \infty$ to the distribution of the stationary process. Ergodicity follows from the lemma below.

Lemma A.2 *The regenerative event A_l is positive recurrent, i.e., let X_l be the state of the system at epoch l. let $T_1 = \inf\{X_l \in A_l\}$, then $\mathbf{E}\{T_1 | X_0 \in A_0\} < \infty$.*

Proof First note that

$$\mathbf{P}\{X_l \in A_l \text{ at least once}\} = \mathbf{P}\left\{\bigcup_{l=1}^{\infty} A_l\right\}.$$

Let $\mathbf{T}$ denote the shift operator defined earlier, and let $B = \bigcup_{l=1}^{\infty} A_l$, then $\mathbf{T}B \subset B$, and further $\mathbf{P}\{\mathbf{T}B\} = \mathbf{P}\{B\}$ (because B is also a stationary event). Therefore, $\mathbf{T}B$ and B differ by a set of measure zero, and thus B is an invariant set [125, p14]. By the metric transitivity [125, p14] of the *i.i.d.* sequence $\{\tau_l^e, \tau_l^s, v_l\}$, we then have $\mathbf{P}\{B\} = 0$ or 1. However, since $\mathbf{P}\{B\} \geq \mathbf{P}\{A_0\} > 0$, we must have $\mathbf{P}\{B\} = 1$, i.e., $\mathbf{P}\{X_l \in A_l \text{ at least once}\} = 1$.

By [127, Prop. 6.38, p123], we thus have

$$\mathbf{E}\{T_1 | X_0 \in A_0\} = \frac{1}{\mathbf{P}\{A_0\}} < \infty.$$

■

Since the regenerative event is positive recurrent, the stochastic process $\vec{n}(t)$ is asymptotically stationary and the stationary version is ergodic [128, Theorem 2.7, p50].

For the case of multiple classes and multiple links, we can construct the equivalent system in the following way: Assuming there are I classes, we first construct Poisson arrivals with rate $I\lambda_0$. Each of these arrivals is assigned to class i with probability $1/I$, and each of these assignments is independent of each other. The service time is then generated according to the service time distribution of class i. Each class i arrival carries a value v that is independently distributed, with distribution function

$\mathbf{P}_i\{v \geq u_i\} = \lambda_i(u_i)/\lambda_0$. The value v of each arrival is independent of the arrival process and the service times. If $v < u_i$ where u_i is the current price for class i at the time of the arrival, the call will not enter the system. Following the same idea as in the first paragraph of the proof, it is easy to show that such a constructed system is equivalent to the original system.

The initial Poisson arrivals with rate $I\lambda_0$ can be interpreted as "all potential arrivals from all classes." Let $\{\tau_n^e, \tau_n^s\}$ be the n-th arrival's interarrival time and service time respectively. It then follows that the sequence of service times τ_n^s is again i.i.d. with finite mean, and it is independent of the arrivals. Hence, we can construct the event A_0 as before, which is now the event that "all potential arrivals from all classes have cleared the system d time units before epoch n." Again this event is the "regenerative event" for the system, and we can show that $\mathbf{P}\{A_0\} > 0$, and A_0 is positive recurrent. Therefore, the system is asymptotically stationary and the stationary version is ergodic.

A.2 Proof of Lemma 2.2.1

The key idea is to use an insensitivity result from [129]. In [129], Burman *et. al.* investigate a blocking network model, where a call instantaneously seizes channels along a route between the originating and terminating node, holds the channels for a randomly distributed length of time, and frees them instantaneously at the end of the call. If no channels are available, the call is blocked. When the arrivals are Poisson and the holding time distributions are general, the authors in [129] show that the blocking probabilities are still in product form, and are insensitive to the call holding-time distributions. This means that they depend on the call duration only through its mean.

When the static prices are given, our system is a special case of [129]. Hence, we can reuse results for loss networks with Poisson arrivals and exponential holding-times. If we assume that the bandwidth requirements r_i are integers with greatest

common divisor being 1, an upper bound on the blocking probability for calls of class i is given by [130, Proposition 2.1]:

$$\mathbf{P}^c_{loss,i} \leq 2^L \sum_{l:C^l_i=1} \frac{e^{-cI_l(R^l)}}{\sqrt{2\pi c\Gamma^2_l}} \frac{1-e^{-r_i y_l}}{1-e^{-y_l}} \left[1+O(\frac{1}{\sqrt{c}})\right], \tag{A.1}$$

where y_l is the unique solution of

$$R^l = \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i e^{-y_l r_i} C^l_i, \tag{A.2}$$

and

$$\begin{aligned} \Gamma^2_l &= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i^2 e^{-y_l r_i} C^l_i \\ I_l(R^l) &= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} (1-e^{-y_l r_i}) C^l_i - y_l R^l. \end{aligned}$$

From (A.2), it is easy to verify that $y_l < 0$ if $\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i C^l_i < R^l$, and $y_l = 0$ if $\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i C^l_i = R^l$. Note that $-y_l$ is also the minimizer of $\Lambda_l(w)$ in (2.3) over $w \geq 0$, and thus

$$I_l(R^l) = -\inf_{w>0} \Lambda_l(w).$$

Hence, if the load at each resource is less than or equal to 1, i.e.,

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} r_i C^l_i \leq R^l \text{ for all } l,$$

then $y_l \leq 0$ and $I_l(R^l) \geq 0$ for all l. We have,

$$\mathbf{P}^c_{loss,i} \leq 2^L \sum_{l:C^l_i=1} \frac{1}{\sqrt{2\pi c\Gamma^2_l}} \frac{1-e^{-r_i y_l}}{1-e^{-y_l}} \left[1+O(\frac{1}{\sqrt{c}})\right],$$

i.e., $\mathbf{P}^c_{loss,i} = O(\frac{1}{\sqrt{c}})$. On the other hand, when the load of all the links that class i traverses is strictly less than 1, then $y_l < 1$ and $I_l(R^l) > 0$ for all link l that class i traverses. Hence, the exponential terms in (A.1) dominate. We thus have,

$$\begin{aligned} \limsup_{c\to\infty} \frac{1}{c} \log \mathbf{P}^c_{loss,i} &\leq \max_{l:C^l_i=1} -I_l(R^l) \\ &= \max_{l:C^l_i=1} \inf_{w>0} \Lambda_l(w) < 0. \end{aligned}$$

The above techniques (and that of [130]) can easily be generalized to the case when the bandwidth requirements r_i are positive real numbers. For a more elementary proof that does not use the result of [130], see [18].

A.3 Proof of Proposition 2.2.5

We first focus on the c-th network. To simplify notation, we will drop the index c when there is no source of confusion. Let $\lambda_i, i = 1, ..., I$ denote the solution of the upper bound (2.2). Let ϵ be a positive real number smaller than 1. Let $\lambda_i^\epsilon = (1-\epsilon)\lambda_i$ and $u_i^\epsilon = u_i(\lambda_i^\epsilon)$ for all i. Then the static revenue at static prices $u_i^\epsilon, i = 1, ..., I$ is

$$\begin{aligned} J_s^{c,\epsilon} &= \sum_{i=1}^{I} u_i(\lambda_i^\epsilon)\lambda_i^\epsilon \frac{1}{\mu_i}(1 - \mathbf{P}_{loss,i}^\epsilon) \\ &\geq \sum_{i=1}^{I} (1-\epsilon)u_i(\lambda_i)\lambda_i \frac{1}{\mu_i}(1 - \mathbf{P}_{loss,i}^\epsilon), \end{aligned}$$

where $\mathbf{P}_{loss,i}^\epsilon$ is the blocking probability of users of class i at static price $u_i^\epsilon, i = 1, ..., I$, and we have used the property that $u_i(\cdot)$ is decreasing. Thus the relative difference between $J_s^{c,\epsilon}$ and J_{ub}^c is

$$\frac{J_{ub}^c - J_s^{c,\epsilon}}{J_{ub}^c} \leq \epsilon + \max_{i=1,...,I} \mathbf{P}_{loss,i}^\epsilon. \tag{A.3}$$

Next we estimate $\mathbf{P}_{loss,i}^\epsilon$. Let n_j be the random variable that represents the number of flows of class j that are in the system. We now consider another network with the same topology and the same demand λ_i^ϵ. However, each link in the new network has infinite capacity. Let n_j^∞ be the random variable that represents the number of flows of class j that are in the infinite capacity system. By a sample path argument, $n_j \leq n_j^\infty$. Therefore,

$$\begin{aligned} \mathbf{P}_{loss,i}^\epsilon &= \mathbf{P}\{\text{There exists } l \text{ such that } C_i^l = 1 \\ &\qquad \text{and } \sum_{j=1}^{I} n_j r_j C_j^l \geq R^l - r_i\} \\ &\leq \sum_{l:C_i^l=1} \mathbf{P}\{\sum_{j=1}^{I} n_j r_j C_j^l \geq R^l - r_i\} \\ &\leq \sum_{l:C_i^l=1} \mathbf{P}\{\sum_{j=1}^{I} n_j^\infty r_j C_j^l \geq R^l - r_i\}. \end{aligned} \tag{A.4}$$

In the new system with infinite capacity, $n_j^\infty, j = 1, ..., I$ are independent Poisson random variables (by well known $M/G/\infty$ results). We can calculate their moment generating functions as

$$\mathbf{E}[\exp(\theta n_j^\infty)] \;=\; \exp[\frac{\lambda_j^\epsilon}{\mu_j}(e^\theta - 1)] \text{ for } \theta > 0 \; .$$

Fix i and l such that $C_i^l = 1$. By invoking Markov Inequality, we have,

$$\begin{aligned}
&\mathbf{P}\{\sum_{j=1}^{I} n_j^\infty r_j C_j^l \geq R^l - r_i\} \leq \frac{\mathbf{E}[\exp(\sum_{j=1}^{I} \theta r_j n_j^\infty C_j^l)]}{\exp[\theta(R^l - r_i)]} \\
= \;& \exp[\sum_{j=1}^{I} \frac{\lambda_j^\epsilon}{\mu_j} C_j^l (e^{\theta r_j} - 1) - \theta(R^l - r_i)] \\
\leq \;& \exp[\sum_{j=1}^{I} \frac{\lambda_j^\epsilon}{\mu_j} r_j C_j^l \frac{S(c)}{R^l}(e^{\theta R^l / S(c)} - 1) - \theta(R^l - r_i)] \\
\leq \;& \exp[(1-\epsilon)S(c)(e^{\frac{\theta R^l}{S(c)}} - 1) - \frac{\theta R^l}{S(c)}(S(c) - 1)], \qquad \text{(A.5)}
\end{aligned}$$

where in the last two inequalities we have used the definition of the *scale* $S(c)$ such that, for all j with $C_j^l = 1$, we have

$$r_j \leq \frac{R^l}{S(c)},$$

and thus,

$$\frac{e^{\theta r_j} - 1}{r_j} \leq \frac{e^{\theta R^l / S(c)} - 1}{R^l / S(c)};$$

and we have used the assumption that,

$$\sum_{j=1}^{I} \frac{\lambda_j^\epsilon}{\mu_j} r_j C_j^l = \sum_{j=1}^{I} (1-\epsilon)\frac{\lambda_j}{\mu_j} r_j C_j^l \leq (1-\epsilon)R^l.$$

Taking infimum of (A.5) over all $\theta > 0$, we have

$$\mathbf{P}\{\sum_{j=1}^{I} n_j^\infty r_j C_j^l \geq R^l - r_i\} \;\leq\; \exp[f(S(c))], \qquad \text{(A.6)}$$

where

$$f(s) = \inf_{\theta > 0} \left[(1-\epsilon)s(e^\theta - 1) - \theta(s-1)\right]. \qquad \text{(A.7)}$$

Note that the inequality (A.6) holds for all i and l such that $C_i^l = 1$. Substituting (A.6) into (A.4), we have,

$$\mathbf{P}_{loss,i}^{\epsilon} \leq \sum_{l:C_i^l=1} \exp[f(S(c))] \leq M \exp[f(S(c))], \tag{A.8}$$

where M is the maximum number of hops for all routes by Assumption B of scaling **(S2)**. Note that the right hand side is uniform for all class i. Substituting (A.8) into (A.3), we have,

$$\frac{J_{ub}^c - J_s^{c,\epsilon}}{J_{ub}^c} \quad \leq \quad \epsilon + M \exp[f(S(c))].$$

The function $f(s)$ can be evaluated analytically. We can easily show that

$$\lim_{s \to +\infty} \frac{f(s)}{s} \quad = \quad \epsilon + \ln(1-\epsilon).$$

Since $\epsilon \in (0,1)$, $\epsilon + \ln(1-\epsilon) < 0$. Hence, $\lim_{s\to\infty} f(s) = -\infty$. Now by Assumption A of scaling **(S2)**, $S(c) \to \infty$ as $c \to \infty$. Fix ϵ and let $c \to \infty$, we have

$$\lim_{c\to\infty} \frac{J_{ub}^c - J_s^{c,\epsilon}}{J_{ub}^c} \leq \lim_{c\to\infty} \{\epsilon + M \exp[f(S(c))]\} = \epsilon.$$

Note that $J_s^{c,\epsilon}$ is always no greater than the optimal static revenue J_s^c. Hence,

$$\lim_{c\to\infty} \frac{J_{ub}^c - J_s^c}{J_{ub}^c} \leq \epsilon.$$

This holds for any $\epsilon \in (0,1)$. Letting $\epsilon \to 0$ and noting that $J_s^c \leq J^{*,c} \leq J_{ub}^c$, the result then follows.

Appendix B: Supporting Results for Chapter 3

B.1 Proof of Proposition 3.2.1

We will focus on the case when the performance objective is the total utility. The case when the performance objective is the total revenue then follows as a special case where the utility function is the identify function.

When the performance objective is the total utility, the definition of J^*, J_0, J_s and J_{ub} needs to be modified accordingly. Given any dynamic routing policy g, one can show that the system under g will converge to a stationary version and the stationary version is ergodic (see Proposition 2.2.1 in Chapter 2). Let $N_i(0,t)$ denote the number of arrivals of class i that arrive to the system from time 0 to t. Let $N_{ij}(0,t)$ denote the number of arrivals that are admitted and routed to path j from time 0 to t. Let

$$\lambda_{ij} = \lim_{t\to\infty} \frac{N_{ij}(0,t)}{t}.$$

The quantity λ_{ij} denotes the average rate of flows of class i routed to path j. It is well defined under a given policy g due to the stationary and ergodicity of the system. By definition

$$\lambda_i = \lim_{t\to\infty} \frac{N_i(0,t)}{t}.$$

Further,

$$\sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\lambda_i} = \lim_{t\to\infty} \frac{\sum_{j=1}^{\theta(i)} N_{ij}(0,t)}{N_i(0,t)}$$

is the average proportions of flows of class i that are admitted.

Define the performance of policy g as the weighted total utility:

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\lambda_i} \right). \tag{B.1}$$

Note that this performance objective is equivalent to the average revenue when the utility function is the identify function, i.e., $U_i(p) = p$, Precisely, when $U_i(p) = p$, then

$$\begin{aligned}\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\lambda_i} \right) &= \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\mu_i} v_i \\ &= \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \mathbf{E}_g[n_{ij}] v_i,\end{aligned}$$

where n_{ij} is the random variable that denotes the number of flows of class i routed to path j that are in the system at any time, and $\mathbf{E}_g$ denote the expectation taken with respect to the stationary distribution under policy g. The last equality is by the Little's Law

$$\mathbf{E}_g[n_{ij}] = \frac{\lambda_{ij}}{\mu_i}.$$

When $U_i(p)$ is a concave function, it represents the "utility" when the average admission probability of class i is p, and $U_i'(p)$ represents the marginal utility lost if the admission probability for class i is further reduced from p.

The performance of the optimal dynamic scheme is defined as

$$J^* = \max_g \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\lambda_i} \right).$$

Note that λ_{ij} is a function of g in the above formula.

The following Proposition establishs an upper bound for J^*.

Proposition B.1 *Let J_{ub} be the solution of the following optimization problem:*

$$\begin{aligned} J_{ub} = \max_{\vec{p} \in \Omega} \quad & \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i (\sum_{j=1}^{\theta(i)} p_{ij}) \\ \textit{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij} r_i H_{ij}^l \le R^l \textit{ for all } l. \end{aligned} \tag{B.2}$$

Then

$$J^* \le J_{ub}.$$

Proof Fix a routing policy g. Let

$$p_{ij} = \frac{\lambda_{ij}}{\lambda_i}.$$

Then

$$0 \leq p_{ij} \leq 1 \text{ and } \sum_{j=1}^{\theta(i)} p_{ij} \leq 1.$$

Hence, $\vec{p} \in \Omega$.

Let n_{ij} be the random variable that denotes the number of flows of class i routed to path j that are in the system at any time. Let $\mathbf{E}_g$ denote the expectation taken with respect to the stationary distribution under policy g. By Little's Law,

$$\mathbf{E}_g[n_{ij}] = \frac{\lambda_{ij}}{\mu_i}.$$

Hence,

$$\frac{\lambda_i}{\mu_i} p_{ij} = \frac{\lambda_{ij}}{\mu_i} = \mathbf{E}_g[n_{ij}],$$

and

$$\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij} r_i H_{ij}^l = \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \mathbf{E}_g[n_{ij}] r_i H_{ij}^l.$$

By definition of n_{ij}, at any time

$$\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} n_{ij} r_i H_{ij}^l \leq R^l.$$

Therefore,

$$\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{\lambda_i}{\mu_i} p_{ij} r_i H_{ij}^l \leq R^l,$$

i.e., $\vec{p}$ satisfies the constraint of (B.2). Therefore,

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} \frac{\lambda_{ij}}{\lambda_i} \right) = \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i (\sum_{j=1}^{\theta(i)} p_{ij}) \leq J_{ub}.$$

Since this is true for any policy g, we thus have,

$$J^* \leq J_{ub}.$$

■

The performance of a static policy can be defined analogously to (B.1). That is, let $\vec{p}$ denote the static policy, the weighted utility under $\vec{p}$ is

$$J_0 = \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} p_{ij}(1 - \mathbf{P}_{Loss,ij}) \right),$$

where $\mathbf{P}_{Loss,ij}$ is the blocking probability of flows of class i routed to path j. Let J_s denote the performance of the static policy induced by the solution to the upper bound, i.e., when $\vec{p}$ is the optimal point of (B.2).

Finally, we scale the capacity and the demand proportionally by $c > 1$, i.e., in the c-scaled network, the capacity at each link l is $R^{l,c} = cR^l$, and the arrival rate of each class i is $\lambda_i^c = c\lambda_i$. Let $J^{*,c}$ and J_s^c be the utility of the optimal dynamic scheme and the utility of the static scheme induced by the solution to the upper bound, respectively, in the c-scaled system, We can now use Lemma 2.2.1 in Chapter 2 to proceed with the main proof of Proposition 3.2.1.

Proof **[of Proposition 3.2.1]** Firstly, note that the upper bound in the c-scaled system is obtained by

$$\begin{aligned} \max_{\vec{p} \in \Omega} \quad & \sum_{i=1}^{I} \frac{c\lambda_i}{\mu_i} v_i U_i (\sum_{j=1}^{\theta(i)} p_{ij}) \\ \text{subject to} \quad & \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \frac{c\lambda_i}{\mu_i} p_{ij} r_i H_{ij}^l \leq cR^l \text{ for all } l. \end{aligned}$$

Hence, the upper bound is equal to cJ_{ub} where J_{ub} is the upper bound for the base system (i.e., $c = 1$). Further, the optimal point is independent of c.

Now consider J_s^c. Note that

$$J_s^c = \sum_{i=1}^{I} \frac{c\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} p_{ij}(1 - \mathbf{P}_{Loss,ij}^c) \right),$$

where $\vec{p} = [p_{ij}, i = 1, ..., I, j = 1, ..., \theta(i)]$ is the solution to the upper bound, and $\mathbf{P}_{Loss,ij}^c$ is the blocking probability of flows of class i on path j in the c-scaled system. Since the arrivals of each class i are Poisson with rate λ_i, the flows that are routed

to path j by the static policy $\vec{p}$ also form a Poisson process with rate $\lambda_i p_{ij}$ and are independent of flows that are routed to other paths. Since $\vec{p}$ satisfies the constraint in (B.2), the load at each link is less that 1. Hence, by Lemma 2.2.1, the blocking probability $\mathbf{P}^c_{Loss,ij}$ goes to zero as $c \to \infty$. Note that U_i is continuous due to concavity, therefore,

$$\begin{aligned}
\lim_{c\to\infty} \frac{J^c_s}{c} &= \lim_{c\to\infty} \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i \left(\sum_{j=1}^{\theta(i)} p_{ij}(1 - \mathbf{P}^c_{Loss,ij}) \right) \\
&= \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} v_i U_i (\sum_{j=1}^{\theta(i)} p_{ij}) \\
&= J_{ub}.
\end{aligned}$$

Finally, since $J^c_s \le J^{*,c} \le cJ_{ub}$, the result then follows. ■

B.2 An Efficient Algorithm for Solving the Local Subproblem (3.8)

Given the implicit costs $\vec{q}$, each class i solves its local subproblem (3.8) to obtain the routing probabilities. Recall that the local subproblem is:

$$B_i(\vec{q_i}, \vec{y_i}) = \max_{\vec{p_i}\in\Omega_i} \left\{ U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} q_{ij} - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - y_{ij})^2 v_i \right\}, \quad \text{(B.3)}$$

where

$$\Omega_i \triangleq \{p_{ij} \ge 0, \sum_{j=1}^{\theta(i)} p_{ij} \le 1, \text{ for all } j\}.$$

Let L_j be the Lagrangian multiplier for the constraint $p_{ij} \ge 0$, and let L_0 be the Lagrangian multiplier for the constraint $\sum_{j=1}^{\theta(i)} p_{ij} \le 1$. Then, the Karush-Kuhn-Tucker condition becomes:

$$\begin{aligned}
&L_0 \ge 0, L_j \ge 0, j = 1, ..., \theta(i), \\
&p_{ij} \ge 0, \sum_{j=1}^{\theta(i)} p_{ij} \le 1, j = 1, ..., \theta(i), \\
&L_j p_{ij} = 0, L_0(\sum_{j=1}^{\theta(i)} p_{ij} - 1) = 0, j = 1, ..., \theta(i),
\end{aligned}$$

$$U_i'(\sum_{j=1}^{\theta(i)} p_{ij})v_i - r_i q_{ij} - \nu_i(p_{ij} - y_{ij})v_i + L_j - L_0 = 0.$$

Let $Q_{ij} = \nu_i y_{ij} v_i - r_i q_{ij}$. The last equation becomes:

$$U_i'(\sum_{j=1}^{\theta(i)} p_{ij})v_i - \nu_i v_i p_{ij} + Q_{ij} + L_j - L_0 = 0. \tag{B.4}$$

Without loss of generality, assume that the alternate paths are ordered such that $Q_{i,1} \geq Q_{i,2} \geq ... \geq Q_{i,\theta(i)}$. Then we can show the following:

Lemma B.3 *For any* $1 \leq k \leq j \leq \theta(i)$, $p_{ik} \geq p_{ij}$.

Proof The result trivially holds if $p_{ij} = 0$. If $p_{ij} > 0$, we have $L_j = 0$. Then for any $k < j$,

$$U_i'(\sum_{j=1}^{\theta(i)} p_{ij})v_i - \nu_i v_i p_{ij} + Q_{ij} - L_0 = 0, \text{ and}$$
$$U_i'(\sum_{j=1}^{\theta(i)} p_{ij})v_i - \nu_i v_i p_{ik} + Q_{ik} + L_k - L_0 = 0.$$

Hence,

$$\nu_i v_i(p_{ik} - p_{ij}) = (Q_{ik} - Q_{ij}) + L_k \geq 0,$$

i.e.,

$$p_{ik} \geq p_{ij}.$$

■

By the above Lemma, there must exists a number J such that

$$p_{ij} > 0 \text{ for any } j \leq J, \text{ and } p_{ij} = 0 \text{ for any } j > J.$$

This number J is important because once J is known, the routing probabilities p_{ij} can be easily found. To see this, let $L_j = 0$ and sum (B.4) for all $j \leq J$. We have,

$$JU_i'(\sum_{j=1}^{\theta(i)} p_{ij})v_i - \nu_i v_i \sum_{j=1}^{\theta(i)} p_{ij} + \sum_{j=1}^{J} Q_{ij} - JL_0 = 0,$$
$$0 \leq \sum_{j=1}^{\theta(i)} p_{ij} \leq 1, L_0 \geq 0, \text{ and } L_0 \left(\sum_{j=1}^{\theta(i)} p_{ij} - 1 \right) = 0.$$

Let $f(x) = JU_i'(x)v_i - \nu_i v_i x$. To find the value of $\sum_{j=1}^{\theta(i)} p_{ij}$, it is equivalent to solve x and L_0 such that

$$f(x) + \sum_{j=1}^{J} Q_{ij} - JL_0 = 0, \tag{B.5}$$

where either $x = 1$ and $L_0 \geq 0$, or $L_0 = 0$ and $0 \leq x \leq 1$. Note that $f(x)$ is decreasing in x due the concavity of U_i. If

$$f(1) + \sum_{j=1}^{J} Q_{ij} < 0,$$

then the solution to (B.5) should be some $x < 1$ with $L_0 = 0$, in which case x is the solution of

$$f(x) + \sum_{j=1}^{J} Q_{ij} = 0.$$

Otherwise, x should be equal to 1, and

$$L_0 = \frac{f(1) + \sum_{j=1}^{J} Q_{ij}}{J}.$$

In both cases, we can find the values of $x = \sum_{j=1}^{\theta(i)} p_{ij}$ and L_0 easily. Once these values are found, we can solve the routing probabilities p_{ij} via (B.4), i.e.,

$$p_{ij} = \begin{cases} \frac{U_i'(x)v_i + Q_{ij} - L_0}{\nu_i v_i} & \text{if } j \leq J \\ 0 & \text{if } j > J \end{cases}. \tag{B.6}$$

We have just shown that, once the number J is know, the routing probabilities p_{ij} can be easily computed. It remains to find the correct value of J. We use a linear search for finding J. We start the search by assuming $J = \theta(i)$. We then verify whether the current value of J is correct by solving p_{ij} via the procedure described earlier. If the values of p_{ij} are all non-negative, then J is correct. If fact, since the solution for p_{ij} computed in (B.6) is decreasing in j, we only need to ensure that $p_{i,J}$ is non-negative. On the other hand, if the verification fails, we reduce J by 1, and verify again; until either a correct value of J is found, or $J = 0$ and hence all p_{ij} should be zero.

We summarize below the algorithm for solving the subproblem (3.8):

1. Sort the index j such that Q_{ij} is in decreasing order.

2. Let $J = \theta(i)$ and $Q = \sum_{j=1}^{\theta(i)} Q_{ij}$.

3. If $JU_i'(1)v_i - \nu_i v_i + Q < 0$, then solve

$$JU_i'(x)v_i - \nu_i v_i x + Q = 0$$

for x[1] and let $L_0 = 0$. Otherwise, let $x = 1$ and

$$L_0 = \frac{JU_i'(1) - \nu_i v_i + Q}{J}.$$

4. Compute

$$p_{i,J} = \frac{U_i'(x)v_i + Q_{i,J} - L_0}{\nu_i v_i}.$$

   (a) If $p_{i,J} \geq 0$, then the correct value of J is found. Compute p_{ij} as

$$p_{ij} = \begin{cases} \frac{U_i'(x)v_i + Q_{ij} - L_0}{\nu_i v_i} & \text{if } j \leq J \\ 0 & \text{if } j > J \end{cases},$$

   and the algorithm terminates.

   (b) Otherwise, let $J \Leftarrow J - 1$ and let $Q \Leftarrow Q - Q_{i,J+1}$. If $J \geq 1$, go to step 3. If $J = 0$, set $p_{ij} = 0$ for all j and terminate.

We now summarize the complexity of the above algorithm. All steps except Step 1 and Step 4(a) are $O(1)$, and they may need to be executed $\theta(i)$ times in the worst case. The Step 4(a) is $O(\theta(1))$ but it only needs to be executed once. Sorting $Q_{i,j}$ in Step 1 can be executed in $O(\theta(i)\log\theta(i))$ time using an efficient sorting algorithm such as quicksort. Hence, the overall complexity is at most $O(\theta(i)\log\theta(i))$.

B.3 Properties of the Stationary Point of Algorithm A

From the definition of the stationary point of algorithm $\mathcal{A}$, one can establish the following properties that characterize any stationary point towards which algorithm $\mathcal{A}$ converges:

[1] With the choice of the utility function in Section 3.5, the solution can be written explicitly.

Proposition B.2 *Let $(\vec{p^*}, \vec{q^*})$ be a stationary point of algorithm $\mathcal{A}$. Define the cost of path j of class i to be the sum of the implicit costs over all links along the path, i.e., $q^*_{ij} = \sum_{l=1}^{L} H^l_{ij} q^{l,*}$. Let $q_{i,0} = \min_{j=1,\ldots,\theta(i)} q^*_{ij}$ denote the minimum cost among all alternate paths of class i. Then:*

1. *$p^*_{ij} > 0 \Rightarrow q^*_{ij} = q_{i,0}$ for all i, j.*
2. *Further, if the functions U_i are strictly concave, then for any two stationary points $(\vec{p}^{*,1}, \vec{q}^{*,1})$ and $(\vec{p}^{*,2}, \vec{q}^{*,2})$, we have*

$$\sum_{j=1}^{\theta(i)} p^{*,1}_{ij} = \sum_{j=1}^{\theta(i)} p^{*,2}_{ij} \text{ for all } i, \text{ and} \tag{B.7}$$

$$q^{*,1}_{i,0} = q^{*,2}_{i,0} \text{ for all } i \text{ such that } 0 < \sum_{j=1}^{\theta(i)} p^{*,1}_{ij} < 1. \tag{B.8}$$

Proof To prove Part 1, assume in the contrary that there exists a $p^*_{ih} > 0$ and there exists another path k of the same class i such that the cost of path k is lower than that of path h, i.e., $q^*_{ih} > q^*_{ik}$. Consider a small perturbation $\vec{p}_i$ around $\vec{p}^*_i$ such that:

$$\begin{aligned} p_{ih} &= p^*_{ih} - \delta, \\ p_{ik} &= p^*_{ik} + \delta, \\ p_{ij} &= p^*_{ij} \text{ if } j \neq h \text{ and } j \neq k, \end{aligned}$$

where $0 < \delta < p^*_{ih}$. Then $\sum_{j=1}^{\theta(i)} p_{ij} = \sum_{j=1}^{\theta(i)} p^*_{ij}$, $\vec{p}_i \in \Omega_i$ and

$$\begin{aligned} &\left\{ U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} q^*_{ij} - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - p^*_{ij})^2 v_i \right\} \\ &- \left\{ U_i(\sum_{j=1}^{\theta(i)} p^*_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p^*_{ij} q^*_{ij} - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p^*_{ij} - p^*_{ij})^2 v_i \right\} \\ = \;& r_i \delta q^*_{ih} - r_i \delta q^*_{ik} - \nu_i \delta^2 v_i, \end{aligned}$$

which is positive for small enough δ. This contradicts with the definition that $\vec{p}^*_i$ should maximize

$$U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} q^*_{ij} - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - p^*_{ij})^2 v_i$$

over all $\vec{p}_i \in \Omega_i$. This proves Part 1.

To prove Part 2, let $\vec{p}$ be a convex combination of $\vec{p}^{*,1}$ and $\vec{p}^{*,2}$, i.e., let $0 < \delta < 1$ and let

$$\vec{p} = \delta \vec{p}^{*,1} + (1-\delta)\vec{p}^{*,2}.$$

Then $\vec{p}$ also satisfies the constraints of the upper bound (3.3) and

$$\sum_{j=1}^{\theta(i)} p_{ij} = \delta \sum_{j=1}^{\theta(i)} p_{ij}^{*,1} + (1-\delta) \sum_{j=1}^{\theta(i)} p_{ij}^{*,2}.$$

Let u_0 be the optimal value of (3.3), which is achieved at both $\vec{p}^{*,1}$ and $\vec{p}^{*,2}$ by definition. Due to the concavity of U_i, we have

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i \geq \delta \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}^{*,1}) v_i + (1-\delta) \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}^{*,2}) v_i = u_0.$$

However, since u_0 is the optimal value, only equality can hold. Hence

$$\sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i = \delta \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}^{*,1}) v_i + (1-\delta) \sum_{i=1}^{I} \frac{\lambda_i}{\mu_i} U_i(\sum_{j=1}^{\theta(i)} p_{ij}^{*,2}) v_i.$$

Since U_i is strictly concave, the above equality is possible only if

$$\sum_{j=1}^{\theta(i)} p_{ij}^{*,1} = \sum_{j=1}^{\theta(i)} p_{ij}^{*,2} \text{ for all } i,$$

which proves (B.7).

Finally, to show (B.8), note again that $\vec{p}^{*,1}$ optimizes

$$U_i(\sum_{j=1}^{\theta(i)} p_{ij}) v_i - r_i \sum_{j=1}^{\theta(i)} p_{ij} q_{ij}^{*,1} - \sum_{j=1}^{\theta(i)} \frac{\nu_i}{2} (p_{ij} - p_{ij}^{*,1})^2 v_i$$

over all $\vec{p} \in \Omega$. If $0 < \sum_{j=1}^{\theta(i)} p_{ij}^{*,1} < 1$, then there exists some k such that $p_{ik}^{*,1} > 0$. Taking derivative of the above function with respect to p_{ik} at $\vec{p}^{*,1}$, and setting the derivative to zero, we have

$$U_i'(\sum_{j=1}^{\theta(i)} p_{ij}^{*,1}) v_i - r_i q_{ik}^{*,1} = 0.$$

(No Lagrangian multipliers are needed here because $p_{ik}^{*,1} > 0$ and $\sum_{j=1}^{\theta(i)} p_{ij}^{*,1} < 1$.) Hence,

$$q_{ik}^{*,1} = U_i'(\sum_{j=1}^{\theta(i)} p_{ij}^{*,1})\frac{v_i}{r_i}.$$

Since $p_{ik}^{*,1} > 0$, by the result in Part 1, $q_{ik}^{*,1}$ is also equal to the minimum cost among all paths of class i, i.e.,

$$q_{i,0}^{*,1} = U_i'(\sum_{j=1}^{\theta(i)} p_{ij}^{*,1})\frac{v_i}{r_i}.$$

Since $\sum_{j=1}^{\theta(i)} p_{ij}^{*,1} = \sum_{j=1}^{\theta(i)} p_{ij}^{*,1}$ by (B.7), we thus have $q_{i,0}^{*,1} = q_{i,0}^{*,2}$. ■

Appendix C: Supporting Results for Chapter 4

C.1 Proof of Lemma 4.3.2

We need to use the fact that $\mathbf{f}(\vec{x})$ is of the form $\sum_{i=1}^{I} f_i(\sum_{j=1}^{\theta(i)} x_{ij})$, and that the Lagrangian $L(\vec{x}, \vec{q}, \vec{y})$ is given by (4.6). The maximization of the Lagrangian (in (4.7)) is taken over $\vec{x}_i \in C_i$ for all i. Since C_i is of the form in (4.3), we can incorporate the constraint $\sum_{j=1}^{\theta(i)} x_{ij} \in [m_i, M_i]$ into the definition of the function f_i by setting $f_i(x) = -\infty$ when $x \notin [m_i, M_i]$. Then the function f_i is still concave, and the maximization of the Lagrangian $L(\vec{x}, \vec{q}, \vec{y})$ can be taken over all $\vec{x} \geq 0$. Given $\vec{y}$ and $\vec{q}$, we associate a Lagrange multiplier L^0_{ij} for each constraint $x_{ij} \geq 0$ in the maximization of $L(\vec{x}, \vec{q}, \vec{y})$, and let $\vec{x^0} = \text{argmax}_{\vec{x} \geq 0} L(\vec{x}, \vec{q}, \vec{y})$. Using the Karush-Kuhn-Tucker condition, we can conclude that, for each i, there must exist a subgradient $\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,0})$ of f_i at $\sum_{j=1}^{\theta(i)} x_{ij,0}$ such that, for all j,

$$\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,0}) - \sum_{l=1}^{L} E^l_{ij} q^l - c_i(x_{ij,0} - y_{ij}) + L^0_{ij} = 0, \text{ and } L^0_{ij} x_{ij,0} = 0. \quad \text{(C.1)}$$

Similarly, let $(\vec{y^*}, \vec{q^*})$ denote a stationary point of algorithm $\mathcal{A}$. Then $\vec{y^*} = \text{argmax}_{\vec{x} \geq 0} L(\vec{x}, \vec{q^*}, \vec{y^*})$. Associate a Lagrange multiplier L^*_{ij} for each constraint $x_{ij} \geq 0$ in the maximization of $L(\vec{x}, \vec{q^*}, \vec{y^*})$. Then, for all i, j,

$$\partial f_i(\sum_{j=1}^{\theta(i)} y^*_{ij}) - \sum_{l=1}^{L} E^l_{ij} q^{l,*} + L^*_{ij} = 0, \text{ and } L^*_{ij} y^*_{ij} = 0. \quad \text{(C.2)}$$

Comparing (C.1) and (C.2) with (4.21) and (4.22), we see that

$$[\nabla \mathbf{f}(\vec{x^0})]_{ij} = \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,0}) + L^0_{ij} \text{ for all } i, j,$$

and

$$[\nabla \mathbf{f}(\vec{y^*})]_{ij} = \partial f_i(\sum_{j=1}^{\theta(i)} y^*_{ij}) + L^*_{ij} \text{ for all } i, j,$$

where $[\cdot]_{ij}$ is the element in $[\cdot]$ that corresponds to x_{ij}.

We can now proceed with the proof of Lemma 4.3.2. Let $\vec{x}_1 = \text{argmax}_{\vec{x}\geq 0} L(\vec{x}, \vec{q}_1, \vec{y})$ and $\vec{x}_2 = \text{argmax}_{\vec{x}\geq 0} L(\vec{x}, \vec{q}_2, \vec{y})$. Analogously to L^0_{ij} and $\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,0})$, define $L_{ij,1}$, $\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1})$ and $L_{ij,2}$, $\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2})$ for the case when the implicit cost vectors are $\vec{q}_1$ and $\vec{q}_2$, respectively. Then,

$$\begin{aligned}
&\left[\nabla \mathbf{f}(\vec{x}_1) - \nabla \mathbf{f}(\vec{y^*})\right]^T (\vec{x}_2 - \vec{y^*}) \\
= &\sum_{i=1}^{I} \left[\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1}) - \partial f_i(\sum_{j=1}^{\theta(i)} y^*_{ij})\right] (\sum_{j=1}^{\theta(i)} x_{ij,2} - \sum_{j=1}^{\theta(i)} y^*_{ij}) \qquad \text{(C.3)} \\
&+ \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} (L_{ij,1} - L^*_{ij})(x_{ij,2} - y^*_{ij}). \qquad \text{(C.4)}
\end{aligned}$$

Lemma 4.3.2 will follow if we can show that both of the two terms (C.3) and (C.4) are bounded by $\frac{1}{4c_i} \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \left[\sum_{l=1}^{L} E^l_{ij}(q^l_2 - q^l_1)\right]^2$. We will first bound the term (C.3). Apply equation (C.1) for $\vec{q}_1$ and $\vec{q}_2$, respectively, and take difference. We have, for each i, j,

$$\begin{aligned}
&\sum_{l=1}^{L} E^l_{ij}(q^l_2 - q^l_1) \\
= &\left[\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2}) - \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1})\right] - c_i(x_{ij,2} - x_{ij,1}) + L_{ij,2} - L_{ij,1}. \qquad \text{(C.5)}
\end{aligned}$$

Now fix i. Let J_i denote the set $\{j : x_{ij,2} > 0 \text{ or } x_{ij,1} > 0\}$. Note that if $x_{ij,2} > 0$ and $x_{ij,1} = 0$, then $L_{ij,2} = 0$ and $L_{ij,1} \geq 0$. Hence, $x_{ij,2} - x_{ij,1} > 0$ and $L_{ij,2} - L_{ij,1} \leq 0$. Let

$$\gamma_{ij} \triangleq -\frac{L_{ij,2} - L_{ij,1}}{c_i(x_{ij,2} - x_{ij,1})} \geq 0,$$

then,

$$\sum_{l=1}^{L} E^l_{ij}(q^l_2 - q^l_1) = \left[\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2}) - \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1})\right] - (1 + \gamma_{ij}) c_i(x_{ij,2} - x_{ij,1}). \qquad \text{(C.6)}$$

Similarly, we can show that (C.6) holds for any $j \in J_i$ with some appropriate choice of $\gamma_{ij} \geq 0$. Multiplying (C.6) by $1/(1+\gamma_{ij})$ and summing over all $j \in J_i$, we have, for all i,

$$\begin{aligned} & \sum_{j\in J_i}\left[\frac{1}{1+\gamma_{ij}}\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right] \\ = \; & \frac{1}{\gamma_i^0}\left[\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2}) - \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1})\right] - c_i(\sum_{j=1}^{\theta(i)} x_{ij,2} - \sum_{j=1}^{\theta(i)} x_{ij,1}), \end{aligned} \tag{C.7}$$

where $\gamma_i^0 \triangleq \frac{1}{\sum_{j\in J_i}\frac{1}{1+\gamma_{ij}}}$, and we have used the fact that $x_{ij,2} = x_{ij,1} = 0$ for $j \notin J_i$.

Let

$$a_1 = \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1}) - \partial f_i(\sum_{j=1}^{\theta(i)} y_{ij}^*), \qquad a_2 = \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2}) - \partial f_i(\sum_{j=1}^{\theta(i)} y_{ij}^*),$$
$$b_1 = \sum_{j=1}^{\theta(i)} x_{ij,1} - \sum_{j=1}^{\theta(i)} y_{ij}^*, \qquad b_2 = \sum_{j=1}^{\theta(i)} x_{ij,2} - \sum_{j=1}^{\theta(i)} y_{ij}^*.$$

Since the function f_i is concave, we have $a_1 b_1 \leq 0$ and $a_2 b_2 \leq 0$. Let $\gamma_i \triangleq -\frac{c_i\gamma_i^0 b_1}{a_1} \geq 0$. (The term (C.3) will be bounded by $\frac{1}{4c_i}\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)}\left[\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right]^2$ trivially if $a_1 = 0$.) Then

$$\begin{aligned} (1+\gamma_i)a_1 b_2 &= (a_1 - c_i\gamma_i^0 b_1)b_2 \\ &= [(a_1 - a_2) - c_i\gamma_i^0(b_1 - b_2)]b_2 + (a_2 - c_i\gamma_i^0 b_2)]b_2 \\ &\leq \gamma_i^0 \sum_{j\in J_i}\left[\frac{1}{1+\gamma_{ij}}\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right] b_2 \quad \text{(by (C.7))} \\ &\quad -c_i\gamma_i^0 b_2^2 \quad \text{(by } a_2 b_2 \leq 0) \\ &\leq \frac{\gamma_i^0}{4c_i}\left\{\sum_{j\in J_i}\left[\frac{1}{1+\gamma_{ij}}\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right]\right\}^2 \\ &\qquad \text{(by completing the square)} \\ &\leq \frac{\gamma_i^0}{4c_i}\left\{\sum_{j\in J_i}(\frac{1}{1+\gamma_{ij}})^2\right\}\left\{\sum_{j\in J_i}\left[\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right]^2\right\} \\ &\qquad \text{(by Cauchy-Schwarz)} \\ &\leq \frac{1}{4c_i}\sum_{j=1}^{\theta(i)}\left[\sum_{l=1}^{L} E_{ij}^l(q_2^l - q_1^l)\right]^2, \end{aligned}$$

where in the last inequality we have used

$$\gamma_i^0 \left\{ \sum_{j \in J_i} (\frac{1}{1+\gamma_{ij}})^2 \right\} = \frac{\sum_{j \in J_i} (\frac{1}{1+\gamma_{ij}})^2}{\sum_{j \in J_i} \frac{1}{1+\gamma_{ij}}} \leq 1.$$

Since $1 + \gamma_i \geq 1$, the term (C.3) (i.e., $a_1 b_2$) is then bounded by $\frac{1}{4c_i} \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \left[\sum_{l=1}^{L} E_{ij}^l (q_2^l - q_1^l) \right]^2$.

To bound the term (C.4), note that $x_{ij,2} \geq 0, L_{ij,1} \geq 0, L_{ij}^* \geq 0$ and $L_{ij}^* y_{ij}^* = L_{ij,2} x_{ij,2} = L_{ij,1} x_{ij,1} = 0$. Therefore,

$$\begin{aligned} (L_{ij,1} - L_{ij}^*)(x_{ij,2} - y_{ij}^*) &\leq x_{ij,2} L_{ij,1} \leq x_{ij,1} L_{ij,2} + x_{ij,2} L_{ij,1} \\ &= -(L_{ij,2} - L_{ij,1})(x_{ij,2} - x_{ij,1}). \end{aligned}$$

We thus have,

$$\begin{aligned} & \sum_{j=1}^{\theta(i)} (L_{ij,1} - L_{ij}^*)(x_{ij,2} - y_{ij}^*) \\ \leq & -\sum_{j=1}^{\theta(i)} (L_{ij,2} - L_{ij,1})(x_{ij,2} - x_{ij,1}) \\ \leq & -\sum_{j=1}^{\theta(i)} \left[\partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,2}) - \partial f_i(\sum_{j=1}^{\theta(i)} x_{ij,1}) + L_{ij,2} - L_{ij,1} \right] (x_{ij,2} - x_{ij,1}) \\ & \qquad \text{(by the concavity of } \partial f_i) \\ \leq & -\sum_{j=1}^{\theta(i)} \left[\sum_{l=1}^{L} E_{ij}^l (q_2^l - q_1^l) + c_i (x_{ij,2} - x_{ij,1}) \right] (x_{ij,2} - x_{ij,1}) \quad \text{(by (C.5))} \\ \leq & \frac{1}{4c_i} \sum_{j=1}^{\theta(i)} \left[\sum_{l=1}^{L} E_{ij}^l (q_2^l - q_1^l) \right]^2 \quad \text{(by completing the square).} \end{aligned}$$

The result of Lemma 4.3.2 then follows.

C.2 Proof of Proposition 4.3.2 for $K = \infty$ or $1 < K < \infty$

The following lemma will be quite convenient later on.

Lemma C.4 *If*

$$\max_l \alpha^l < \frac{a}{\mathcal{SL}} \min_i c_i \text{ for some positive number } a,$$

then

$$aV > E^T AE \text{ and } aA^{-1} > EV^{-1}E^T.$$

Proof To show the first part, let $\vec{x}$ be any vector,

$$\begin{aligned}
\vec{x}^T E^T AE\vec{x} &= \sum_{l=1}^{L} \alpha^l \left[\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij} \right]^2 \\
&\leq \sum_{l=1}^{L} \alpha^l (\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l) \left[\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}^2 \right] \leq \mathcal{S} \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} \left[\sum_{l=1}^{L} \alpha^l E_{ij}^l \right] x_{ij}^2 \\
&\leq \mathcal{SL} \max_l \alpha^l \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} x_{ij}^2.
\end{aligned}$$

Hence a sufficient condition for $aV - E^T AE$ to be positive definite is

$$\max_l \alpha_l < \frac{a}{\mathcal{SL}} \min_i c_i.$$

The second part can be shown analogously. ■

Now we can proceed with the proof of Proposition 4.3.2 for $K = \infty$ or $1 < K < \infty$.

The Case with K = ∞:

We only need to show that the step A1 converges. The convergence of the entire algorithm $\mathcal{A}$ then follows the standard results on Proximal Optimization Algorithms [64, p233]. Fix $\vec{y}(t)$. Let $\vec{q}_0$ denote a stationary point of (4.10) in step A1. Let $\vec{x^0}$ be the corresponding prime variables. Note that $\vec{x^0}$ is unique and $\vec{x^0} = \text{argmax}_{\vec{x}} L(\vec{x}, \vec{q}_0, \vec{y}(t))$. Using the property of the projection mapping [64, Proposition 3.2(c), p211], we have

$$
\begin{aligned}
& ||\vec{q}(t,k+1)-\vec{q}_0||_A \\
= \;& ||\Big[\vec{q}(t,k)+A(E\vec{x}(t,k)-R)\Big]^+ - \Big[\vec{q}_0+A(E\vec{x^0}-R)\Big]^+ ||_A \\
\leq \;& ||\Big[\vec{q}(t,k)+A(E\vec{x}(t,k)-R)\Big] - \Big[\vec{q}_0+A(E\vec{x^0}-R)\Big] ||_A \\
= \;& ||\left[\vec{q}(t,k)-\vec{q}_0\right]+AE(\vec{x}(t,k)-\vec{x^0})||_A \\
= \;& ||\vec{q}(t,k)-\vec{q}_0||_A+(\vec{x}(t,k)-\vec{x^0})^TE^TAE(\vec{x}(t,k)-\vec{x^0}) \\
& +2(\vec{q}(t,k)-\vec{q}_0)^TE(\vec{x}(t,k)-\vec{x^0}).
\end{aligned}
$$

By Lemma 4.3.1, part 1,

$$
(\vec{q}(t,k)-\vec{q}_0)^TE(\vec{x}(t,k)-\vec{x^0}) \;\leq\; -(\vec{x}(t,k)-\vec{x^0})^TV(\vec{x}(t,k)-\vec{x^0}).
$$

Hence, if we can ensure that

$$
C_0 = 2V - E^TAE
$$

is positive definite, then

$$
\begin{aligned}
||\vec{q}(t,k+1)-\vec{q}_0||_A &\leq ||\vec{q}(t,k)-\vec{q}_0||_A-(\vec{x}(t,k)-\vec{x^0})^TC_0(\vec{x}(t,k)-\vec{x^0}) \qquad \text{(C.8)} \\
&\leq ||\vec{q}(t,k)-\vec{q}_0||_A.
\end{aligned}
$$

Therefore, $||\vec{q}(t,k)-\vec{q}_0||_A, k=1,2,...$ is a nonnegative and decreasing sequence, and hence must have a limit. Then by (C.8), $\vec{x}(t,k)\to\vec{x^0}$ as $k\to\infty$. By Lemma C.4, a sufficient condition for C_0 to be positive definite is

$$
\max_l \alpha_l < \frac{2}{\mathcal{SL}} \min_i c_i.
$$

The Case with K > 1:

Let $(\vec{y^*},\vec{q^*})$ be any stationary point of algorithm $\mathcal{A}$. We only need to show that the following inequality holds: for sufficiently small step-sizes $\alpha^l, l=1,...,L$,

$$
\begin{aligned}
& ||\vec{q}(t+1)-\vec{q^*}||_A+K||\vec{y}(t+1)-\vec{y^*}||_{BV}-(||\vec{q}(t)-\vec{q^*}||_A+K||\vec{y}(t)-\vec{y^*}||_{BV}) \\
\leq \;& -(\vec{q}(t+1)-\vec{q}(t))^TC_2(\vec{q}(t+1)-\vec{q}(t))-||\vec{y}(t)-\vec{x}(t)||_V \qquad \text{(C.9)}
\end{aligned}
$$

for some positive definite matrix C_2. This corresponds to the inequality (4.44) for the case when $K = 1$. The proof then follows along the same line as the case when $K = 1$.

To show (C.9), we start from (4.35). For each $k = 0, 1, ...K$,

$$\begin{aligned} & ||\vec{q}(t,k+1) - \vec{q^*}||_A - ||\vec{q}(t,k) - \vec{q^*}||_A \\ \leq \; & -||\vec{q}(t,k+1) - \vec{q}(t,k)||_A + 2(\vec{q}(t,k+1) - \vec{q^*})^T(E\vec{x}(t,k) - R). \end{aligned} \quad \text{(C.10)}$$

Since for any $k \leq K-2$,

$$\begin{aligned} \vec{q}(t,k+1) - \vec{q^*} &= \vec{q}(t,K) - \vec{q^*} - (\vec{q}(t,K) - \vec{q}(t,k+1)) \\ &= \vec{q}(t,K) - \vec{q^*} - \sum_{m=k+1}^{K-1} \gamma_m A(E\vec{x}(t,m) - R), \end{aligned}$$

where γ_m reflects the "truncation" when the implicit costs are projected to $\mathbf{R}^+$ and $0 \leq \gamma_m \leq 1$. Hence,

$$\begin{aligned} & 2(\vec{q}(t,k+1) - \vec{q^*})^T(E\vec{x}(t,k) - R) \quad \text{(C.11)} \\ = \; & 2(\vec{q}(t,K) - \vec{q^*})^T(E\vec{x}(t,k) - R) - 2\sum_{m=k+1}^{K-1} \gamma_m (E\vec{x}(t,m) - R)^T A(E\vec{x}(t,k) - R). \end{aligned}$$

If we choose α^l such that

$$\max_l \alpha^l < \frac{1}{K\mathcal{SL}} \min_i c_i, \quad \text{(C.12)}$$

then by Lemma C.4, $E^T AE < \frac{1}{K}V$. Hence,

$$\begin{aligned} & -2\sum_{m=k+1}^{K-1} \gamma_m (E\vec{x}(t,m) - R)^T A(E\vec{x}(t,k) - R) \\ \leq \; & \frac{1}{2}\sum_{m=k+1}^{K-1} \gamma_m (\vec{x}(t,m) - \vec{x}(t,k))^T E^T AE(\vec{x}(t,m) - \vec{x}(t,k)) \\ \leq \; & \frac{1}{2K}\sum_{m=k+1}^{K-1} ||\vec{x}(t,m) - \vec{x}(t,k)||_V \\ \leq \; & \frac{1}{2K}\sum_{m=k+1}^{K-1} (\vec{q}(t,m) - \vec{q}(t,k))^T EV^{-1}E(\vec{q}(t,m) - \vec{q}(t,k)), \end{aligned} \quad \text{(C.13)}$$

where in the last step we have used the result in part 2 of Lemma 4.3.1. Further, since $(\vec{y^*}, \vec{q^*})$ is a stationary point of algorithm $\mathcal{A}$, $E\vec{y^*} - R \leq 0$ and $\vec{q^*}^T(E\vec{y^*} - R) = 0$. Combining this, (C.11) and (C.13) into (C.10), we have,

$$\begin{aligned} & ||\vec{q}(t,k+1) - \vec{q^*}||_A - ||\vec{q}(t,k) - \vec{q^*}||_A \\ \leq\ & -||\vec{q}(t,k+1) - \vec{q}(t,k)||_A + 2(\vec{q}(t,K) - \vec{q^*})^T E(\vec{x}(t,k) - \vec{y^*}) \\ & + \frac{1}{2K} \sum_{m=k+1}^{K-1} (\vec{q}(t,m) - \vec{q}(t,k))^T E V^{-1} E (\vec{q}(t,m) - \vec{q}(t,k)). \end{aligned}$$

Summing over all $k = 0, 1, ...K-1$, we have

$$\begin{aligned} & ||\vec{q}(t,K) - \vec{q^*}||_A - ||\vec{q}(t,0) - \vec{q^*}||_A \\ \leq\ & -\sum_{k=0}^{K-1} ||\vec{q}(t,k+1) - \vec{q}(t,k)||_A + 2\sum_{k=0}^{K-1} (\vec{q}(t,K) - \vec{q^*})^T E(\vec{x}(t,k) - \vec{y^*}) \\ & + \frac{1}{2K} \sum_{k=0}^{K-2} \sum_{m=k+1}^{K-1} (\vec{q}(t,m) - \vec{q}(t,k))^T E V^{-1} E^T (\vec{q}(t,m) - \vec{q}(t,k)). \end{aligned}$$

Therefore, using (4.37) and (4.40), we have,

$$\begin{aligned} & ||\vec{q}(t,K) - \vec{q^*}||_A - ||\vec{q}(t,0) - \vec{q^*}||_A + K(||\vec{y}(t+1) - \vec{y^*}||_{BV} - ||\vec{y}(t) - \vec{y^*}||_{BV}) \\ \leq\ & -\sum_{k=0}^{K-1} ||\vec{q}(t,k+1) - \vec{q}(t,k)||_A \\ & + \left\{ K||\vec{z}(t) - \vec{y^*})||_V - K||\vec{y}(t) - \vec{y^*}||_V - 2\sum_{k=0}^{K-1} (\vec{z}(t) - \vec{y}(t))^T V(\vec{x}(t,k) - \vec{y^*}) \right\} \qquad \text{(C.14)} \\ & + 2\sum_{k=0}^{K-1} (\nabla \mathbf{f}(\vec{z}(t) - \nabla \mathbf{f}(\vec{y^*}))^T (\vec{x}(t,k) - \vec{y^*}) \qquad \text{(C.15)} \\ & + \frac{1}{2K} \sum_{k=0}^{K-2} \sum_{m=k+1}^{K-1} (\vec{q}(t,m) - \vec{q}(t,k))^T E V^{-1} E^T (\vec{q}(t,m) - \vec{q}(t,k)). \end{aligned}$$

The second term (C.14) on the right hand side can be bounded by

$$\begin{aligned} & K||\vec{z}(t) - \vec{y^*})||_V - K||\vec{y}(t) - \vec{y^*}||_V - 2\sum_{k=0}^{K-1} (\vec{z}(t) - \vec{y}(t))^T V(\vec{x}(t,k) - \vec{y^*}) \\ =\ & -\sum_{k=0}^{K-1} ||\vec{x}(t,k) - \vec{y}(t)||_V + \sum_{k=0}^{K-1} ||\vec{z}(t) - \vec{x}(t,k)||_V \end{aligned}$$

$$\leq \; -\sum_{k=0}^{K-1} ||\vec{x}(t,k) - \vec{y}(t)||_V + \sum_{k=0}^{K-1} (\vec{q}(t,K) - \vec{q}(t,k))^T E V^{-1} E^T (\vec{q}(t,K) - \vec{q}(t,k)),$$

where in the last step we have used the result in part 2 of Lemma 4.3.1. For the third term (C.15), by Lemma 4.3.2,

$$\begin{aligned} & 2\sum_{k=0}^{K-1} (\nabla \mathbf{f}(\vec{z}(t) - \nabla \mathbf{f}(\vec{y^*}))^T (\vec{x}(t,k) - \vec{y^*}) \\ \leq \; & \sum_{k=0}^{K-1} (\vec{q}(t,k) - \vec{q}(t,K))^T E V^{-1} E^T (\vec{q}(t,k) - \vec{q}(t,K)). \end{aligned}$$

Substituting them back to (C.14) and (C.15), we have

$$\begin{aligned} & ||\vec{q}(t,K) - \vec{q^*}||_A - ||\vec{q}(t,0) - \vec{q^*}||_A + K(||\vec{y}(t+1) - \vec{y^*}||_{BV} - ||\vec{y}(t) - \vec{y^*}||_{BV}) \\ \leq \; & -\sum_{k=0}^{K-1} ||\vec{q}(t,k+1) - \vec{q}(t,k)||_A - \sum_{k=0}^{K-1} ||\vec{x}(t,k) - \vec{y}(t)||_V \\ & +2\sum_{k=0}^{K-1} (\vec{q}(t,k) - \vec{q}(t,K))^T E V^{-1} E^T (\vec{q}(t,k) - \vec{q}(t,K)) \qquad \text{(C.16)} \\ & +\frac{1}{2K} \sum_{k=0}^{K-2} \sum_{m=k+1}^{K-1} (\vec{q}(t,m) - \vec{q}(t,k))^T E V^{-1} E^T (\vec{q}(t,m) - \vec{q}(t,k)). \qquad \text{(C.17)} \end{aligned}$$

Note that

$$\vec{q}(t,K) - \vec{q}(t,k) = \sum_{m=k}^{K-1} \vec{q}(t,m+1) - \vec{q}(t,m).$$

Using the Cauchy-Schwarz inequality, we can show that the term in (C.16) is equal to

$$\begin{aligned} & 2\sum_{k=0}^{K-1} (\vec{q}(t,k) - \vec{q}(t,K))^T E V^{-1} E^T (\vec{q}(t,k) - \vec{q}(t,K)) \\ \leq \; & 2\sum_{k=0}^{K-1} (K-k) \sum_{m=k}^{K-1} (\vec{q}(t,m+1) - \vec{q}(t,m))^T E V^{-1} E^T (\vec{q}(t,m+1) - \vec{q}(t,m)) \\ = \; & 2\sum_{m=0}^{K-1} \left[\sum_{k=0}^{m} (K-k) \right] (\vec{q}(t,m+1) - \vec{q}(t,m))^T E V^{-1} E^T (\vec{q}(t,m+1) - \vec{q}(t,m)) \\ \leq \; & K(K+1) \sum_{m=0}^{K-1} (\vec{q}(t,m+1) - \vec{q}(t,m))^T E V^{-1} E^T (\vec{q}(t,m+1) - \vec{q}(t,m)). \end{aligned}$$

Similarly, the term in (C.17) is equal to

$$\begin{aligned}
& \frac{1}{2K}\sum_{k=0}^{K-2}\sum_{m=k+1}^{K-1}(\vec{q}(t,m)-\vec{q}(t,k))^T EV^{-1}E^T(\vec{q}(t,m)-\vec{q}(t,k)) \\
\leq\ & \frac{1}{2K}\sum_{k=0}^{K-2}\sum_{m=k+1}^{K-1}(m-k)\sum_{n=k}^{m-1}(\vec{q}(t,n+1)-\vec{q}(t,n))^T EV^{-1}E(\vec{q}(t,n+1)-\vec{q}(t,n)) \\
=\ & \frac{1}{2K}\sum_{k=0}^{K-2}\sum_{n=k}^{K-2} \\
& \qquad \left[\sum_{m=n+1}^{K-1}(m-k)\right](\vec{q}(t,n+1)-\vec{q}(t,n))^T EV^{-1}E^T(\vec{q}(t,n+1)-\vec{q}(t,n)) \\
\leq\ & \frac{1}{2K}\sum_{n=0}^{K-2}\sum_{k=0}^{n}\frac{K(K-1)}{2}(\vec{q}(t,n+1)-\vec{q}(t,n))^T EV^{-1}E^T(\vec{q}(t,n+1)-\vec{q}(t,n)) \\
\leq\ & \frac{K(K+1)}{4}\sum_{n=0}^{K-2}(\vec{q}(t,n+1)-\vec{q}(t,n))^T EV^{-1}E^T(\vec{q}(t,n+1)-\vec{q}(t,n)).
\end{aligned}$$

Hence,

$$\begin{aligned}
& ||\vec{q}(t,K)-\vec{q^*}||_A - ||\vec{q}(t,0)-\vec{q^*}||_A + K(||\vec{y}(t+1)-\vec{y^*}||_{BV} - ||\vec{y}(t)-\vec{y^*}||_{BV}) \\
\leq\ & -\sum_{k=0}^{K-1}||\vec{q}(t,k+1)-\vec{q}(t,k)||_A - \sum_{k=0}^{K-1}||\vec{x}(t,k)-\vec{y}(t)||_V \\
& +\frac{5K(K+1)}{4}\sum_{m=0}^{K-1}(\vec{q}(t,m+1)-\vec{q}(t,m))^T EV^{-1}E^T(\vec{q}(t,m+1)-\vec{q}(t,m)).
\end{aligned}$$

Therefore, in order to have

$$||\vec{q}(t,K)-\vec{q^*}||_A - ||\vec{q}(t,0)-\vec{q^*}||_A + K(||\vec{y}(t+1)-\vec{y^*}||_{BV} - ||\vec{y}(t)-\vec{y^*}||_{BV}) \leq 0,$$

it is sufficient to let

$$C_2 = A^{-1} - \frac{5K(K+1)}{4}EV^{-1}E^T$$

be positive definite and to satisfy (C.12). By Lemma C.4, a sufficient condition is

$$\max_l \alpha^l < \frac{4}{5K(K+1)\mathcal{SL}}\min_i c_i,$$

which also satisfies (C.12) automatically.

C.3 Proof of Proposition 4.4.1

For the sake of brevity, we will drop the subscripts and write matrices A, B instead of A_0, B_0. We will follow the line of proof of Proposition 4.3.2 as in Section 4.3. Since (4.10) is replaced by (4.48), the inequality (4.34) should also be replaced by

$$(\vec{q}(t+1) - \vec{q^*})^T A^{-1}(\vec{q}(t+1) - [\vec{q}(t) + \eta_t A(E\vec{x}(t) - R + N(t))]) \leq 0.$$

Hence, following the techniques in the proof of Proposition 4.3.2, we have,

$$\begin{aligned}
& ||\vec{q}(t+1) - \vec{q^*}||_A \\
= \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2(\vec{q}(t+1) - \vec{q^*})^T A^{-1}(\vec{q}(t+1) - \vec{q}(t)) \\
\leq \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2\eta_t(\vec{q}(t+1) - \vec{q^*})^T(E\vec{x}(t) - R) \\
& +2\eta_t(\vec{q}(t+1) - \vec{q^*})^T N(t) \\
\leq \; & ||\vec{q}(t) - \vec{q^*}||_A - ||\vec{q}(t+1) - \vec{q}(t)||_A + 2\eta_t(\vec{q}(t+1) - \vec{q^*})^T E(\vec{x}(t) - \vec{y^*}) \\
& +2\eta_t(\vec{q}(t+1) - \vec{q^*})^T N(t). \qquad \text{(C.18)}
\end{aligned}$$

Let $\mathcal{F}_t$ be the σ-algebra generated by $\vec{x}(s), \vec{y}(s)$ and $\vec{q}(s)$ for all $s \leq t$. Then

$$\begin{aligned}
& \mathbf{E}[(q^l(t+1) - q^{l,*})n^l(t)|\mathcal{F}_t] \\
= \; & \mathbf{E}\left[\left\{\left[q^l(t) + \eta_t\alpha^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) - R^l)\right]^+ - q^{l,*}\right\} n^l(t)|\mathcal{F}_t\right] \\
& +\mathbf{E}\left[\left\{\left[q^l(t) + \eta_t\alpha^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) - R^l) + \eta_t\alpha^l n^l(t)\right]^+ \right.\right. \\
& \qquad \left.\left. - \left[q^l(t) + \eta_t\alpha^l(\sum_{i=1}^{I}\sum_{j=1}^{\theta(i)} E_{ij}^l x_{ij}(t) - R^l)\right]^+\right\} n^l(t)|\mathcal{F}_t\right].
\end{aligned}$$

By (4.49), the first expectation is zero. Hence,

$$\mathbf{E}[(q^l(t+1) - q^{l,*})n^l(t)|\mathcal{F}_t] \;\; \leq \;\; \mathbf{E}[\eta_t\alpha^l(n^l(t))^2|\mathcal{F}_t].$$

Combining (4.37-4.43) and (C.18), we have,

$$\mathbf{E}[||\vec{q}(t+1) - \vec{q^*}||_A + ||\vec{y}(t+1) - \vec{y^*}||_{BV}|\mathcal{F}_t]$$

$$\begin{aligned} \leq \quad & \mathbf{E}\left[||\vec{q}(t)-\vec{q^*}||_A + ||\vec{y}(t)-\vec{y^*}||_{BV} \right. \\ & -(\vec{q}(t+1)-\vec{q}(t))^T(A^{-1}-2\eta_t EV^{-1}E^T)(\vec{q}(t+1)-\vec{q}(t)) \\ & \left. -\eta_t||\vec{y}(t)-\vec{x}(t)||_V + \eta_t(\vec{q}(t+1)-\vec{q^*})^T N(t)|\mathcal{F}_t\right]. \end{aligned}$$

Without loss of generality, we can assume that η_t is small enough such that $A^{-1} - 2\eta_t EV^{-1}E^T$ is positive definite. Therefore,

$$\begin{aligned} & \mathbf{E}[||\vec{q}(t+1)-\vec{q^*}||_A + ||\vec{y}(t+1)-\vec{y^*}||_{BV}|\mathcal{F}_t] \\ \leq \quad & ||\vec{q}(t)-\vec{q^*}||_A + ||\vec{y}(t)-\vec{y^*}||_{BV} + {\eta_t}^2\mathbf{E}[N(t)^T AN(t)|\mathcal{F}_t]. \end{aligned}$$

By condition (4.50),

$$\sum_t {\eta_t}^2\mathbf{E}[N(t)^T AN(t)] < \infty.$$

Hence, by [131, Lemma 1.10, p9], there exists a non-negative number $\mathcal{V}_0 < \infty$ such that

$$||\vec{q}(t)-\vec{q^*}||_A + ||\vec{y}(t)-\vec{y^*}||_{BV} \to \mathcal{V}_0, \tag{C.19}$$

as $t \to \infty$.

It remains to show that $\mathcal{V}_0 = 0$ for some choice of the stationary point $(\vec{y^*}, \vec{q^*})$. Analogous to the proof of Proposition 4.3.2 in Section 4.3, we need to show that there exists a limit point of $(\vec{y}(t), \vec{q}(t))$ that is also a stationary point of the algorithm. The existence of such a limit point is unfortunately harder to establish than in Proposition 4.3.2. The reason is that, when A in (4.46) is replaced by $\eta_t A$, the properties in (4.47) no longer necessarily hold. To circumvent this difficulty, we use the techniques of [66] to find such a limit point. Let $s_t = \sum_{m=0}^{t-1}\eta_m$, and define the functions $\vec{y}^{\,0}(s)$ and $\vec{q}^{\,0}(s)$ by

$$\vec{y}^{\,0}(s) = \vec{y}(t) \text{ and } \vec{q}^{\,0}(s) = \vec{q}(t) \text{ if } s = s_t \ ,$$

and by linear interpolation for all other value of s. Define the left-shifted process $(\vec{y}^{\,t}(s), \vec{q}^{\,t}(s))$ by

$$\vec{y}^{\,t}(s) \quad = \quad \vec{y}^{\,0}(s+s_t) \text{ and } \vec{y}^{\,t}(s) = \vec{y}^{\,0}(s+s_t).$$

Then $(\vec{y}^{\,t}(s), \vec{q}^{\,t}(s))$ defines a sequence of continuous functions of s. Following [66], we can show that there exists a subsequence $(\vec{y}^{\,t_h}(s), \vec{q}^{\,t_h}(s))$ that converges uniformly on finite intervals to a continuous function $(\vec{\tilde{y}}(s), \vec{\tilde{q}}(s))$, which satisfies the ordinary differential equations (4.14) and (4.15). By Proposition 4.3.1, $(\vec{\tilde{y}}(s), \vec{\tilde{q}}(s))$ converges to a stationary point of the algorithm as $s \to \infty$. Let $(\vec{y}_0, \vec{q}_0)$ denote this stationary point. Then the fact that $(\vec{y}^{\,t_h}(s), \vec{q}^{\,t_h}(s))$ converges to $(\vec{\tilde{y}}(s), \vec{\tilde{q}}(s))$ uniformly on finite intervals implies that $(\vec{y}_0, \vec{q}_0)$ is also a limit point of the sequence $(\vec{y}(t), \vec{q}(t)), t = 1, 2, ...$ We can then replace $(\vec{y^*}, \vec{q^*})$ by $(\vec{y}_0, \vec{q}_0)$ in (C.19) and obtain

$$||\vec{q}(t) - \vec{q}_0||_A + ||\vec{y}(t) - \vec{y}_0||_{BV} \to 0.$$

The result then follows.

C.4 The Transfer Function from $N(t)$ to $\vec{x}(t)$

Without loss of generality, assume that $x_{ij}(t) > 0$ for all i, j, and $q^l(t) > 0$ for all l. Since

$$\vec{x}_i(t) = \underset{\vec{x}_i}{\operatorname{argmax}} \left\{ f_i(\sum_{j=1}^{\theta(i)} x_{ij}) - \sum_{j=1}^{\theta(i)} x_{ij} q_{ij}(t) - \sum_{j=1}^{\theta(i)} \frac{c_i}{2}(x_{ij} - y_{ij}(t))^2 \right\},$$

we have,

$$f_i'(\sum_{j=1}^{\theta(i)} x_{ij}(t)) - q_{ij}(t) - c_i(x_{ij}(t) - y_{ij}(t)) = 0 \text{ for all } i, j. \tag{C.20}$$

Assume that the problem (4.1) has a unique solution. Linearize the system around this stationary point, i.e.,

$$\begin{aligned}
x_{ij}(t) &= y_{ij}^* + \tilde{x}_{ij}(t), \\
y_{ij}(t) &= y_{ij}^* + \tilde{y}_{ij}(t), \\
q^l(t) &= q^{l,*} + \tilde{q}^l(t), \text{ and} \\
q_{ij}(t) &= q_{ij}^* + \tilde{q}_{ij}(t) = \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l q^{l,*} + \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{q}^l(t).
\end{aligned}$$

We obtain,

$$f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) \sum_{j=1}^{\theta(i)} \tilde{x}_{ij}(t) - \tilde{q}_{ij}(t) - c_i(\tilde{x}_{ij}(t) - \tilde{y}_{ij}(t)) = 0.$$

Summing over all j, we have,

$$\theta(i) f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) \sum_{j=1}^{\theta(i)} \tilde{x}_{ij}(t) - \sum_{j=1}^{\theta(i)} \tilde{q}_{ij}(t) - c_i(\sum_{j=1}^{\theta(i)} \tilde{x}_{ij}(t) - \sum_{j=1}^{\theta(i)} \tilde{y}_{ij}(t)) = 0.$$

Therefore,

$$\sum_{j=1}^{\theta(i)} \tilde{x}_{ij}(t) = \frac{\sum_{j=1}^{\theta(i)} \tilde{q}_{ij}(t) - c_i \sum_{j=1}^{\theta(i)} \tilde{y}_{ij}(t)}{\theta(i) f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) - c_i}.$$

We can then solve for each $\tilde{x}_{ij}(t)$ by

$$\begin{aligned} \tilde{x}_{ij}(t) &= \tilde{y}_{ij}(t) + \frac{1}{c_i}\left[f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) \sum_{j=1}^{\theta(i)} \tilde{x}_{ij}(t) - \tilde{q}_{ij}(t) \right] \\ &= \tilde{y}_{ij}(t) + \frac{1}{c_i}\left[\frac{f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*)}{\theta(i) f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) - c_i)} \left(\sum_{j=1}^{\theta(i)} \tilde{q}_{ij}(t) - c_i \sum_{j=1}^{\theta(i)} \tilde{y}_{ij}(t) \right) - \tilde{q}_{ij}(t) \right]. \end{aligned} \tag{C.21}$$

When there is measurement noise $n^l(t)$ at each link l, we can get the linearization of (4.14) and (4.15) as

$$\frac{d}{dt}\tilde{q}^l(t) = \hat{\alpha}^l \left(\sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{x}_{ij}(t) + n^l(t) \right) \tag{C.22}$$

$$\frac{d}{dt}\tilde{y}_{ij}(t) = \hat{\beta}_i(\tilde{x}_{ij}(t) - \tilde{y}_{ij}(t). \tag{C.23}$$

Let $\mathcal{Q}(s), \mathcal{Y}(s), \mathcal{X}(s), \mathcal{N}(s)$ denote the Laplace transforms of the perturbation of $\vec{q}(t)$, $\vec{y}(t)$, $\vec{x}(t)$, and the noise $N(t)$, respectively. Taking Laplace transform of (C.21-C.23), and using the fact that $\tilde{q}_{ij}(t) = \sum_{i=1}^{I} \sum_{j=1}^{\theta(i)} E_{ij}^l \tilde{q}^l(t)$, we have,

$$\begin{aligned} s\mathcal{Q}(s) &= \hat{A}(E\mathcal{X}(s) + \mathcal{N}(s)) \\ s\mathcal{Y}(s) &= \hat{B}(\mathcal{X}(s) - \mathcal{Y}(s)) \\ \mathcal{X}(s) &= \mathcal{Y}(s) + V^{-1}\left[G(E^T\mathcal{Q} - V\mathcal{Y}) - E^T\mathcal{Q} \right], \end{aligned}$$

where $G = \mathbf{diag}\{G_i, i = 1, ..., I\}$, and each G_i is a $\theta(i) \times \theta(i)$ matrix whose terms are all

$$g_i = \frac{f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*)}{\theta(i) f_i''(\sum_{j=1}^{\theta(i)} y_{ij}^*) - c_i}.$$

From the above set of equations, we can solve for $\mathcal{X}(s)$ as

$$\mathcal{X}(s) = H(s)\mathcal{N}(s)$$

with

$$\begin{aligned} H(s) &= -\left\{ \hat{B}^{-1} s\mathcal{I} + G + V^{-1}(\mathcal{I} - G)\frac{E^T \hat{A} E}{s} \hat{B}^{-1}(s\mathcal{I} + \hat{B}) \right\}^{-1} \\ &\times \quad \hat{B}^{-1}(s\mathcal{I} + \hat{B}) V^{-1}(\mathcal{I} - G)\frac{E^T \hat{A}}{s}, \end{aligned}$$

where $\mathcal{I}$ is the $\sum_{i=1}^{I} \theta(i) \times \sum_{i=1}^{I} \theta(i)$ identity matrix.

Appendix D: Supporting Results for Chapter 5

D.1 Proof of Proposition 5.4.1

We will need the following lemma on the minimum distance from the mobile relays to the destination at any time slot. Fix a bit b that enters into the system at time slot $t_0(b)$. At each time slot $t \geq t_0(b)$, recall that $r_b(t)$ is the number of mobile relays holding the bit b at the beginning of the time slot. Among these $r_b(t)$ mobile relays, there is one mobile relay whose distance to the destination of bit b is the smallest. Let $\tilde{L}_b(t)$ denote this minimum distance, and let

$$L_b(t) = \max\{\frac{1}{n^2}, \tilde{L}_b(t)\}. \tag{D.1}$$

Lemma D.5 *Under the* i.i.d. *mobility model, if* $n \geq 3$*, then*

$$\mathbf{E}\left[\frac{1}{L_b^2(t) r_b(t)} | \mathcal{F}_{t-1}\right] \leq 8\pi \log n \text{ for all } t \geq t_0(b).$$

Proof Let $\mathbf{I}_A$ be the indicator function on the set A. By the definition of $L_b(t)$, we have,

$$\begin{aligned}\mathbf{E}\left[\frac{1}{L_b^2(t)} | \mathcal{F}_{t-1}\right] &= \mathbf{E}\left[n^4 \mathbf{I}_{\{\tilde{L}_b(t) \leq \frac{1}{n^2}\}} | \mathcal{F}_{t-1}\right] \\ &\quad + \mathbf{E}\left[\frac{1}{\tilde{L}_b^2(t)} \mathbf{I}_{\{\tilde{L}_b(t) > \frac{1}{n^2}\}} | \mathcal{F}_{t-1}\right].\end{aligned}$$

Since the nodes move on a unit square, $\tilde{L}_b(t) \leq \sqrt{2}$. Hence,

$$\begin{aligned}&\mathbf{E}\left[\frac{1}{\tilde{L}_b^2(t)} \mathbf{I}_{\{\tilde{L}_b(t) > \frac{1}{n^2}\}} | \mathcal{F}_{t-1}\right] \\ =& \int_{\frac{1}{n^2}}^{\sqrt{2}} \frac{1}{u^2} d\mathbf{P}[\tilde{L}_b(t) \leq u | \mathcal{F}_{t-1}] \\ =& \frac{1}{u^2}\mathbf{P}[\tilde{L}_b(t) \leq u | \mathcal{F}_{t-1}]|_{\frac{1}{n^2}}^{\sqrt{2}} - \int_{\frac{1}{n^2}}^{\sqrt{2}} \mathbf{P}[\tilde{L}_b(t) \leq u | \mathcal{F}_{t-1}] d\frac{1}{u^2} \\ =& \frac{1}{2} - n^4 \mathbf{P}[\tilde{L}_b(t) \leq \frac{1}{n^2} | \mathcal{F}_{t-1}] + \int_{\frac{1}{n^2}}^{\sqrt{2}} \frac{2}{u^3} \mathbf{P}[\tilde{L}_b(t) \leq u | \mathcal{F}_{t-1}] du.\end{aligned}$$

Hence,

$$\mathbf{E}\left[\frac{1}{L_b^2(t)}|\mathcal{F}_{t-1}\right] = \frac{1}{2} + \int_{\frac{1}{n^2}}^{\sqrt{2}} \frac{2}{u^3}\mathbf{P}[\tilde{L}_b(t) \leq u|\mathcal{F}_{t-1}]du.$$

Let ρ_b be the distance from any one mobile node to the destination of the bit b. Then, due to the *i.i.d.* mobility model, we have,

$$\mathbf{P}[\rho_b \leq u|\mathcal{F}_{t-1}] \leq \pi u^2,$$

and,

$$\begin{aligned}\mathbf{P}[\tilde{L}_b(t) \leq u|\mathcal{F}_{t-1}] &\leq 1-(1-\pi u^2)^{r_b(t)} \\ &\leq \pi r_b(t)u^2.\end{aligned}$$

Therefore,

$$\begin{aligned}\mathbf{E}\left[\frac{1}{L_b^2(t)}|\mathcal{F}_{t-1}\right] &= \frac{1}{2} + \int_{\frac{1}{n^2}}^{\sqrt{2}} \frac{2}{u^3}\mathbf{P}[\tilde{L}_b(t) \leq u|\mathcal{F}_{t-1}]du \\ &\leq \frac{1}{2} + \int_{\frac{1}{n^2}}^{\sqrt{2}} \pi r_b(t)\frac{2}{u}du \\ &= \frac{1}{2} + 2\pi r_b(t)\log u|_{\frac{1}{n^2}}^{\sqrt{2}} \\ &= \frac{1}{2} + 2\pi r_b(t)(\log\sqrt{2} + 2\log n) \\ &\leq 8\pi r_b(t)\log n,\end{aligned}$$

when $n \geq 3$. Finally, since $r_b(t)$ is $\mathcal{F}_{t-1}$-measurable, we have

$$\begin{aligned}\mathbf{E}\left[\frac{1}{L_b^2(t)r_b(t)}|\mathcal{F}_{t-1}\right] &= \frac{1}{r_b(t)}\mathbf{E}\left[\frac{1}{L_b^2(t)}|\mathcal{F}_{t-1}\right] \\ &\leq 8\pi\log n.\end{aligned}$$

■

Proof [**of Proposition 5.4.1**] Let

$$V_t = 8\pi\log n\,[t - t_0(b)] - \sum_{s=t_0(b)+1}^{t} \frac{1}{L_b^2(s)r_b(s)}\mathbf{I}_{\{C_b(s)=1\}},$$

Then for all $t \geq t_0(b)$, V_t is also $\mathcal{F}_t$-measurable and $V_{t_0(b)} = 0$. By Lemma D.5, we have

$$\begin{aligned} & \mathbf{E}[V_t - V_{t-1}|\mathcal{F}_{t-1}] \\ = \; & 8\pi \log n - \mathbf{E}\left[\frac{1}{L_b^2(t) r_b(t)} \mathbf{I}_{\{C_b(t)=1\}} | \mathcal{F}_{t-1}\right] \\ \geq \; & 8\pi \log n - \mathbf{E}\left[\frac{1}{L_b^2(t) r_b(t)} | \mathcal{F}_{t-1}\right] \\ \geq \; & 0. \end{aligned}$$

Hence,

$$\mathbf{E}[V_t|\mathcal{F}_{t-1}] \geq V_{t-1},$$

i.e., V_t is a sub-martingale. Recall that $s_b \triangleq \min\{t : t \geq t_0(b) \text{ and } C_b(t) = 1\}$ denotes the first time when a successful capture for bit b occurs. Since s_b is a stopping time, by appropriately invoking the Optional Stopping Theorem [88, p249, Theorem 4.1], we have,

$$\mathbf{E}[V_{s_b}] \;\; \geq \;\; 0.$$

Hence, substituting $D_b \triangleq s_b - t_0(b)$, $R_b \triangleq r_b(s_b)$, $C_b(s_b) = 1$ and $C_b(t) = 0$ for $t < s_b$, we have,

$$8\pi \log n \mathbf{E}[D_b] \geq \mathbf{E}\left[\frac{1}{L_b^2(s_b) R_b}\right].$$

Using Hőlder's Inequality [88, p15],

$$\mathbf{E}^2[\frac{1}{L_b(s_b)}] \leq \mathbf{E}[R_b]\mathbf{E}[\frac{1}{L_b^2(s_b) R_b}],$$

we thus have,

$$\begin{aligned} 8\pi \log n \mathbf{E}[D_b] \;\; & \geq \;\; \mathbf{E}^2[\frac{1}{L_b(s_b)}] \frac{1}{\mathbf{E}[R_b]} \\ & \geq \;\; \frac{1}{\mathbf{E}^2[L_b(s_b)]\mathbf{E}[R_b]}. \end{aligned}$$

Finally, recall that $l_b \triangleq \tilde{l}_b(s_b)$ denotes the distance from the last mobile relay node to the destination. By definition (D.1),

$$l_b = \tilde{l}_b(s_b) \geq L_b(s_b) - \frac{1}{n^2},$$

therefore,

$$8\pi \log n \mathbf{E}[D_b] \geq \frac{1}{(\mathbf{E}[l_b] + \frac{1}{n^2})^2 \mathbf{E}[R_b]}.$$

■

D.2 Proof of Proposition 5.4.3

The next Lemma will be used frequently in the proof of Propositions 5.4.3 and 5.6.1. Consider an experiment where we randomly throw n balls into $m \leq n$ urns. The probability that each ball j enters urn i is $\frac{p}{m}$ and is independent of the position of other balls. Thus, $p \leq 1$ is the *success probability* that the ball is thrown into *any* one of the urns. Let $B_i, i = 1, ...m$ be the number of balls in urn i after n balls are thrown. It is obvious that $\mathbf{E}[B_i] = \frac{np}{m}$. The following Lemma shows that, when n is large, the probability that any B_i deviates substantially from its mean will be very small.

Lemma D.6 *As $n \to \infty$,*

1) If $\frac{np}{m} \geq c \log n$ and $c \geq 8$, then

$$\mathbf{P}[B_i = 0 \text{ for any } i] = O(\frac{1}{n^3}).$$

2) If $\frac{np}{m} \geq c \log n$ and $c \geq 16$, then

$$\mathbf{P}[B_i \geq 2\frac{np}{m} \text{ for any } i] \leq \frac{1}{n^3}.$$

3) If $\frac{np}{m} \leq c \log n$ and $c \geq 16$, then

$$\mathbf{P}[B_i \geq 2c \log n \text{ for any } i] \leq \frac{1}{n^3}.$$

4) If $\frac{np}{m} \leq cn^\alpha$, where $c > 0$ and $\alpha > 0$, then

$$\mathbf{P}[B_i \geq 2cn^\alpha \text{ for any } i] = O(\frac{1}{n^3}).$$

Proof To prove part 1, note that for any i,

$$\begin{aligned}\mathbf{P}[B_i = 0] &= \left[1 - \frac{p}{m}\right]^n \\ &\leq \left[1 - \frac{8\log n}{n}\right]^n \\ &= \left[1 - \frac{8\log n}{n}\right]^{\frac{n}{8\log n}8\log n}.\end{aligned}$$

Since $\lim_{x\to 0}(1-x)^{1/x} = 1/e$, we have

$$\left[1 - \frac{8\log n}{n}\right]^{\frac{n}{8\log n}} \leq \frac{1}{\sqrt{e}} \text{ for large } n.$$

Hence, for large n,

$$\mathbf{P}[B_i = 0] \leq \left[\frac{1}{\sqrt{e}}\right]^{8\log n} = \frac{1}{n^4}.$$

Therefore,

$$\mathbf{P}[B_i = 0 \text{ for any } i\ ] \leq n\frac{1}{n^4} = O(\frac{1}{n^3}).$$

To prove the other parts, we use known results on the characteristic function of Bernoulli random variables. For any $\theta > 0$, we have

$$\begin{aligned}\mathbf{E}[e^{\theta B_i}] &= \left[e^{\theta}\frac{p}{m} + (1 - \frac{p}{m})\right]^n \\ &= \left[1 + (e^{\theta} - 1)\frac{p}{m}\right]^n \\ &\leq \exp\left[\frac{np}{m}(e^{\theta} - 1)\right] \text{ for all urn } i,\end{aligned}$$

where in the last step we have used the inequality that

$$(1 + x)^{\frac{1}{x}} \leq e \text{ for } x > 0.$$

Using the Markov Inequality [88, p15], for any $y > 0$,

$$\begin{aligned}\mathbf{P}[B_i \geq y] &\leq \frac{\mathbf{E}[e^{\theta B_i}]}{e^{\theta y}} \\ &\leq \exp\left[\frac{np}{m}(e^{\theta} - 1) - \theta y\right].\end{aligned}$$

Hence, by the union bound

$$\mathbf{P}[B_i \geq y \text{ for any } i] \leq n \exp\left[\frac{np}{m}(e^\theta - 1) - \theta y\right].$$

To prove part 2, let $y = 2\frac{np}{m}$, hence

$$\begin{aligned}\mathbf{P}[B_i \geq y \text{ for any } i] &\leq n \exp\left[\frac{np}{m}(e^\theta - 1) - \theta y\right] \\ &= n \exp\left[\frac{np}{m}(e^\theta - 1 - 2\theta)\right].\end{aligned}$$

Let $\theta = \log 2$, then

$$e^\theta - 1 - 2\theta = -(2\log 2 - 1) = -0.386 \leq -\frac{1}{4}. \tag{D.2}$$

Hence, when $\frac{np}{m} \geq c \log n$ and $c \geq 16$, we have

$$\begin{aligned}\mathbf{P}[B_i \geq y \text{ for any } i] &\leq n \exp\left[-\frac{c}{4}\log n\right] \\ &\leq n\frac{1}{n^4} = \frac{1}{n^3}.\end{aligned}$$

To prove part 3, let $y = 2c \log n$. Since $\frac{np}{m} \leq c \log n$ and $\theta > 0$, we have,

$$\begin{aligned}\mathbf{P}[B_i \geq y \text{ for any } i] &\leq n \exp\left[\frac{np}{m}(e^\theta - 1) - \theta y\right] \\ &\leq n \exp\left[c \log n(e^\theta - 1 - 2\theta)\right].\end{aligned}$$

Let $\theta = \log 2$, then using (D.2), we have

$$\begin{aligned}\mathbf{P}[B_i \geq y \text{ for any } i] &\leq n \exp\left[-\frac{c}{4}\log n\right] \\ &\leq n\frac{1}{n^4} = \frac{1}{n^3}.\end{aligned}$$

To prove part 4, let $y = 2cn^\alpha$. Since $\frac{np}{m} \leq cn^\alpha$ and $\theta > 0$, we have,

$$\begin{aligned}\mathbf{P}[B_i \geq y \text{ for any } i] &\leq n \exp\left[\frac{np}{m}(e^\theta - 1) - \theta y\right] \\ &\leq n \exp\left[cn^\alpha(e^\theta - 1 - 2\theta)\right].\end{aligned}$$

Let $\theta = \log 2$, then using (D.2), we have

$$\mathbf{P}[B_i \geq y \text{ for any } i] \leq n \exp\left[-\frac{c}{4}n^\alpha\right] \leq O(\frac{1}{n^3}).$$

■

Proof [of Proposition 5.4.3] At each time slot t, an opportunistic broadcast scheme has to determine how to replicate each bit b to a larger number of mobile nodes. Some of the mobile nodes that already have the bit b have to be selected to transmit the bit b, and some of the other mobile nodes have to be selected to receive the bit.

Let $v_b(t,i)$ be the distance from a node i that is chosen to transmit the bit b at time slot t, to the furthest node that is chosen to receive the bit directly from node i ($v_b(t,i) = 0$ if node i is not chosen to transmit the bit or if the bit b has cleared the system). Let $u_b(t,i)$ be the number of mobile nodes that are chosen to receive the bit b directly from node i and that do not have the bit b prior to time slot t ($u_b(t,i) = 0$ if $v_b(t,i) = 0$). Then $u_b(t,i)$ is bounded from above by the number of nodes covered by a disk of radius $v_b(t,i)$ centered at node i. It is easy to verify that

$$R_b - 1 = \sum_{t=1}^{T} \sum_{i=1}^{n} u_b(t,i).$$

Fix a time slot t. We next bound the number of nodes that are covered by each disk of radius $v_b(t,i)$ centered at node i. We divide the unit square into $g_4(n) = \lfloor \left(\frac{n}{16 \log n} \right)^{\frac{1}{2}} \rfloor^2$ cells (in $\sqrt{g_4(n)}$ rows and $\sqrt{g_4(n)}$ columns). Each cell is a square of area $1/g_4(n)$. Let B_i be the number of nodes in cell i, $i = 1, ..., g_4(n)$. Then $\mathbf{E}[B_i] = \frac{n}{g_4(n)}$. When n is large, we have

$$16 \log n \leq \frac{n}{g_4(n)} \leq 32 \log n.$$

Let $\mathcal{A}$ be the event that

$$B_i \leq \frac{2n}{g_4(n)} \text{ for all } i = 1, ..., g_4(n).$$

By part 2 of Lemma D.6, $\mathbf{P}[\mathcal{A}^c] \leq 1/n^3$, Now consider each disk of radius $v_b(t,i)$. We need at most

$$\left[2v_b(t,i)\sqrt{g_4(n)} + 2 \right]^2$$

cells to completely cover the disk. Hence, if event $\mathcal{A}$ occurs, the number of nodes in the disk of radius $v_b(t,i)$ will be bounded from above by

$$\begin{aligned}\left[2v_b(t,i)\sqrt{g_4(n)}+2\right]^2\frac{2n}{g_4(n)} &\leq 16nv_b^2(t,i)+\frac{16n}{g_4(n)}\\ &\leq 16nv_b^2(t,i)+512\log n.\end{aligned}$$

Note that the above relationship holds for all b and i. Let $c_7 = 16/\pi$, and $c_8 = 512$. Since $u_b(t,i)$ is no greater than the number of nodes covered by the disk of radius $v_b(t,i)$, we have,

$$\mathbf{P}\left[\frac{u_b(t,i)}{n} > c_7\pi v_b^2(t,i)+c_8\frac{\log n}{n} \text{ for any } b,i\right] \leq \mathbf{P}[\mathcal{A}^c] \leq \frac{1}{n^3}.$$

Fix a bit b. Let $\mathcal{B}$ be the event that

$$\frac{u_b(t,i)}{n} \leq c_7\pi v_b^2(t,i)+c_8\frac{\log n}{n} \text{ for all } i \text{ and } t = t_0(b),...,t_0(b)+c_2n^2.$$

Then,

$$\mathbf{P}[\mathcal{B}^c] \leq \frac{1}{n^3}c_2n^2 = \frac{c_2}{n}.$$

Since $u_b(t,i) \leq n$, we have

$$\begin{aligned}\mathbf{E}\left[\frac{u_b(t,i)}{n}\right] &= \mathbf{E}\left[\frac{u_b(t,i)}{n}\mathbf{I}_{\{\mathcal{B}\}}\right]+\mathbf{E}\left[\frac{u_b(t,i)}{n}\mathbf{I}_{\{\mathcal{B}^c\}}\right]\\ &\leq \mathbf{E}\left[c_7\pi v_b^2(t,i)+c_8\frac{\log n}{n}\right]+\mathbf{P}[\mathcal{B}^c]\\ &\leq c_7\pi\mathbf{E}[v_b^2(t,i)]+c_8\frac{\log n}{n}+\frac{c_2}{n}\\ &\leq c_7\pi\mathbf{E}[v_b^2(t,i)]+(c_8+1)\frac{\log n}{n}\end{aligned}$$

when $n \geq \max\{N_0, \exp(c_2)\}$,

We now use the idea in Section 5.4 that disks of radius $\frac{\Delta}{2}$ times the transmission range centered at the transmitter are disjoint from each other. For each unicast transmission (i.e., the transmission over each hop S_b^h), the transmission range is just S_b^h. For broadcast, the transmission range is the distance from the transmitter to the furthest node that can successfully receive the bit, i.e. $v_b(t,i)$. By counting the area covered by all the disks, we have

$$\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\pi\frac{\Delta^2}{4}v_b^2(t,i)+\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}\frac{\pi\Delta^2}{4}(S_b^h)^2 \leq WT. \tag{D.3}$$

Since there are at most n nodes that can serve as transmitters at any time, we have

$$\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\mathbf{I}_{\{v_b(t,i)>0\}} \le WTn.$$

Hence,

$$\begin{aligned}
\sum_{b=1}^{\lambda nT}\frac{\mathbf{E}[R_b]-1}{n} &= \mathbf{E}\left[\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{u_b(t,i)}{n}\right] \\
&\le c_7\pi\mathbf{E}\left[\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}v_b^2(t,i)\right] \\
&\quad +(c_8+1)\mathbf{E}\left[\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\log n}{n}\mathbf{I}_{\{v_b(t,i)>0\}}\right] \\
&\le c_7\pi\mathbf{E}\left[\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}v_b^2(t,i)\right]+(c_8+1)WT\log n. \qquad \text{(D.4)}
\end{aligned}$$

Substituting (D.4) into (D.3), we have

$$\begin{aligned}
&\sum_{b=1}^{\lambda nT}\frac{\Delta^2}{4}\frac{\mathbf{E}[R_b]-1}{n}+\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}\frac{\pi\Delta^2}{4}\mathbf{E}[(S_b^h)^2] \\
\le\ &\left\{c_7\mathbf{E}\left[\sum_{b=1}^{\lambda nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\pi\frac{\Delta^2}{4}v_b^2(t,i)\right]+\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}\frac{\pi\Delta^2}{4}\mathbf{E}[(S_b^h)^2]\right\} \\
&\quad +\frac{(c_8+1)\Delta^2}{4}WT\log n \\
\le\ & c_7WT+\frac{(c_8+1)\Delta^2}{4}WT\log n \\
\le\ & \frac{(c_8+2)\Delta^2}{4}WT\log n,
\end{aligned}$$

when $n \ge \max\{N_0, \exp(c_2), \exp(\frac{4c_7}{\Delta^2})\}$. ■

D.3 Proof of Lemma 5.6.1

We can group all cells into $c_4 = \lfloor 2\Delta+6\rfloor^2$ lattices. Each lattice consists of nodes that are $\lfloor 2\Delta+6\rfloor/m$ apart along the X-axis or the Y-axis (see Fig.5.2). The cells of each lattice can be active at the same time since

- the transmission range from a node to a neighboring node is at most $2/m$, and

- any interfering transmitters are at least $(2\Delta + 2)/m$ distance away from the receiver.

We can schedule the c_4 lattices in a round-robin fashion and each lattice is active for $1/c_4$ amount of time.

D.4 Proof of Proposition 5.6.1

We only need to show that the probabilities of errors of all types will go to zero as $n \to \infty$. We first study Type-I errors that may occur in the odd super-frame. Fix a sending time slot t in the odd super-frame, $t = 1, ..., \lfloor n^d \rfloor$. Let $z_j(t)$ denote the number of nodes in the sending cell j at the sending time slot t, $j = 1, ..., g_1(n)$. Let $p_{\rm I}(t)$ be the probability that a Type-I error occurs at time slot t, i.e.,

$$p_{\rm I}(t) = \mathbf{P}[z_j(t) \geq 64 n^{(1-d)/3} \log n \text{ for any } j].$$

Equivalently, we can consider the experiment that we throw n balls into $g_1(n)$ urns with success probability $p = 1$ (see Lemma D.6). Since the average number of nodes in each sending cell is

$$\frac{n}{g_1(n)} \leq 32 n^{(1-d)/3} \log n, \text{ when } n \text{ is large,}$$

using part 4 of Lemma D.6, we have,

$$\begin{aligned} p_{\rm I}(t) &= \mathbf{P}[x_j \geq 64 n^{(1-d)/3} \log n \text{ for any } j] \\ &\leq O(1/n^3). \end{aligned}$$

Finally, the probability $p_{\rm I}$ that Type-I errors occurs at any of the $\lfloor n^d \rfloor$ sending time slots in the odd super-frame is

$$p_{\rm I} \leq \lfloor n^d \rfloor p_{\rm I}(t) \leq O(1/n^2).$$

We next study errors of Type-II. Fix a packet k. Let $S(k)$ and $T(k)$ denote the source node and the destination node, respectively, of packet k. Let t denote

the sending time slot at which packet k is broadcasted into the system. Pick any mobile relay node $i \neq S(k), T(k)$. The probability that node i holds the packet k, i.e., node i resides in the same sending cell as the source node $S(k)$ at sending time slot t, is $1/g_1(n)$. Conditioned on the event that node i holds the packet k, for any receiving time slot t' in the even super-frame, the probability that node i captures the destination node $T(k)$ of packet k at time slot t', i.e., node i resides in the same *receiving* cell as the destination node $T(k)$ at time slot t', is $1/g_2(n)$. Note that there are $\lfloor n^d \rfloor$ receiving time slots and the distribution of the position of node i is independent across receiving time slots. Hence, the probability that node i captures the destination node $T(k)$ of packet k in any of the $\lfloor n^d \rfloor$ receiving time slots is

$$p = \frac{1}{g_1(n)} \left[1 - \left(1 - \frac{1}{g_2(n)} \right)^{\lfloor n^d \rfloor} \right]. \tag{D.5}$$

Since $d < 1$ and

$$\frac{\lfloor n^d \rfloor}{g_2(n)} \to 0, \text{ as } n \to \infty,$$

it is easy to show that,

$$\frac{1 - \left(1 - \frac{1}{g_2(n)} \right)^{\lfloor n^d \rfloor}}{\frac{\lfloor n^d \rfloor}{g_2(n)}} \to 1, \text{ as } n \to \infty.$$

Hence, when n is large,

$$p \geq \frac{2}{3} \frac{\lfloor n^d \rfloor}{g_1(n) g_2(n)}.$$

Further, there are $(n-2)$ nodes (other than $S(k)$ and $T(k)$) that can potentially serve as mobile relays for packet k, and the distribution of the positions of these $(n-2)$ nodes are again independent from each other. Let $p_{\mathrm{II}}(k)$ be the probability that a Type-II error occurs for packet k, i.e., $p_{\mathrm{II}}(k)$ is the probability that none of these $(n-2)$ nodes capture the destination node $T(k)$ of packet k in any of the $\lfloor n^d \rfloor$ receiving time slots. Equivalently, we can consider the experiment that we throw $(n-2)$ balls into one urn with success probability p given by (D.5). When n is large, the average number of balls in the urn is

$$(n-2)p \geq \frac{1}{2} \frac{n^{(1+d)}}{g_1(n) g_2(n)} \geq 8 \log n.$$

Hence, by part 1 of Lemma D.6, the probability $p_{\mathrm{II}}(k)$ that the urn is empty is

$$p_{\mathrm{II}}(k) \leq O(1/n^3).$$

Since there are a total of $n\lfloor n^d \rfloor$ distinct packets in every two super-frames, the probability p_{II} that Type-II errors occur for any of these packets is

$$p_{\mathrm{II}} \leq n\lfloor n^d \rfloor O(1/n^3) \leq O(1/n).$$

Hence, with probability $(1 - p_{\mathrm{I}} - p_{\mathrm{II}})$ approaching one as $n \to \infty$, each packet will have at least one opportunity to be carried into the same receiving cell as its destination node. We now show that these packets can then be delivered successfully to their destination nodes by eliminating Type-III errors with high probability. Let p_{III} denote the probability that Type-III errors occur at any of the receiving cells in any of the $\lfloor n^d \rfloor$ receiving time slots. We need to specify how to schedule the hop-by-hop transmissions from the mobile relays to the destination nodes within each receiving cell. Fix a receiving time slot t, and fix a receiving cell j. Let $\mathcal{Y}_j(t)$ denote the set of packets that meet the criteria for capture in the receiving cell j at receiving time slot t. We will refer to these packets in the set $\mathcal{Y}_j(t)$ as the *active packets* in receiving cell j at time slot t. We divide the receiving cell j into $g_3(n) = \lfloor \left(\frac{n^{(2-2d)/3}}{8 \log n} \right)^{\frac{1}{2}} \rfloor^2$ *mini-cells* (in $\sqrt{g_3(n)}$ rows and $\sqrt{g_3(n)}$ columns, see Fig. 5.5). Each mini-cell is a square of area $1/(g_2(n)g_3(n))$. By Lemma 5.6.1, there exists a scheduling scheme where each mini-cell can be active for $\frac{1}{c_4}$ amount of time. When each mini-cell is active, it forwards an active packet (or a part of the packet) to one other node in the neighboring mini-cell. If the destination of the active packet is in the neighboring cell, the packet is forwarded directly to the destination node. The active packets from each mobile relay are first forwarded towards neighboring cells along the X-axis, then to their destination nodes along the Y-axis (see Fig. 5.5).

The above scheduling scheme can successfully forward all active packets in $\mathcal{Y}_j(t)$ from the mobile relays to the destination nodes in the same receiving cell provided that:

- Each mini-cell contains at least one node. Hence, each node can always find some node in the neighboring mini-cell to serve as static relays.
- The number of active packets that go through any mini-cell is bounded by $4096n^{(1-d)/3}\log^{3/2} n$. Because each packet is of length $\frac{W}{4096c_4 n^{(1-d)/3}\log^{3/2} n}$, each mini-cell thus only needs to be active for at most $\frac{1}{c_4}$ amount of time, which is always possible by Lemma 5.6.1[1].

In order to show that p_{III} (the probability of Type-III errors) goes to zero as $n \to \infty$, we only need to show that, with probability approaching one, both of the above conditions will hold for all receiving cells and for all receiving time slots. To show this, we will take four steps.

Step 1: We first bound the number of nodes in any mini-cell. Fix a receiving time slot t. Note that the average number of nodes in each mini-cell is

$$\frac{n}{g_2(n)g_3(n)} \geq 8\log n.$$

Let $p^{\text{a}}_{\text{III}}(t)$ be the probability that any of the $g_2(n)g_3(n)$ mini-cells in the network are empty at receiving time slot t. Equivalently, we can consider the experiment that we throw n balls into $g_2(n)g_3(n)$ urns with success probability $p = 1$. Then, by part 1 of Lemma D.6,

$$p^{\text{a}}_{\text{III}}(t) = O(1/n^3).$$

Hence, the probability $p^{\text{a}}_{\text{III}}$ that any mini-cells are empty in any of the $\lfloor n^d \rfloor$ receiving time slots is

$$p^{\text{a}}_{\text{III}} \leq \lfloor n^d \rfloor O(1/n^3) = O(1/n^2).$$

Step 2: We next bound the number of nodes in each receiving cell. Fix a receiving time slot t. Let $z_j(t)$ be the number of nodes in the receiving cell j. Then

$$\mathbf{E}[z_j(t)] = \frac{n}{g_2(n)} \leq 2n^{\frac{2-2d}{3}}, \text{ when } n \text{ is large.}$$

[1]An assumption we have used here is the *separation of time scales*, i.e., we assume that radio transmissions can be scheduled at a time scale much faster than that of node mobility. Hence, each packet can be divided into many smaller pieces and the transmissions of different pieces can be pipelined to achieve maximum throughput [71]. We also assume that the overhead of dividing a packet into many smaller pieces is negligible.

Let $p_{\rm III}^{\rm b}(t)$ be the probability that $z_j(t) \geq 4n^{\frac{2-2d}{3}}$ for any j. Equivalently, we can consider the experiment that we throw n balls into $g_2(n)$ urns with success probability $p = 1$. Then, by part 4 of Lemma D.6,

$$p_{\rm III}^{\rm b}(t) = O(1/n^3).$$

Let $p_{\rm III}^{\rm b}$ denote the probability that the number of nodes in any of the $g_2(n)$ receiving cells at any of the $\lfloor n^d \rfloor$ receiving time slots is greater than $4n^{\frac{2-2d}{3}}$. Then,

$$p_{\rm III}^{\rm b} \leq \lfloor n^d \rfloor p_{\rm III}^{\rm b}(t) \leq O(1/n^2).$$

Step 3: We shall show that, with probability approaching one as $n \to \infty$, the number of active packets that go through any mini-cell *along the X-axis* is bounded by $2048n^{(1-d)/3} \log^{3/2} n$. Towards this end, we first bound the number of active packets that each mobile relay node may carry at each receiving time slot. Fix a receiving time slot t and a receiving cell j. As in Step 2, we use $z_j(t)$ to denote the number of nodes in the receiving cell j at time slot t. *We shall condition the following discussion on the event that* $z_j(t) = m$. For each mobile node i among the m nodes in receiving cell j at time slot t, let $x_{ij}(t)$ denote the number of *active packets* that are held by mobile relay node i, i.e., these packets are the ones whose destination nodes are also in the receiving cell j at time slot t. Pick any other node i' that is also in the receiving cell j at time slot t. Note that, for a given packet k destined to node i', with probability $\frac{1}{g_1(n)}$ node i holds the packet k, i.e., with probability $\frac{1}{g_1(n)}$ node i was in the same sending cell as the source node of packet k when the packet k was broadcast into the system. Further, conditioned on the event that $z_j(t) = m$, the event that node i holds a packet k towards node i' is independent of the event that node i holds another packet h towards node i'' when $(k, i') \neq (h, i'')$. Since there are $\lfloor n^d \rfloor$ packets destined to each of the $(m-1)$ nodes (other than node i) in the receiving cell j, we have

$$\mathbf{E}[x_{ij}(t) | z_j(t) = m] = (m-1)\lfloor n^d \rfloor \frac{1}{g_1(n)}.$$

We now consider the probability that $x_{ij}(t) \geq 256 \log n$ for a given node i in a given receiving cell j. Note that if $m \leq 4n^{(2-2d)/3}$, we have

$$\mathbf{E}[x_{ij}(t)|z_j(t) = m] \leq 128 \log n, \text{ when } n \text{ is large.}$$

Conditioned on the event that $z_j(t) = m$, we can equivalently consider the experiment that we throw $(m-1)\lfloor n^d \rfloor$ balls into one urn with success probability $p = 1/g_1(n)$. Hence, by part 3 of Lemma D.6, the conditional probability that $x_{ij}(t) \geq 256 \log n$ satisfies

$$\mathbf{P}[x_{ij}(t) \geq 256 \log n|z_j(t) = m] \leq O(1/n^3), \text{ if } m \leq 4n^{(2-2d)/3}.$$

Fix a receiving time slot t. Let $p^{\mathrm{c}}_{\mathrm{III}}(t)$ be the probability that $x_{ij}(t) \geq 256 \log n$ for any node $i = 1, ..., z_j(t)$ in any receiving cell $j = 1, ..., g_2(n)$. Then

$$\begin{aligned}
p^{\mathrm{c}}_{\mathrm{III}}(t) &\leq \sum_{j=1}^{g_2(n)} \sum_{m=1}^{4n^{(2-2d)/3}} \mathbf{P}[x_{ij}(t) \geq 256 \log n \text{ for any } i|z_j(t) = m]\mathbf{P}[z_j(t) = m] \\
&\quad + \sum_{j=1}^{g_2(n)} \mathbf{P}[x_{ij}(t) \geq 256 \log n \text{ for any } i|z_j(t) > 4n^{(2-2d)/3}] \\
&\qquad \times \mathbf{P}[z_j(t) > 4n^{(2-2d)/3}] \\
&\leq g_2(n) \sum_{m=1}^{4n^{(2-2d)/3}} mO(1/n^3)\mathbf{P}[z_j(t) = m] + g_2(n)\mathbf{P}[z_j(t) > 4n^{(2-2d)/3}] \\
&\leq g_2(n)(4n^{(2-2d)/3})O(1/n^3)\mathbf{P}[z_j(t) \leq 4n^{(2-2d)/3}] + g_2(n)p^{\mathrm{b}}_{\mathrm{III}}(t) \\
&\leq O(1/n^2) + O(1/n^2) = O(1/n^2).
\end{aligned}$$

Hence, the probability $p^{\mathrm{c}}_{\mathrm{III}}$ that $x_{ij}(t) \geq 256 \log n$ for any i, j, t is

$$p^{\mathrm{c}}_{\mathrm{III}} \leq \lfloor n^d \rfloor p^{\mathrm{c}}_{\mathrm{III}}(t) \leq O(1/n).$$

Therefore, with probability approaching one as $n \to \infty$, each mobile relay node will serve no more than $256 \log n$ active packets in each receiving time slot. As presented earlier, these active packets will first be forwarded along the X-axis. Fix a receiving time slot t. We next bound the number of mobile relay nodes whose active packets need to go through a given mini-cell along the X-axis. Pick any mini-cell k in a given

receiving cell j, $k = 1, ..., g_3(n)$, $j = 1, ..., g_2(n)$. Let $Z^x_{j,k}(t)$ be the number of mobile relays in receiving cell j that reside at the same *row* with the mini-cell k, i.e., these mobile relays are the ones whose active packets need to go through mini-cell k along the X-axis. Note that the mean of $Z^x_{j,k}(t)$ is

$$n/(g_2(n)\sqrt{g_3(n)}) \leq 4n^{(1-d)/3}\sqrt{\log n}, \text{ when } n \text{ is large.}$$

Let $p^{\rm d}_{\rm III}(t)$ be the probability that $Z^x_{j,k}(t) \geq 8n^{(1-d)/3}\sqrt{\log n}$ for any mini-cell k in any receiving cell j. Equivalently, we can consider the experiment that we throw n balls into $g_2(n)\sqrt{g_3(n)}$ urns with success probability $p = 1$. Hence, by part 4 of Lemma D.6,

$$\begin{aligned} p^{\rm d}_{\rm III}(t) &= \mathbf{P}[Z^x_{j,k}(t) \geq 8n^{(1-d)/3}\sqrt{\log n} \text{ for any } k, j\ ] \\ &= O(\frac{1}{n^3}). \end{aligned}$$

Therefore, the probability $p^{\rm d}_{\rm III}$ that $Z^x_{j,k} \geq 8n^{(1-d)/3}\sqrt{\log n}$ for any mini-cell in any receiving time slot t is

$$p^{\rm d}_{\rm III} \leq \lfloor n^d \rfloor p^{\rm d}_{\rm III}(t) \leq O(1/n^2).$$

Combining the discussion above, with probability $1 - p^{\rm c}_{\rm III} - p^{\rm d}_{\rm III}$, for any given mini-cell k, there are at most $8n^{(1-d)/3}\sqrt{\log n}$ mobile relays whose active packets have to go through the mini-cell k along the X-axis, and each of these mobile relays will have at most $256 \log n$ active packets. Hence, with probability approaching one as $n \to \infty$, the number of active packets that go through any mini-cell along the X-axis in any receiving time slot t is at most

$$2048n^{(1-d)/3} \log^{3/2} n.$$

Step 4: Similar to Step 3, we can show that, with probability approaching one as $n \to \infty$, the number of active packets that go through any mini-cell *along the Y-axis* is bounded by $2048n^{(1-d)/3} \log^{3/2} n$. Towards this end, let $p^{\rm e}_{\rm III}$ denote the probability that, in any of the $\lfloor n^d \rfloor$ receiving time slots, any destination node

needs to receive more than $256 \log n$ active packets from the mobile relay nodes in the same receiving cell. Let $p_{\rm III}^{\rm f}$ denote the probability that, in any of the $\lfloor n^d \rfloor$ receiving time slots and for any given mini-cell k, the number of destination nodes whose active packets need to go through the mini-cell k along the Y-axis is greater than $8n^{(1-d)/3}\sqrt{\log n}$. Analogous to Step 3, we can show that $p_{\rm III}^{\rm e} = O(1/n)$ and $p_{\rm III}^{\rm f} = O(1/n^2)$. Hence, with probability approaching one as $n \to \infty$, the number of packets that go through any mini-cell along the Y-axis in any receiving time slot t is at most

$$2048n^{(1-d)/3} \log^{3/2} n.$$

Combining Step 1-4, with probability no less than

$$1 - (p_{\rm III}^{\rm b} + p_{\rm III}^{\rm c} + p_{\rm III}^{\rm d} + p_{\rm III}^{\rm e} + p_{\rm III}^{\rm f}),$$

the number of packets that have to go through any mini-cell at any receiving time slot t is less than

$$4096n^{(1-d)/3} \log^{3/2} n.$$

Hence, the probability of Type-III errors is bounded by

$$p_{\rm III} \leq p_{\rm III}^{\rm a} + p_{\rm III}^{\rm b} + p_{\rm III}^{\rm c} + p_{\rm III}^{\rm d} + p_{\rm III}^{\rm e} + p_{\rm III}^{\rm f} = O(1/n).$$

Proposition 5.6.1 then follows.

D.5 Proof of Proposition 5.7.1

We start from inequality (5.19). Since $\mathbf{E}[l_b] \leq \sqrt{c_5} n^{-1/2}$ for some positive constant c_t, we have,

$$\begin{aligned} \frac{1}{c_1 n \log n} \frac{(\lambda n T)^3}{\bar{D}(2\sum_{b=1}^{\lambda n T} \mathbf{E}[l_b])^2} &\geq \frac{1}{4c_1 n \log n} \frac{(\lambda n T)^3}{\bar{D}(\lambda n T)^2 c_5 n^{-1}} \\ &= \frac{1}{4c_1 c_5 \log n} \frac{\lambda n T}{\bar{D}}, \end{aligned}$$

and,

$$\begin{aligned}\frac{2\pi}{WTn}(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2 &\leq \frac{2\pi}{WTn}(\lambda nT)^2 c_5 n^{-1} \\ &= \frac{2\pi c_5}{W}\lambda^2 T.\end{aligned}$$

Substitute the above two inequalities into (5.19). Note that when $\bar{D} = o(n)$, the first term of (5.19) dominates the rest for large n. Hence, when n is large,

$$\begin{aligned}\frac{4c_3}{\Delta^2}WT\log n &\geq \frac{1}{2}\frac{1}{c_1 n\log n}\frac{(\lambda nT)^3}{\bar{D}(2\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2} \\ &\geq \frac{1}{8c_1c_5\log n}\frac{\lambda nT}{\bar{D}}.\end{aligned}$$

We can then solve for λ,

$$\lambda \leq \frac{\bar{D}\log^2 n}{n}\frac{32c_1c_3c_5W}{\Delta^2}.$$

D.6 Proof of Proposition 5.7.2

Since $h_b \leq c_6$ for some positive number c_6, similar to the initial steps of the proof of Proposition 5.5.1, we have,

$$\begin{aligned}\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h\right)^2 &\leq \left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}1\right)\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2\right) \\ &\leq \lambda nTc_6\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2,\end{aligned}$$

and,

$$\begin{aligned}\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2] &\geq \frac{1}{c_6\lambda nT}\mathbf{E}[\left(\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h\right)^2] \\ &\geq \frac{1}{c_6\lambda nT}\left(\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}S_b^h]\right)^2 \\ &\geq \frac{1}{c_6\lambda nT}\left(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2. \qquad \text{(D.6)}\end{aligned}$$

Substitute (5.17) and (D.6) into (5.9), we have,

$$\begin{aligned}\frac{4c_3}{\Delta^2}WT\log n &\geq \sum_{b=1}^{\lambda nT}\frac{\mathbf{E}[R_b]-1}{n}+\pi\mathbf{E}[\sum_{b=1}^{\lambda nT}\sum_{h=1}^{h_b}(S_b^h)^2] \\ &\geq \frac{1}{c_1 n\log n}\frac{(\sum_{b=1}^{\lambda nT}1)^3}{\bar{D}\left(\sum_{b=1}^{\lambda nT}(\mathbf{E}[l_b]+\frac{1}{n^2})\right)^2} \\ &\quad +\frac{\pi}{c_6\lambda nT}(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2-\lambda T.\end{aligned}$$

As in the proof of Proposition 5.5.1, the case with $\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\geq\lambda T/n$ will again prevail. Hence,

$$\begin{aligned}\frac{4c_3}{\Delta^2}WT\log n &\geq \frac{1}{c_1 n\log n}\frac{(\sum_{b=1}^{\lambda nT}1)^3}{\bar{D}\left(2\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b]\right)^2} \\ &\quad + \frac{\pi}{c_6\lambda nT}(\sum_{b=1}^{\lambda nT}\mathbf{E}[l_b])^2-\lambda T \\ &\geq 2\left[\frac{\pi}{4c_1c_6 n\log n\bar{D}}(\lambda nT)^2\right]^{1/2}-\lambda T \\ &= 2\sqrt{\frac{\pi}{4c_1c_6}\frac{\lambda^2nT^2}{\bar{D}\log n}}-\lambda T.\end{aligned}$$

When $\bar{D}=o(n)$, the first term dominates for large n. Hence, when n is large,

$$\begin{aligned}\frac{4c_3}{\Delta^2}WT\log n &\geq \sqrt{\frac{\pi}{4c_1c_6}\frac{\lambda^2nT^2}{\bar{D}\log n}} \\ \lambda^2 &\leq \frac{\bar{D}\log^3 n}{n}\frac{64c_1c_3^2c_6W^2}{\pi\Delta^4}.\end{aligned}$$

Appendix E: Supporting Results for Chapter 6

E.1 Proof of Proposition 6.3.1

The proof of part a) is quite standard (see, for example, Theorem 3.2.8 in [132, p44]). In fact, let $\vec{x^*}$ denote the optimal solution to the primal problem (6.2), and let $\vec{r}^*$ denote the corresponding vector of link rates that satisfies (6.6). It is easy to verify that, given $\vec{q}$,

$$\max_{0\leq x_s\leq M_s, \vec{r}\in \mathrm{Co}(\mathcal{R})} L(\vec{x},\vec{r},\vec{q}) \geq L(\vec{x^*},\vec{r}^*,\vec{q}) \geq \sum_{s=1}^{S} U_s(x_s^*) \text{ for all } \vec{q}\geq 0 \ .$$

To prove part a), we only need to find Lagrange multipliers $\vec{q}\geq 0$ such that

$$\max_{0\leq x_s\leq M_s, \vec{r}\in \mathrm{Co}(\mathcal{R})} L(\vec{x},\vec{r},\vec{q}) = \sum_{s=1}^{S} U_s(x_s^*).$$

Towards this end, let $\vec{b}=[b_l, l\in\mathcal{L}]$ and let

$$\begin{aligned} G(\vec{b}) = \max_{0\leq x_s\leq M_s} \quad & \sum_{s=1}^{S} U_s(x_s) \\ \text{subject to} \quad & \sum_{s=1}^{S} H_s^l x_s \leq r_l + b_l \text{ for all } l\in\mathcal{L} \\ \text{and} \quad & [r_l]\in \mathrm{Co}(\mathcal{R}). \end{aligned} \tag{E.1}$$

The original problem (6.2) corresponds to $\vec{b}=0$ and hence $G(0)=\sum_{s=1}^{S} U_s(x_s^*)$. It is easy to show that $G(\vec{b})$ is a concave function of $\vec{b}$. Hence, by Theorem 3.1.8 of [132, p36], there exists a subgradient $\vec{q_0}$ of $G(\vec{b})$ at $\vec{b}=0$. We now show that $\vec{q_0}$ is the desired Lagrange multipliers. For any $\vec{b}\geq 0$, by the concavity of $G(\vec{b})$, we have

$$G(\vec{b}) \leq G(0) + \vec{q_0}^{\mathrm{tr}}\vec{b},$$

where $[\cdot]^{\mathrm{tr}}$ denotes the transpose. Further, by the definition of $G(\vec{b})$ in (E.1),

$$G(0) \leq G(\vec{b}) \text{ for all } \vec{b}\geq 0.$$

Hence, for any $\vec{b}\geq 0$,

$$G(0) \geq G(\vec{b}) - \vec{q_0}^{\mathrm{tr}}\vec{b} \geq G(0) - \vec{q_0}^{\mathrm{tr}}\vec{b},$$

and we have,

$$\vec{q_0}^{\rm tr}\vec{b} \geq 0 \text{ for all } \vec{b} \geq 0.$$

This implies that $\vec{q_0} \geq 0$. Next, for any $\vec{x}$ such that $x_s \leq M_s$ for all s, and for any $\vec{r} \in \text{Co}(\mathcal{R})$, if we let $\vec{g}(\vec{x},\vec{r}) = \left[\sum_{s=1}^{S} H_s^l x_s - r_l, l \in \mathcal{L}\right]$, then $(\vec{x},\vec{r})$ is a feasible point in the problem (E.1) with $\vec{b} = \vec{g}(\vec{x},\vec{r})$. Hence, using the concavity of $G(\vec{b})$ again, we have

$$\begin{aligned}
\sum_{s=1}^{S} U_s(x_s) &\leq G(\vec{g}(\vec{x},\vec{r})) \\
&\leq G(0) + \vec{q_0}^{\rm tr}\vec{g}(\vec{x},\vec{r}) \\
&= \sum_{s=1}^{S} U_s(x_s^*) + \vec{q_0}^{\rm tr}\vec{g}(\vec{x},\vec{r}). \qquad \text{(E.2)}
\end{aligned}$$

Choosing $\vec{x} = \vec{x^*}$ and $\vec{r} = \vec{r}^*$, we have

$$\sum_{s=1}^{S} U_s(x_s^*) \leq \sum_{s=1}^{S} U_s(x_s^*) + \vec{q_0}^{\rm tr}\vec{g}(\vec{x^*},\vec{r}^*),$$

i.e.,

$$\vec{q_0}^{\rm tr}\vec{g}(\vec{x^*},\vec{r}^*) \geq 0.$$

However, since $\vec{g}(\vec{x^*},\vec{r}^*) \leq 0$ and $\vec{q_0} \geq 0$, we must have

$$\vec{q_0}^{\rm tr}\vec{g}(\vec{x^*},\vec{r}^*) = 0.$$

Finally, using (E.2) again, we obtain

$$\begin{aligned}
L(\vec{x},\vec{r},\vec{q_0}) &= \sum_{s=1}^{S} U_s(x_s) - \vec{q_0}^{\rm tr}\vec{g}(\vec{x},\vec{r}) \\
&\leq \sum_{s=1}^{S} U_s(x_s^*) = \sum_{s=1}^{S} U_s(x_s^*) - \vec{q_0}^{\rm tr}\vec{g}(\vec{x^*},\vec{r}^*) \\
&= L(\vec{x^*},\vec{r}^*,\vec{q_0}).
\end{aligned}$$

Hence

$$\max_{0\leq x_s\leq M_s, \vec{r}\in \text{Co}(\mathcal{R})} L(\vec{x},\vec{r},\vec{q_0}) = L(\vec{x^*},\vec{r}^*,\vec{q_0}) = \sum_{s=1}^{S} U_s(x_s^*),$$

i.e., there is no duality gap.

Proof of part b): Let $\vec{x^*}$ denote the optimal solution to the primal problem (6.2), and let $\vec{r}^*$ denote the corresponding vector of link rates that satisfies (6.6). Note that $\vec{x^*}$ is unique due to the strict concavity of $U_s(x_s)$. Given $\vec{q}$, the points $(\vec{x}, \vec{r})$ that maximize

$$L(\vec{x}, \vec{r}, \vec{q}) = \sum_{s=1}^{S} U_s(x_s) - \sum_{l=1}^{L} q^l \left(\sum_{s=1}^{S} H_s^l x_s - r_l \right)$$

must have the same component $\vec{x}$ due to the strict concavity of $U_s(x_s)$. Hence, in order to show part (b), it suffices to show that for any $\vec{q} \in \Phi$, the following holds,

$$\max_{0 \le x_s \le M_s, \vec{r} \in \mathrm{Co}(\mathcal{R})} L(\vec{x}, \vec{r}, \vec{q}) = L(\vec{x^*}, \vec{r}^*, \vec{q}). \tag{E.3}$$

(Note that $\vec{q}$ may be different from $\vec{q}_0$ in the proof of part (a).) Towards this end, since $\vec{q} \in \Phi$, we have,

$$\begin{aligned} \sum_{s=1}^{S} U_s(x_s^*) &= D(\vec{q}) = \max_{0 \le x_s \le M_s, \vec{r} \in \mathrm{Co}(\mathcal{R})} L(\vec{x}, \vec{r}, \vec{q}) \\ &\ge \sum_{s=1}^{S} U_s(x_s^*) - \sum_{l=1}^{L} q^l \left(\sum_{s=1}^{S} H_s^l x_s^* - r_l^* \right). \end{aligned} \tag{E.4}$$

Hence,

$$\sum_{l=1}^{L} q^l \left(\sum_{s=1}^{S} H_s^l x_s^* - r_l^* \right) \ge 0.$$

However, since $\sum_{s=1}^{S} H_s^l x_s^* - r_l^* \le 0$ for all l and $\vec{q} \ge 0$, we must have

$$\sum_{l=1}^{L} q^l \left(\sum_{s=1}^{S} H_s^l x_s^* - r_l^* \right) = 0.$$

Substituting into (E.4), we have,

$$\max_{0 \le x_s \le M_s, \vec{r} \in \mathrm{Co}(\mathcal{R})} L(\vec{x}, \vec{r}, \vec{q}) = \sum_{s=1}^{S} U_s(x_s^*) = L(\vec{x^*}, \vec{r}^*, \vec{q}),$$

which shows (E.3). The result of part (b) then follows.

Proof of part c): (This part of the proof is analogous to that of Theorem 2.3 in [109, p26]). Let A denote the $L \times L$ diagonal matrix whose l-th diagonal element is

α_l^0. Let H denote the $L \times S$ matrix whose (l,s)-element is H_s^l. Then $||\vec{q}||_A = \vec{q}^{\,\mathrm{tr}} A^{-1} \vec{q}$. For any $\vec{q}^{\,*,0} \in \Phi$, by (6.13), we have

$$\begin{aligned} ||\vec{q}(t+1) - \vec{q}^{\,*,0}||_A &\leq ||\vec{q}(t) - \vec{q}^{\,*,0}||_A + 2h[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}}[\vec{q}(t) - \vec{q}^{\,*,0}] \\ &\quad + h^2[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}} A[H\vec{x}(t) - \vec{r}(t)]. \end{aligned} \tag{E.5}$$

Note that

$$D(\vec{q}(t)) = \sum_{s=1}^{S} U_s(x_s(t)) - [H^{\,\mathrm{tr}}\vec{q}(t)]^{\,\mathrm{tr}}\vec{x}(t) + \vec{r}^{\,\mathrm{tr}}(t)\vec{q}(t),$$

and

$$\begin{aligned} D(\vec{q}^{\,*,0}) &= \max_{0 \leq x_s \leq M_s} \left\{ \sum_{s=1}^{S} U_s(x_s) - (H^{\,\mathrm{tr}}\vec{q}^{\,*,0})^{\,\mathrm{tr}}\vec{x} \right\} + \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \vec{r}^{\,\mathrm{tr}}\vec{q}^{\,*,0} \\ &\geq \sum_{s=1}^{S} U_s(x_s(t)) - (H^{\,\mathrm{tr}}\vec{q}^{\,*,0})^{\,\mathrm{tr}}\vec{x}(t) + \vec{r}^{\,\mathrm{tr}}(t)\vec{q}^{\,*,0}. \end{aligned}$$

Hence,

$$D(\vec{q}^{\,*,0}) - D(\vec{q}(t)) \geq [H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}}[\vec{q}(t) - \vec{q}^{\,*,0}].$$

Substituting into (E.5), we have

$$\begin{aligned} &||\vec{q}(t+1) - \vec{q}^{\,*,0}||_A \\ \leq\ &||\vec{q}(t) - \vec{q}^{\,*,0}||_A + 2h[D(\vec{q}^{\,*,0}) - D(\vec{q}(t))] + h^2[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}} A[H\vec{x}(t) - \vec{r}(t)]. \end{aligned}$$

Fix $\eta > 0$. Let

$$\Phi(\eta) = \{\vec{q} | D(\vec{q}) \leq D(\vec{q}^{\,*,0}) + \eta\}. \tag{E.6}$$

Since both $\vec{x}(t)$ and $\vec{r}(t)$ are bounded, there exists $M < \infty$ such that

$$\max_{0 \leq x_s \leq M_s, \vec{r} \in \mathrm{Co}(\mathcal{R})} (H\vec{x} - \vec{r})^{\,\mathrm{tr}} A(H\vec{x} - \vec{r}) \leq M.$$

If we pick

$$h \leq \eta / M,$$

then as long as $\vec{q}(t) \notin \Phi(\eta)$, we have

$$||\vec{q}(t+1) - \vec{q}^{\,*,0}||_A \leq ||\vec{q}(t) - \vec{q}^{\,*,0}||_A - h\eta.$$

Hence, eventually, $\vec{q}(t)$ will enter the set $\Phi(\eta)$. On the other hand, if we pick

$$h \leq \eta/\sqrt{M},$$

then once $\vec{q}(t) \in \Phi(\eta)$, we have

$$\begin{aligned} \sqrt{||\vec{q}(t+1) - \vec{q}^{*,0}||_A} &\leq \sqrt{||\vec{q}(t) - \vec{q}^{*,0}||_A} + \sqrt{||\vec{q}(t+1) - \vec{q}(t)||_A} \\ &\leq \sqrt{||\vec{q}(t) - \vec{q}^{*,0}||_A} + \eta. \end{aligned} \tag{E.7}$$

Since the inequality (E.7) holds for any $\vec{q}^{*,0} \in \Phi \subset \Phi(\eta)$, it implies that

$$d(\vec{q}(t+1), \Phi) \leq d(\vec{q}(t), \Phi) + \eta,$$

where $d(\vec{q}, \Phi) = \min_{\vec{p} \in \Phi} \sqrt{||\vec{q} - \vec{p}||_A}$. Hence, if

$$h \leq \min\{\eta/M, \eta/\sqrt{M}\},$$

then there exists time T_0 such that

$$d(\vec{q}(t), \Phi) \leq \xi(\eta) \text{ for all } t \geq T_0,$$

where

$$\xi(\eta) = \max_{\vec{p} \in \Phi(\eta)} d(\vec{p}, \Phi) + \eta.$$

It is easy to show that, as $\eta \to 0$,

$$\xi(\eta) \to 0.$$

Hence, for any $\epsilon > 0$, we can pick η (and h) sufficiently small such that $\xi(\eta) < \epsilon$, i.e., there exists time T_0 such that

$$d(\vec{q}(t), \Phi) < \epsilon \text{ for all } t \geq T_0.$$

Finally, since the mapping from $\vec{q}(t)$ to $\vec{x}(t)$ is continuous, we can pick η (and h) sufficiently small such that

$$||\vec{x}(t) - \vec{x^*}|| < \epsilon \text{ for all } t \geq T_0.$$

E.2 Proof of Proposition 6.3.2

We need the following assumption on the queueing discipline at each link. We assume that, when each link forwards data, data at smaller number of hops away from their source will have priority over data at a large number of hops away from their source. Hence, first-hop data will be forwarded before all second-hop data, then second-hop data will be forwarded before all third-hop data, and so on. One way to achieve such priority queueing is to have each link maintain separate queues for data at different number of hops away from their sources.

The above assumption allows us to study the queue lengths at all links in the network in an inductive manner. We first study all first-hop traffic in isolation because first-hop traffic takes precedence over all other traffic. Once we compute the contribution to the queue lengths by the first-hop traffic, we can then study the second-hop traffic in the network, and so on.

Let $x_s(t,k)$ denote the data from user s injected at time t to the link that is at the k-th hop from the source of user s (let $x_s(t,k)=0$ if data of user s travels at most k_0 hops, and $k > k_0$). Let $H_s^l(k)=1$, if link l is at the k-th hop from user s, and let $H_s^l(k)=0$, otherwise. Let $A^l(t,k)$ denote the amount of data injected to link l at time t by all first-hop through k-th hop traffic, i.e.,

$$A^l(t,k) = \sum_{s=1}^{S} \sum_{m=1}^{k} H_s^l(m) x_s(t,m).$$

Let $Q^l(t,k)$ denote the queue length at link l contributed by all first-hop through k-th hop traffic. Applying Loynes' formula, we have

$$Q^l(t,k) = \max_{0\le t'\le t} \left[\sum_{u=t-t'}^{t} A^l(u,k) - \sum_{u=t-t'}^{t} r_l(u) \right].$$

We now use induction to show that the queue lengths at all links are bounded. The induction hypothesis is as follows.

The Induction Hypothesis:

Fix k.

- For each user s, there exists a positive constant $M_s(k)$ such that

$$\sum_{u=t_0}^{t_0+t} x_s(u,k) \leq \sum_{u=t_0}^{t_0+t} x_s(u) + M_s(k), \text{ for all } t_0 \text{ and } t. \tag{E.8}$$

- The queue length $Q^l(t,k)$ at link l contributed by all first-hop through k-th hop traffic is bounded for all t.

We first show that the inequality (E.8) implies the second part of the induction hypothesis. Assume that $\alpha_l = h\alpha_l^0$. By Proposition 6.3.1, there exists some $h_0 > 0$ such that for all $h < h_0$ and any initial implicit costs $\vec{q}(0)$, there exists a time T_0 such that

$$d(\vec{q}(t), \Phi) \leq 1 \text{ for all } t \geq T_0.$$

Hence, $\vec{q}(t)$ is bounded for all t. Since

$$q^l(t+1) \geq q^l(t) + \alpha_l(\sum_{s=1}^{S} H_s^l x_s(t) - r_l(t)).$$

we have,

$$\sum_{u=t_0}^{t_0+t} \sum_{s=1}^{S} H_s^l x_s(u) - \sum_{u=t_0}^{t_0+t} r_l(u) \leq \frac{1}{\alpha_l} \left[q^l(t_0+t+1) - q^l(t_0) \right].$$

Hence, the left hand side is bounded from above for all t_0 and t. Let $M^l(0)$ be this upper bound. Using (E.8), we then have,

$$\begin{aligned}
Q^l(t,k) &= \max_{0\leq t'\leq t} \left[\sum_{u=t-t'}^{t} A^l(u,k) - \sum_{u=t-t'}^{t} r_l(u) \right] \\
&= \max_{0\leq t'\leq t} \left[\sum_{u=t-t'}^{t} \sum_{m=1}^{k} \sum_{s=1}^{S} H_s^l(m) x_s(u,m) - \sum_{u=t-t'}^{t} r_l(u) \right] \\
&\leq \max_{0\leq t'\leq t} \left[\sum_{u=t-t'}^{t} \sum_{s=1}^{S} H_s^l x_s(u) - \sum_{u=t-t'}^{t} r_l(u) + \sum_{m=1}^{k} \sum_{s=1}^{S} H_s^l(m) M_s(m) \right] \\
&\leq M^l(0) + \sum_{m=1}^{k} \sum_{s=1}^{S} H_s^l(m) M_s(m).
\end{aligned}$$

Hence, $Q^l(t,k)$ is bounded for all t.

We now use induction to show that the first part of the induction hypothesis, i.e., the inequality (E.8), holds for all k. We first consider the case $k = 1$, i.e., the first-hop traffic only. Since

$$x_s(t, 1) = x_s(t),$$

the inequality (E.8) trivially holds. The second part of the induction hypothesis then follows (for $k = 1$).

Next, assume that the induction hypothesis holds for $1, 2, ..., k-1$. Let $M(k-1)$ be the upper bound for $Q^l(t, k-1)$ for all t and l, i.e.,

$$M(k-1) = \sup_t \max_l Q^l(t, k-1).$$

We now consider the contribution by the k-th hop traffic. Note that

$$\sum_{u=t_0}^{t_0+t} x_s(u, k) \leq \sum_{u=t_0}^{t_0+t} x_s(u, k-1) + M(k-1),$$

where the first term on the right hand side corresponds to contribution from $(k-1)$-th hop traffic of user s, and the second term corresponds to the maximum amount of backlog at time t_0. Since the inequality (E.8) holds for $(k-1)$, we have,

$$\sum_{u=t_0}^{t_0+t} x_s(u, k) \leq \sum_{u=t_0}^{t_0+t} x_s(u) + M_s(k-1) + M(k-1),$$

and hence the inequality (E.8) now holds for k. Again, by the discussion above, the second part of the induction hypothesis also holds for k.

Finally, let $\bar{L}$ denote the maximum number of hops of any users. Note that the overall queue length $Q^l(t)$ at link l is equal to $Q^l(t, \bar{L})$. Hence,

$$\sup_t Q^l(t) < +\infty \text{ for all } l \in \mathcal{L}.$$

E.3 Proof of Proposition 6.4.1

Define

$$V_q(\vec{q}) = \sum_{l=1}^{L} \frac{(q^l)^2}{2\alpha_l}.$$

We now show that $V_q(\cdot)$ is a Lyapunov function of the system. In fact, using (6.18), we have,

$$V_q(\vec{q}(t+1)) - V_q(\vec{q}(t)) \;\leq\; \sum_{l=1}^{L} q^l(t) \left[\sum_{s=1}^{S} H_s^l x_s - r_l(t)\right] + E_1(t),$$

where

$$E_1(t) \;=\; \frac{1}{2}\sum_{l=1}^{L} \alpha_l \left[\sum_{s=1}^{S} H_s^l x_s - r_l(t)\right]^2.$$

Since both x_s and $r_l(t)$ are bounded, $E_1(t)$ is bounded for all t. Hence,

$$V_q(\vec{q}(t+1)) - V_q(\vec{q}(t)) \;\leq\; \sum_{l=1}^{L} q^l(t) \left[\sum_{s=1}^{S} H_s^l x_s - r_l(t)\right] + E_1^0, \tag{E.9}$$

for some positive constant E_1^0. By assumption, $\vec{x}$ lies strictly inside $\gamma\Lambda$. Hence, there exists some $\epsilon \geq 0$ such that

$$(1+\epsilon)\vec{x} \in \gamma\Lambda,$$

i.e.,

$$\left[(1+\epsilon)\sum_{l=1}^{L} H_s^l x_s\,, \quad l \in \mathcal{L}\right] \in \gamma \mathrm{Co}(\mathcal{R}).$$

By the definition of the imperfect scheduling policy S_γ,

$$\sum_{l=1}^{L} q^l(t) r_l(t) \geq \gamma \max_{\vec{r}\geq 0, \vec{r}\in\mathcal{R}} \sum_{l=1}^{L} r_l q^l(t) = \gamma \max_{\vec{r}\geq 0, \vec{r}\in \mathrm{Co}(\mathcal{R})} \sum_{l=1}^{L} r_l q^l(t) \geq (1+\epsilon)\sum_{l=1}^{L} q^l(t) H_s^l x_s.$$

Substituting into (E.9), we have,

$$V_q(\vec{q}(t+1)) - V_q(\vec{q}(t)) \;\leq\; -\epsilon \sum_{l=1}^{L} q^l(t) H_s^l x_s + E_1^0.$$

By Theorem 2 of [113] (or Theorem 3 of [133]), the system in stable.

E.4 Proof of Proposition 6.4.3

Fix a positive number ϵ such that

$$\epsilon < \min_{l: q_I^{l,*} \neq 0} q_I^{l,*}.$$

Then there exists a time slot T_0 such that for all $t \geq T_0$

$$|x_s(t) - x_s^{*,I}| \leq \epsilon, \text{ and} \tag{E.10}$$

$$|q^l(t) - q_I^{l,*}| \leq \epsilon. \tag{E.11}$$

We shall first show that there exists a large enough T such that,

$$\sum_{l=1}^{L} q_I^{l,*} \frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) \leq \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} + O(\epsilon), \tag{E.12}$$

where we have used $O(\epsilon)$ to denote the class of functions $f(\epsilon)$ such that $\limsup_{\epsilon \to 0} f(\epsilon)/\epsilon < +\infty$. Towards this end, note that for each link $l \in \mathcal{L}$, there are two cases:

Case 1: If $q_I^{l,*} > 0$, then (E.11) implies that $q^l(t) > 0$ for all $t \geq T_0$. Hence, using (6.18), we have

$$\alpha_l \left(\sum_{s=1}^{S} H_s^l x_s(t) - r_l(t) \right) = q^l(t+1) - q^l(t) \text{ for all } t \geq T_0. \tag{E.13}$$

For any $T > 0$, summing (E.13) over $t = T_0, T_0 + 1, ..., T_0 + T$, we have,

$$\alpha_l \left| \sum_{t=T_0}^{T_0+T} \sum_{s=1}^{S} H_s^l x_s(t) - \sum_{t=T_0}^{T_0+T} r_l(t) \right| = \left| q^l(T_0 + T + 1) - q^l(T_0) \right| \leq q_I^{l,*} + \epsilon.$$

We can thus pick T large enough such that

$$\left| \frac{1}{T} \sum_{t=T_0}^{T_0+T} \sum_{s=1}^{S} H_s^l x_s(t) - \frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) \right| < \epsilon.$$

Using (E.10), we have

$$\frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) \leq \sum_{s=1}^{S} H_s^l x_s^{*,I} + O(\epsilon). \tag{E.14}$$

Multiplying both side of (E.14) by $q_I^{l,*}$ and using (E.11) again, we have

$$q_I^{l,*} \frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) \leq q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} + O(\epsilon). \tag{E.15}$$

Case 2: If $q_I^{l,*} = 0$, the inequality (E.15) also holds trivially.

Summing (E.15) over all $l \in \mathcal{L}$, we thus obtain (E.12). Now, since the utility function is logarithmic, we have,

$$x_s(t) = \min \left\{ \frac{w_s}{\sum_{l=1}^{L} H_s^l q^l(t)}, M_s \right\} \text{ for all } t.$$

Taking limits as $t \to \infty$, we have,

$$x_s^{*,I} = \min \left\{ \frac{w_s}{\sum_{l=1}^{L} H_s^l q_I^{l,*}}, M_s \right\}. \tag{E.16}$$

Let $J = \{s : x_s^{*,I} = M_s\}$, then we have,

$$\begin{aligned} & \sum_{l=1}^{L} q_I^{l,*} \frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) \\ \leq & \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,I} + O(\epsilon) \\ = & \sum_{s=1}^{S} x_s^{*,I} (\sum_{l=1}^{L} H_s^l q_I^{l,*}) + O(\epsilon) \\ = & \sum_{s \notin J} w_s + \sum_{s \in J} M_s (\sum_{l=1}^{L} H_s^l q_I^{l,*}) + O(\epsilon). \end{aligned} \tag{E.17}$$

Let $\vec{x}^{\,*,\gamma}$ denote the solution to the γ-reduced problem. Then, $\frac{\vec{x}^{\,*,\gamma}}{\gamma} \in \Lambda$ by definition. Hence, by the definition of the imperfect schedule policy S_γ, $r_l(t)$ must satisfies

$$\begin{aligned} \sum_{l=1}^{L} q^l(t) r_l(t) & \geq \gamma \sum_{l=1}^{L} q^l(t) \frac{\sum_{s=1}^{S} H_s^l x_s^{*,\gamma}}{\gamma} \\ & = \sum_{l=1}^{L} q^l(t) \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} \text{ for all } t. \end{aligned}$$

Using (E.11) again, we obtain,

$$\sum_{l=1}^{L} q_I^{l,*} r_l(t) \geq \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} - O(\epsilon) \text{ for all } t \geq T_0.$$

Substituting into (E.17), we have,

$$\begin{aligned}
& \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} \\
\leq & \sum_{l=1}^{L} q_I^{l,*} \frac{1}{T} \sum_{t=T_0}^{T_0+T} r_l(t) + O(\epsilon) \\
\leq & \sum_{s \notin J} w_s + \sum_{s \in J} M_s (\sum_{l=1}^{L} H_s^l q_I^{l,*}) + O(\epsilon).
\end{aligned}$$

Noting that

$$\sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} = \sum_{s=1}^{S} x_s^{*,\gamma} \sum_{l=1}^{L} H_s^l q_I^{l,*} = \sum_{s \notin J} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}} + \sum_{s \in J} x_s^{*,\gamma} \sum_{l=1}^{L} H_s^l q_I^{l,*},$$

we have,

$$\begin{aligned}
& \sum_{s=1}^{S} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}} \\
= & \sum_{s \notin J} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}} + \sum_{s \in J} \frac{w_s x_s^{*,\gamma}}{M_s} \\
= & \sum_{l=1}^{L} q_I^{l,*} \sum_{s=1}^{S} H_s^l x_s^{*,\gamma} - \sum_{s \in J} x_s^{*,\gamma} \sum_{l=1}^{L} H_s^l q_I^{l,*} + \sum_{s \in J} \frac{w_s x_s^{*,\gamma}}{M_s} \\
\leq & \sum_{s \notin J} w_s + \sum_{s \in J} (M_s - x_s^{*,\gamma}) (\sum_{l=1}^{L} H_s^l q_I^{l,*}) + \sum_{s \in J} \frac{w_s x_s^{*,\gamma}}{M_s} + O(\epsilon).
\end{aligned}$$

Since $x_s^{*,\gamma} \leq M_s$ and $\sum_{l=1}^{L} H_s^l q_I^{l,*} \leq w_s/M_s$ when $s \in J$, we have,

$$\begin{aligned}
& \sum_{s=1}^{S} \frac{w_s x_s^{*,\gamma}}{x_s^{*,I}} \\
\leq & \sum_{s \notin J} w_s + \sum_{s \in J} (M_s - x_s^{*,\gamma}) \frac{w_s}{M_s} + \sum_{s \in J} \frac{w_s x_s^{*,\gamma}}{M_s} + O(\epsilon) \\
= & \sum_{s=1}^{S} w_s + O(\epsilon).
\end{aligned}$$

Finally, let $\epsilon \to 0$. The result then follows.

E.5 Proof of Proposition 6.4.4

Let A denote the $L \times L$ diagonal matrix whose l-th diagonal element is α_l^0. Let H denote the $L \times S$ matrix whose (l,s)-element is H_s^l. Then $||\vec{q}||_A = \vec{q}^{\,\mathrm{tr}} A^{-1} \vec{q}$. By (6.18), we have

$$\begin{aligned} ||\vec{q}(t+1) - \vec{q}^{\,*,0}||_A &\leq ||\vec{q}(t) - \vec{q}^{\,*,0}||_A + 2h[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}}[\vec{q}(t) - \vec{q}^{\,*,0}] \\ &\quad + h^2[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}} A[H\vec{x}(t) - \vec{r}(t)]. \quad \text{(E.18)} \end{aligned}$$

Note that

$$\begin{aligned} D_\gamma(\vec{q}(t)) &= \sum_{s=1}^{S} U_s(x_s(t)) - [H^{\,\mathrm{tr}}\vec{q}(t)]^{\,\mathrm{tr}}\vec{x}(t) + \gamma \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \vec{r}^{\,\mathrm{tr}}\vec{q}(t) \\ &\leq \sum_{s=1}^{S} U_s(x_s(t)) - [H^{\,\mathrm{tr}}\vec{q}(t)]^{\,\mathrm{tr}}\vec{x}(t) + \vec{r}^{\,\mathrm{tr}}(t)\vec{q}(t), \end{aligned}$$

and

$$\begin{aligned} D(\vec{q}^{\,*,0}) &= \max_{0 \leq x_s \leq M_s} \left\{ \sum_{s=1}^{S} U_s(x_s) - [H^{\,\mathrm{tr}}\vec{q}^{\,*,0}]^{\,\mathrm{tr}}\vec{x} \right\} + \max_{\vec{r} \in \mathrm{Co}(\mathcal{R})} \vec{r}^{\,\mathrm{tr}}\vec{q}^{\,*,0} \\ &\geq \sum_{s=1}^{S} U_s(x_s(t)) - [H^{\,\mathrm{tr}}\vec{q}^{\,*,0}]^{\,\mathrm{tr}}\vec{x}(t) + \vec{r}^{\,\mathrm{tr}}(t)\vec{q}^{\,*,0}. \end{aligned}$$

Hence,

$$D(\vec{q}^{\,*,0}) - D_\gamma(\vec{q}(t)) \geq [H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}}[\vec{q}(t) - \vec{q}^{\,*,0}].$$

Substituting into (E.18), we have

$$\begin{aligned} &||\vec{q}(t+1) - \vec{q}^{\,*,0}||_A \\ \leq\ &||\vec{q}(t) - \vec{q}^{\,*,0}||_A + 2h[D(\vec{q}^{\,*,0}) - D_\gamma(\vec{q}(t))] + h^2[H\vec{x}(t) - \vec{r}(t)]^{\,\mathrm{tr}} A[H\vec{x}(t) - \vec{r}(t)] \end{aligned}$$

Fix $\eta > 0$. Let

$$\Phi_\gamma(\eta) = \{\vec{q} | D_\gamma(\vec{q}) \leq D(\vec{q}^{\,*,0}) + \eta\}. \quad \text{(E.19)}$$

Since both $\vec{x}(t)$ and $\vec{r}(t)$ are bounded, there exists $M < +\infty$ such that

$$\max_{0 \leq x_s \leq M_s, \vec{r} \in \mathrm{Co}(\mathcal{R})} (H\vec{x} - \vec{r})^{\,\mathrm{tr}} A(H\vec{x} - \vec{r}) \leq M.$$

If we pick

$$h \leq \eta/M,$$

then as long as $\vec{q}(t) \notin \Phi_\gamma(\eta)$, we have

$$||\vec{q}(t+1) - \vec{q}^{*,0}||_A \leq ||\vec{q}(t) - \vec{q}^{*,0}||_A - h\eta.$$

Hence, eventually, $\vec{q}(t)$ will enter the set $\Phi_\gamma(\eta)$. On the other hand, if we pick

$$h \leq \eta/\sqrt{M},$$

then once $\vec{q}(t) \in \Phi_\gamma(\eta)$, we have

$$\begin{aligned}\sqrt{||\vec{q}(t+1) - \vec{q}^{*,0}||_A} &\leq \sqrt{||\vec{q}(t) - \vec{q}^{*,0}||_A} + \sqrt{||\vec{q}(t+1) - \vec{q}(t)||_A} \\ &\leq \sqrt{||\vec{q}(t) - \vec{q}^{*,0}||_A} + \eta.\end{aligned}$$

Hence, if

$$h \leq \min\{\eta/M, \eta/\sqrt{M}\}.$$

then there exists a time T_0 such that

$$\sqrt{||\vec{q}(t) - \vec{q}^{*,0}||_A} \leq \xi(\eta) \text{ for all } t \geq T_0,$$

where

$$\xi(\eta) = \max_{\vec{p} \in \Phi_\gamma(\eta)} \sqrt{||\vec{p} - \vec{q}^{*,0}||_A} + \eta.$$

It is easy to show that, as $\eta \to 0$,

$$\xi(\eta) \to \max_{\vec{p} \in \Phi_\gamma} \sqrt{||\vec{p} - \vec{q}^{*,0}||_A}.$$

The result then follows.

E.6 Proof of Proposition 6.5.1

Define

$$\mathcal{V}(\vec{n}, \vec{q}) = (1+\epsilon)V_n(\vec{n}) + V_q(\vec{q}),$$

where

$$V_n(\vec{n}) = \sum_{s=1}^{S} \frac{w_s n_s^2}{2\lambda_s}, \quad V_q(\vec{q}) = \sum_{l=1}^{L} \frac{(q^l)^2}{2\alpha_l},$$

and ϵ is a positive constant to be chosen later. We shall show that $\mathcal{V}(\cdot,\cdot)$ is a Lyapunov function of the system. We begin with a few lemmas. The first two lemmas bound the changes in $V_n(\cdot)$.

Lemma E.7

$$\begin{aligned}
& \mathbf{E}[V_n(\vec{n}((k+1)T) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq & \sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(kT) \right] \left[\rho_s T - \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \right] \right. \\
& \left. -\frac{3w_s}{8\rho_s M_s} \int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \right\} + E_1, \qquad \text{(E.20)}
\end{aligned}$$

where E_1 is a finite positive constant.

Proof Over a small time interval δt, we have

$$\begin{aligned}
& \mathbf{E}\left[\frac{w_s}{2\lambda_s} \left[n_s^2(t+\delta t) - n_s^2(t) \right] |\vec{n}(t), \vec{q}(t) \right] \\
= & \frac{w_s}{2\lambda_s} \left\{ [(n_s(t)+1)^2 - n_s^2(t)]\lambda_s \delta t + [(n_s(t)-1)^2 - n_s^2(t)]\mu_s n_s(t) x_s(t) \delta t \right\} + o(\delta t) \\
= & \frac{w_s}{\lambda_s} [n_s(t)\lambda_s \delta t - n_s(t)\mu_s n_s(t) x_s(t) \delta t] + \frac{w_s}{2\lambda_s} [\lambda_s \delta t + \mu_s n_s(t) x_s(t) \delta t] + o(\delta t)
\end{aligned}$$

Let $\rho_s = \lambda_s/\mu_s$. We have,

$$\begin{aligned}
& \frac{\mathbf{E}[V_n(\vec{n}(t+\delta t)) - V_n(\vec{n}(t))|\vec{n}(t), \vec{q}(t)]}{\delta t} \\
\leq & \sum_{s=1}^{S} \left\{ \frac{w_s n_s(t)}{\lambda_s} [\lambda_s - \mu_s n_s(t) x_s(t)] + \frac{w_s}{2\lambda_s} [\lambda_s + \mu_s n_s(t) x_s(t)] \right\} + o(1) \\
= & \sum_{s=1}^{S} \left\{ \frac{w_s n_s(t)}{\rho_s} [\rho_s - n_s(t) x_s(t)] + \frac{w_s}{2} (1 + \frac{n_s(t) x_s(t)}{\rho_s}) \right\} + o(1) \qquad \text{(E.21)} \\
\leq & \sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(t) \right] [\rho_s - n_s(t) x_s(t)] \right. \\
& \left. + \left[\frac{w_s}{x_s(t)} - \sum_{l=1}^{L} H_s^l q^l(t) \right] [\rho_s - n_s(t) x_s(t)] \right. \qquad \text{(E.22)}
\end{aligned}$$

$$+w_s\left[\frac{n_s(t)}{\rho_s}-\frac{1}{x_s(t)}\right][\rho_s-n_s(t)x_s(t)] \tag{E.23}$$

$$+\frac{w_s}{2}(1+\frac{n_s(t)x_s(t)}{\rho_s})\Bigg\} \tag{E.24}$$

$$+o(1),$$

where $\rho_s=\lambda_s/\mu_s$. We shall bound the three terms (E.22-E.24). By (6.24),

$$\frac{w_s}{x_s(t)}=\max\left\{\sum_{l=1}^{L}H_s^l q^l(t),\frac{w_s}{M_s}\right\}.$$

Hence, the term (E.22) can be bounded by

$$\begin{aligned}
&\left[\frac{w_s}{x_s(t)}-\sum_{l=1}^{L}H_s^l q^l(t)\right][\rho_s-n_s(t)x_s(t)]\\
\leq\ &\left[\frac{w_s}{x_s(t)}-\sum_{l=1}^{L}H_s^l q^l(t)\right]\rho_s\\
\leq\ &\left[\frac{w_s}{M_s}-\sum_{l=1}^{L}H_s^l q^l(t)\right]^+\rho_s\\
\leq\ &\frac{w_s\rho_s}{M_s}.
\end{aligned} \tag{E.25}$$

For the term (E.23), note that

$$\begin{aligned}
&\left[\frac{n_s(t)}{\rho_s}-\frac{1}{x_s(t)}\right][\rho_s-n_s(t)x_s(t)]\\
=\ &-\frac{[\rho_s-n_s(t)x_s(t)]^2}{\rho_s x_s(t)}\\
\leq\ &-\frac{[\rho_s-n_s(t)x_s(t)]^2}{\rho_s M_s}.
\end{aligned}$$

Using

$$[\rho_s-n_s(t)x_s(t)]^2+\rho_s^2\geq\frac{n_s^2(t)x_s^2(t)}{2},$$

we have,

$$\begin{aligned}
&\left[\frac{n_s(t)}{\rho_s}-\frac{1}{x_s(t)}\right][\rho_s-n_s(t)x_s(t)]\\
\leq\ &-\frac{1}{\rho_s M_s}\left[\frac{n_s^2(t)x_s^2(t)}{2}-\rho_s^2\right]\\
=\ &-\left[\frac{n_s^2(t)x_s^2(t)}{2\rho_s M_s}-\frac{\rho_s}{M_s}\right].
\end{aligned} \tag{E.26}$$

Finally, for the last term (E.24), we have

$$\frac{n_s(t)x_s(t)}{2\rho_s} \leq \frac{n_s^2(t)x_s^2(t)}{8\rho_s M_s} + \frac{M_s}{2\rho_s}. \tag{E.27}$$

Substituting (E.25-E.27) back to (E.22-E.24), we have,

$$\begin{aligned}
&\frac{\mathbf{E}[V_n(\vec{n}(t+\delta t)) - V_n(\vec{n}(t))|\vec{n}(t), \vec{q}(t)]}{\delta t} \\
\leq\ & \sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(t) \right] [\rho_s - n_s(t)x_s(t)] \right. \\
& \left. + \frac{w_s\rho_s}{M_s} - w_s \left[\frac{n_s^2(t)x_s^2(t)}{2\rho_s M_s} - \frac{\rho_s}{M_s} \right] + w_s \left[\frac{1}{2} + \frac{n_s^2(t)x_s^2(t)}{8\rho_s M_s} + \frac{M_s}{2\rho_s} \right] \right\} + o(1) \\
=\ & \sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(t) \right] [\rho_s - n_s(t)x_s(t)] - \frac{3w_s n_s^2(t)x_s^2(t)}{8\rho_s M_s} \right. \\
& \left. + w_s \left[\frac{1}{2} + \frac{M_s}{2\rho_s} + \frac{2\rho_s}{M_s} \right] \right\} + o(1).
\end{aligned} \tag{E.28}$$

Integrating over $[kT, (k+1)T]$, and letting

$$E_1 = \sum_{s=1}^{S} w_s T \left[\frac{1}{2} + \frac{M_s}{2\rho_s} + \frac{2\rho_s}{M_s} \right],$$

the result (E.20) follows. ■

Lemma E.8

$$\begin{aligned}
&\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & \sum_{s=1}^{S} \left\{ \rho_s T \left[\sum_{l=1}^{L} H_s^l q^l(kT) \right] - w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \right. \\
& \left. + \frac{w_s}{8\rho_s M_s} \int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \right\} + E_2,
\end{aligned} \tag{E.29}$$

where E_2 is a finite positive constant.

Proof From (E.21),

$$\begin{aligned}
&\frac{\mathbf{E}[V_n(\vec{n}(t+\delta t)) - V_n(\vec{n}(t))|\vec{n}(t), \vec{q}(t)]}{\delta t} \\
\leq\ & \sum_{s=1}^{S} \left\{ \frac{w_s n_s(t)}{\rho_s} [\rho_s - n_s(t)x_s(t)] + \frac{w_s}{2}(1 + \frac{n_s(t)x_s(t)}{\rho_s}) \right\} + o(1)
\end{aligned}$$

$$\begin{aligned} &\leq \sum_{s=1}^{S}\left\{\frac{w_s}{x_s(t)}[\rho_s - n_s(t)x_s(t)] + \frac{w_s}{2}(1+\frac{n_s(t)x_s(t)}{\rho_s})\right\} + o(1) \\ &= \sum_{s=1}^{S}\left\{\left[\frac{w_s\rho_s}{x_s(t)} - w_s n_s(t)\right] + \frac{w_s}{2}(1+\frac{n_s(t)x_s(t)}{\rho_s})\right\} + o(1). \end{aligned}$$

By (6.24),

$$\frac{w_s\rho_s}{x_s(t)} = \rho_s \max\left\{\sum_{l=1}^{L} H_s^l q^l(t), \frac{w_s}{M_s}\right\} \leq \rho_s\left(\sum_{l=1}^{L} H_s^l q^l(t) + \frac{w_s}{M_s}\right).$$

Combining with (E.27), we have,

$$\begin{aligned} &\frac{\mathbf{E}[V_n(\vec{n}(t+\delta t)) - V_n(\vec{n}(t))|\vec{n}(t),\vec{q}(t)]}{\delta t} \\ \leq &\sum_{s=1}^{S}\left\{\left[\rho_s\sum_{l=1}^{L} H_s^l q^l(t) - w_s n_s(t)\right] + \frac{w_s n_s^2(t)x_s^2(t)}{8\rho_s M_s}\right. \\ &\left. + w_s\left[\frac{1}{2} + \frac{\rho_s}{M_s} + \frac{M_s}{2\rho_s}\right]\right\} + o(1). \end{aligned} \tag{E.30}$$

Integrating over $[kT,(k+1)T]$, and letting

$$E_2 = \sum_{s=1}^{S} w_s T\left[\frac{1}{2} + \frac{\rho_s}{M_s} + \frac{M_s}{2\rho_s}\right],$$

the result (E.29) then follows. ■

The next lemma bounds the change in $V_q(\cdot)$. For simplicity, we use the following matrix notation. Let A denote the $L \times L$ diagonal matrix whose l-th diagonal element is α_l. Let H denote the $L \times S$ matrix whose (l,s)-element is H_s^l. Further, let $X_s(t) = n_s(t)x_s(t)$ and let $\vec{X}(t) = [X_1(t),...,X_S(t)]$. Then

$$V_q(\vec{q}) = \frac{\vec{q}^{\,\mathrm{tr}} A^{-1}\vec{q}}{2},$$

and the update on the implicit costs (6.25) can be written as

$$\vec{q}((k+1)T) = \left[\vec{q}(kT) + A\left(H\int_{kT}^{(k+1)T}\vec{X}(t)dt - \vec{r}(kT)T\right)\right]^{+}. \tag{E.31}$$

Lemma E.9

$$\begin{aligned}
& \mathbf{E}[V_q(\vec{q}((k+1)T) - V_q(\vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & \vec{q}^{\,\mathrm{tr}}(kT)\left[H\int_{kT}^{(k+1)T} \mathbf{E}[\vec{X}(t)|\vec{n}(kT), \vec{q}(kT)]dt - \vec{r}(kT)T\right] \\
& + T\alpha_{\max}\bar{S}\bar{L}\sum_{s=1}^{S}\left[\int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt\right] + E_3, \quad \text{(E.32)}
\end{aligned}$$

where

$$\alpha_{\max} = \max_{l\in\mathcal{L}} \alpha^l, \bar{L} = \max_s \sum_{l=1}^{L} H_s^l, \bar{S} = \max_l \sum_{s=1}^{S} H_s^l,$$

and E_3 is a finite positive constant, .

Proof By (E.31),

$$\begin{aligned}
& V_q(\vec{q}((k+1)T) - V_q(\vec{q}(kT)) \\
\leq\ & \vec{q}^{\,\mathrm{tr}}(kT)\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt - \vec{r}(kT)T\right] \\
& + \frac{1}{2}\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt - \vec{r}(kT)T\right]^{\mathrm{tr}} A\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt - \vec{r}(kT)T\right] \\
\leq\ & \vec{q}^{\,\mathrm{tr}}(kT)\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt - \vec{r}(kT)T\right] \\
& + \left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt\right]^{\mathrm{tr}} A\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt\right] + T^2[\vec{r}(kT)]^{\,\mathrm{tr}} A\vec{r}(kT),
\end{aligned}$$

where $[\cdot]^{\,\mathrm{tr}}$ denotes the transpose. Let

$$\alpha_{\max} = \max_{l\in\mathcal{L}} \alpha^l, \bar{L} = \max_s \sum_{l=1}^{L} H_s^l, \bar{S} = \max_l \sum_{s=1}^{S} H_s^l.$$

Then, we have,

$$\begin{aligned}
& \left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt\right]^{\mathrm{tr}} A\left[H\int_{kT}^{(k+1)T} \vec{X}(t)dt\right] \\
=\ & \sum_{l=1}^{L} \alpha_l\left[\sum_{s=1}^{S} H_s^l \int_{kT}^{(k+1)T} n_s(t)x_s(t)dt\right]^2
\end{aligned}$$

$$
\begin{aligned}
&\leq \sum_{l=1}^{L} \alpha_l \left[\sum_{s=1}^{S} H_s^l\right] \left[\sum_{s=1}^{S} H_s^l \left(\int_{kT}^{(k+1)T} n_s(t)x_s(t)dt\right)^2\right] \\
&\leq \bar{S}\sum_{l=1}^{L} \alpha_l \left[\sum_{s=1}^{S} H_s^l \left(\int_{kT}^{(k+1)T} n_s(t)x_s(t)dt\right)^2\right] \\
&\leq T\bar{S}\sum_{l=1}^{L} \alpha_l \sum_{s=1}^{S} H_s^l \int_{kT}^{(k+1)T} n_s^2(t)x_s^2(t)dt \\
&= T\bar{S}\sum_{s=1}^{S} \left[\int_{kT}^{(k+1)T} n_s^2(t)x_s^2(t)dt\right] \left[\sum_{l=1}^{L} \alpha_l H_s^l\right] \\
&\leq T\alpha_{\max}\bar{S}\bar{L}\sum_{s=1}^{S} \left[\int_{kT}^{(k+1)T} n_s^2(t)x_s^2(t)dt\right].
\end{aligned}
$$

Letting

$$E_3 = \max_{\vec{r}\in \mathrm{Co}(\mathcal{R})} T^2 \vec{r}^{\,\mathrm{tr}} A \vec{r},$$

the result (E.32) then follows. ■

Proof [of Proposition 6.5.1] Multiply (E.29) by $\epsilon < 1$ and add to (E.20). We have

$$
\begin{aligned}
&(1+\epsilon)\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & \sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(kT)\right] \left[(1+\epsilon)\rho_s T - \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt\right] \right. \\
&\qquad -\epsilon w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
&\qquad \left. -\frac{w_s}{4\rho_s M_s} \int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \right\} + E_1 + E_2 \quad \text{(E.33)}
\end{aligned}
$$

Adding (E.32) to (E.33), and noting that

$$
\begin{aligned}
&\sum_{s=1}^{S} \left\{ \left[\sum_{l=1}^{L} H_s^l q^l(kT)\right] \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \right\} \\
=\ & \sum_{l=1}^{L} q^l(kT) \sum_{s=1}^{S} H_s^l \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
=\ & \vec{q}^{\,\mathrm{tr}}(kT) \left[H \int_{kT}^{(k+1)T} \mathbf{E}[\vec{X}(t)|\vec{n}(kT), \vec{q}(kT)]dt\right],
\end{aligned}
$$

we have

$$\begin{aligned}
& \mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) - \mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq & \sum_{s=1}^{S}\left[\sum_{l=1}^{L} H_s^l q^l(kT)\right](1+\epsilon)\rho_s T - \vec{q}^{\,\mathrm{tr}}(kT)\vec{r}(kT)T \\
& -\epsilon\sum_{s=1}^{S} w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
& -\sum_{s=1}^{S}\left[\frac{w_s}{4\rho_s M_s} - T\alpha_{\max}\bar{S}\bar{L}\right]\int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \qquad \text{(E.34)} \\
& +E_0,
\end{aligned}$$

where $E_0 = E_1 + E_2 + E_3$. If (6.26) is satisfied, then

$$T\alpha_{\max}\bar{S}\bar{L} \leq \frac{w_s}{4\rho_s M_s} \text{ for all } s.$$

Hence, the term in (E.34) is negative. By a rearrangement of the order of the summation, we have,

$$\begin{aligned}
& \mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) - \mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq & \; T\vec{q}^{\,\mathrm{tr}}(kT)\left[(1+\epsilon)H\vec{\rho} - \vec{r}(kT)\right] \\
& -\epsilon\sum_{s=1}^{S} w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt + E_0,
\end{aligned}$$

where $\vec{\rho} = [\rho_1, ..., \rho_s]$. By assumption, $\vec{\rho}$ lies strictly inside $\gamma\Lambda$. Hence, there exists some $\epsilon > 0$ such that $(1+2\epsilon)\vec{\rho} \in \gamma\Lambda$, i.e.,

$$(1+2\epsilon)H\vec{\rho} \in \gamma\mathrm{Co}(\mathcal{R}).$$

Use this value of ϵ in the definition of $\mathcal{V}(\cdot,\cdot)$. Further, by the definition of the imperfect scheduling policy S_γ,

$$\vec{q}^{\,\mathrm{tr}}(kT)\vec{r}(kT) \geq \gamma \max_{\vec{r}\in\Lambda} \vec{q}^{\,\mathrm{tr}}(kT)\vec{r} \geq (1+2\epsilon)\vec{q}^{\,\mathrm{tr}}(kT)H\vec{\rho}.$$

Hence,

$$\mathbf{E}[\mathcal{V}(\vec{n}((k+1)T), \vec{q}((k+1)T)) - \mathcal{V}(\vec{n}(kT), \vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)]$$

$$\begin{aligned}
&\leq -\epsilon T\vec{q}^{\,\mathrm{tr}}(kT)H\vec{\rho} - \epsilon\sum_{s=1}^{S} w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT),\vec{q}(kT)]dt + E_0 \\
&\leq -\epsilon T\vec{q}^{\,\mathrm{tr}}(kT)H\vec{\rho} - \epsilon'\sum_{s=1}^{S} w_s n_s(kT) + E_0
\end{aligned}$$

for some $\epsilon' > 0$. By Theorem 2 of [113] (or Theorem 3 of [133]), the result then follows. ■

E.7 Proof of Proposition 6.6.1

Recall that

$$Q_i = \sum_{j:(i,j)\in\mathcal{L}} q_{ij} + \sum_{j:(j,i)\in\mathcal{L}} q_{ji}$$

denote the total cost of the links that either start from, or end at node i. Define

$$\mathcal{V}(\vec{n},\vec{q}) = (1+\epsilon)V_n(\vec{n}) + V_q(\vec{q}),$$

where

$$V_n(\vec{n}) = \sum_{s=1}^{S} \frac{w_s n_s^2}{2\lambda_s}, \quad V_q(\vec{q}) = \frac{\sum_{i=1}^{N} Q_i^2}{\alpha}, \tag{E.35}$$

and ϵ is a positive constant to be chosen later. (Note that the definition of $V_q(\cdot)$ is different from that in the earlier proofs.) We shall show that $\mathcal{V}(\cdot,\cdot)$ is a Lyapunov function of the system. In fact, analogous to Lemmas E.7 and E.8, we can show that,

$$\begin{aligned}
&\mathbf{E}[V_n(\vec{n}((k+1)T) - V_n(\vec{n}(kT))|\vec{n}(kT),\vec{q}(kT)] \\
\leq\ & \sum_{s=1}^{S}\Bigg\{\left[2\sum_{(i,j)\in\mathcal{L}} H_s^{ij}\frac{Q_i(kT)+Q_j(kT)}{c_{ij}}\right] \\
&\qquad \times\left[\rho_s T - \int_{kT}^{(k+1)T}\mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT),\vec{q}(kT)]dt\right] \\
&\qquad -\frac{3w_s}{8\rho_s M_s}\int_{kT}^{(k+1)T}\mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT),\vec{q}(kT)]dt\Bigg\} + E_1
\end{aligned} \tag{E.36}$$

and

$$\begin{aligned}
&\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & \sum_{s=1}^{S} \Bigg\{ \rho_s T \left[2 \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{Q_i(kT) + Q_j(kT)}{c_{ij}} \right] \\
& \qquad - w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
& \qquad + \frac{w_s}{8\rho_s M_s} \int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \Bigg\} + E_2, \quad \text{(E.37)}
\end{aligned}$$

where E_1 and E_2 are finite positive constants. Multiply (E.37) by $\epsilon < 1$ and add to (E.36). We have

$$\begin{aligned}
&(1+\epsilon)\mathbf{E}[V_n(\vec{n}((k+1)T)) - V_n(\vec{n}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & \sum_{s=1}^{S} \Bigg\{ \left[2 \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{Q_i(kT) + Q_j(kT)}{c_{ij}} \right] \\
& \qquad \times \left[(1+\epsilon)\rho_s T - \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \right] \\
& \qquad - \epsilon w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
& \qquad - \frac{w_s}{4\rho_s M_s} \int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \Bigg\} + E_1 + E_2 \quad \text{(E.38)}
\end{aligned}$$

As in Lemma E.9, we shall prove the following lemma bounding the change in $V_q(\cdot)$.

Lemma E.10 *If $\alpha < 1/T$, then*

$$\begin{aligned}
&\mathbf{E}[V_q(\vec{q}((k+1)T)) - V_q(\vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & 2 \sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) \Bigg\{ \sum_{m:(i,m)\in\mathcal{L}} \sum_{s=1}^{S} H_s^{im} \int_{kT}^{(k+1)T} \frac{\mathbf{E}[n_s(t)x_s(kT)|\vec{n}(kT), \vec{q}(kT)]}{c_{im}} dt \\
& \quad + \sum_{m:(m,i)\in\mathcal{L}} \sum_{s=1}^{S} H_s^{mi} \int_{kT}^{(k+1)T} \frac{\mathbf{E}[n_s(t)x_s(kT)|\vec{n}(kT), \vec{q}(kT)]}{c_{mi}} dt \\
& \quad + \sum_{h:(j,h)\in\mathcal{L}} \sum_{s=1}^{S} H_s^{jh} \int_{kT}^{(k+1)T} \frac{\mathbf{E}[n_s(t)x_s(kT)|\vec{n}(kT), \vec{q}(kT)]}{c_{jh}} dt
\end{aligned}$$

$$+\sum_{h:(h,j)\in\mathcal{L}}\sum_{s=1}^{S}H_s^{hj}\int_{kT}^{(k+1)T}\frac{\mathbf{E}[n_s(t)x_s(kT)|\vec{n}(kT),\vec{q}(kT)]}{c_{hj}}dt-T\Bigg\}$$

$$+\alpha E_3\sum_{s=1}^{S}\left[\int_{kT}^{(k+1)T}\mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT),\vec{q}(kT)]dt\right]+E_4, \tag{E.39}$$

where E_3 and E_4 are positive constants.

Proof By the definition of the maximal matching $\mathcal{M}(kT)$, $(i,j)\in\mathcal{M}(kT)$ implies $q_{ij}(kT)\geq 1$. Further, since we can choose $\alpha<1/T$, the projection operator in (6.35) is not needed. Hence,

$$\begin{aligned}
&[Q_i((k+1)T)]^2-[Q_i(kT)]^2\\
=\;&2\alpha T\left[\sum_{j:(i,j)\in\mathcal{L}}q_{ij}(kT)+\sum_{j:(j,i)\in\mathcal{L}}q_{ji}(kT)\right]\\
&\times\left[\sum_{j:(i,j)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ij}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ij}}dt-\mathbf{I}_{\{(i,j)\in\mathcal{M}(kT)\}}\right)\right.\\
&\left.+\sum_{j:(j,i)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ji}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ji}}dt-\mathbf{I}_{\{(j,i)\in\mathcal{M}(kT)\}}\right)\right]\\
&+\alpha^2T^2\left[\sum_{j:(i,j)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ij}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ij}}dt-\mathbf{I}_{\{(i,j)\in\mathcal{M}(kT)\}}\right)\right.\\
&\left.+\sum_{j:(j,i)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ji}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ji}}dt-\mathbf{I}_{\{(j,i)\in\mathcal{M}(kT)\}}\right)\right]^2.
\end{aligned}$$

Substituting into (E.35) and rearranging the terms, we have,

$$\begin{aligned}
&V_q(\vec{q}((k+1)T))-V_q(\vec{q}(kT))\\
=\;&2\sum_{(i,j)\in\mathcal{L}}q_{ij}(kT)\left[\sum_{m:(i,m)\in\mathcal{L}}\sum_{s=1}^{S}H_s^{im}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{im}}dt\right.\\
&+\sum_{m:(m,i)\in\mathcal{L}}\sum_{s=1}^{S}H_s^{mi}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{mi}}dt\\
&\left.+\sum_{h:(j,h)\in\mathcal{L}}\sum_{s=1}^{S}H_s^{jh}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{jh}}dt\right.
\end{aligned}$$

$$+\sum_{h:(h,j)\in\mathcal{L}}\sum_{s=1}^{S}H_s^{hj}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{hj}}dt\Bigg]$$

$$-2T\sum_{(i,j)\in\mathcal{L}}q_{ij}(kT)\Bigg[\sum_{m:(i,m)\in\mathcal{L}}\mathbf{I}_{\{(i,m)\in\mathcal{M}(kT)\}}+\sum_{m:(m,i)\in\mathcal{L}}\mathbf{I}_{\{(m,i)\in\mathcal{M}(kT)\}}$$

$$+\sum_{h:(j,h)\in\mathcal{L}}\mathbf{I}_{\{(j,h)\in\mathcal{M}(kT)\}}+\sum_{h:(h,j)\in\mathcal{L}}\mathbf{I}_{\{(h,j)\in\mathcal{M}(kT)\}}\Bigg]$$

$$+E_4(k), \tag{E.40}$$

where

$$\begin{aligned}E_4(k) &= \alpha T^2\sum_{i=1}^{N}\\ &\Bigg[\sum_{j:(i,j)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ij}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ij}}dt-\mathbf{I}_{\{(i,j)\in\mathcal{M}(kT)\}}\right)\\ &+\sum_{j:(j,i)\in\mathcal{L}}\left(\frac{1}{T}\sum_{s=1}^{S}H_s^{ji}\int_{kT}^{(k+1)T}\frac{n_s(t)x_s(kT)}{c_{ji}}dt-\mathbf{I}_{\{(j,i)\in\mathcal{M}(kT)\}}\right)\Bigg]^2.\end{aligned}$$

Similar to the proof of Lemma E.9, we can show that

$$E_4(k)\le\alpha E_3\sum_{s=1}^{S}\left[\int_{kT}^{(k+1)T}n_s^2(t)x_s^2(t)dt\right]+E_5, \tag{E.41}$$

where E_3 and E_5 are positive constants. (Note that E_3 can be shown to only depend on the topology of the network.) Further, by the definition of the maximal matching $\mathcal{M}(kT)$,

$$\sum_{m:(i,m)\in\mathcal{L}}\mathbf{I}_{\{(i,m)\in\mathcal{M}(kT)\}}+\sum_{m:(m,i)\in\mathcal{L}}\mathbf{I}_{\{(m,i)\in\mathcal{M}(kT)\}}$$

$$+\sum_{h:(j,h)\in\mathcal{L}}\mathbf{I}_{\{(j,h)\in\mathcal{M}(kT)\}}+\sum_{h:(h,j)\in\mathcal{L}}\mathbf{I}_{\{(h,j)\in\mathcal{M}(kT)\}}\ge 1,$$

$$\text{for all } (i,j) \text{ such that } q_{ij}(kT)\ge 1.$$

Hence,

$$-q_{ij}(kT)\Bigg[\sum_{m:(i,m)\in\mathcal{L}}\mathbf{I}_{\{(i,m)\in\mathcal{M}(kT)\}}+\sum_{m:(m,i)\in\mathcal{L}}\mathbf{I}_{\{(m,i)\in\mathcal{M}(kT)\}}$$

$$+ \sum_{h:(j,h)\in\mathcal{L}} \mathbf{I}_{\{(j,h)\in\mathcal{M}(kT)\}} + \sum_{h:(h,j)\in\mathcal{L}} \mathbf{I}_{\{(h,j)\in\mathcal{M}(kT)\}} \Bigg] \leq -q_{ij}(kT) + 1,$$
$$\text{for all } (i,j) \in \mathcal{L}. \tag{E.42}$$

Substituting (E.41) and (E.42) into (E.40), the result (E.39) then follows with $E_4 = E_5 + 2LT$, where L is the total number of links. ■

Proof [**of Proposition 6.6.1**] Note that for all $a_s, s = 1, ..., S$,

$$\begin{aligned}
&\sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) \sum_{m:(i,m)\in\mathcal{L}} \sum_{s=1}^{S} \frac{H_s^{im} a_s}{c_{im}} \\
&= \sum_{s=1}^{S} a_s \sum_{(i,m)\in\mathcal{L}} H_s^{im} \frac{\sum_{j:(i,j)\in\mathcal{L}} q_{ij}(kT)}{c_{im}} \\
&= \sum_{s=1}^{S} a_s \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{\sum_{m:(i,m)\in\mathcal{L}} q_{im}(kT)}{c_{ij}}.
\end{aligned} \tag{E.43}$$

Similarly,

$$\begin{aligned}
\sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) \sum_{m:(m,i)\in\mathcal{L}} \sum_{s=1}^{S} \frac{H_s^{mi} a_s}{c_{mi}} &= \sum_{s=1}^{S} a_s \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{\sum_{m:(m,i)\in\mathcal{L}} q_{mi}(kT)}{c_{ij}} \\
\sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) \sum_{h:(j,h)\in\mathcal{L}} \sum_{s=1}^{S} \frac{H_s^{jh} a_s}{c_{jh}} &= \sum_{s=1}^{S} a_s \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{\sum_{h:(j,h)\in\mathcal{L}} q_{jh}(kT)}{c_{ij}} \\
\sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) \sum_{h:(h,j)\in\mathcal{L}} \sum_{s=1}^{S} \frac{H_s^{hj} a_s}{c_{hj}} &= \sum_{s=1}^{S} a_s \sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{\sum_{h:(h,j)\in\mathcal{L}} q_{hj}(kT)}{c_{ij}}.
\end{aligned} \tag{E.44}$$

Hence, by Lemma E.10,

$$\begin{aligned}
&\mathbf{E}[V_q(\vec{q}((k+1)T) - V_q(\vec{q}(kT))|\vec{n}(kT), \vec{q}(kT)] \\
\leq\ & 2\sum_{s=1}^{S} \left[\sum_{(i,j)\in\mathcal{L}} H_s^{ij} \frac{Q_i(kT) + Q_j(kT)}{c_{ij}} \right] \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)x_s(t)|\vec{n}(kT), \vec{q}(kT)]dt \\
& -2T \sum_{(i,j)\in\mathcal{L}} q_{ij}(kT) + \alpha E_3 \sum_{s=1}^{S} \left[\int_{kT}^{(k+1)T} \mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT), \vec{q}(kT)]dt \right] \\
& + E_4.
\end{aligned} \tag{E.45}$$

Adding (E.45) to (E.38), we have

$$\begin{aligned}
&\mathbf{E}[\mathcal{V}(\vec{n}((k+1)T),\vec{q}((k+1)T))-\mathcal{V}(\vec{n}(kT),\vec{q}(kT))|\vec{n}(kT),\vec{q}(kT)]\\
\leq\ & 2T(1+\epsilon)\sum_{s=1}^{S}\rho_s\left[\sum_{(i,j)\in\mathcal{L}}H_s^{ij}\frac{Q_i(kT)+Q_j(kT)}{c_{ij}}\right]-2T\sum_{(i,j)\in\mathcal{L}}q_{ij}(kT)\\
&-\epsilon\sum_{s=1}^{S}w_s\int_{kT}^{(k+1)T}\mathbf{E}[n_s(t)|\vec{n}(kT),\vec{q}(kT)]dt\\
&-\sum_{s=1}^{S}\left[\frac{w_s}{4\rho_s M_s}-\alpha E_3\right]\int_{kT}^{(k+1)T}\mathbf{E}[n_s^2(t)x_s^2(t)|\vec{n}(kT),\vec{q}(kT)]dt\\
&+E_7,
\end{aligned}\tag{E.46}$$

where $E_7=E_1+E_2+E_4$. When α is sufficiently small, the product term in (E.46) is negative. Further, by assumption, $[\rho_s]$ lies strictly inside $\frac{\Lambda}{2}$. Hence, there exists some positive number ϵ such that, for all node i,

$$(1+2\epsilon)\left[\sum_{j:(i,j)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{ij}\rho_s}{c_{ij}}+\sum_{j:(j,i)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{ji}\rho_s}{c_{ji}}\right]\leq 1/2.$$

Hence, applying (E.43-E.44) again on the inequality (E.46), we have,

$$\begin{aligned}
&\mathbf{E}[\mathcal{V}(\vec{n}((k+1)T),\vec{q}((k+1)T))-\mathcal{V}(\vec{n}(kT),\vec{q}(kT))|\vec{n}(kT),\vec{q}(kT)]\\
\leq\ & 2T\sum_{(i,j)\in\mathcal{L}}q_{ij}(kT)\left[\sum_{m:(i,m)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{im}(1+\epsilon)\rho_s}{c_{im}}+\sum_{m:(m,i)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{mi}(1+\epsilon)\rho_s}{c_{mi}}\right.\\
&\left.+\sum_{h:(j,h)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{jh}(1+\epsilon)\rho_s}{c_{jh}}+\sum_{h:(h,j)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{hj}(1+\epsilon)\rho_s}{c_{hj}}-1\right]\\
&-\epsilon\sum_{s=1}^{S}w_s\int_{kT}^{(k+1)T}\mathbf{E}[n_s(t)|\vec{n}(kT),\vec{q}(kT)]dt+E_7\\
\leq\ & -2T\epsilon\sum_{(i,j)\in\mathcal{L}}q_{ij}(kT)\left[\sum_{m:(i,m)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{im}\rho_s}{c_{im}}+\sum_{m:(m,i)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{mi}\rho_s}{c_{mi}}\right.\\
&\left.+\sum_{h:(j,h)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{jh}\rho_s}{c_{jh}}+\sum_{h:(h,j)\in\mathcal{L}}\sum_{s=1}^{S}\frac{H_s^{hj}\rho_s}{c_{hj}}\right]
\end{aligned}$$

$$-\epsilon \sum_{s=1}^{S} w_s \int_{kT}^{(k+1)T} \mathbf{E}[n_s(t)|\vec{n}(kT), \vec{q}(kT)]dt + E_7.$$

By Theorem 2 of [113] (or Theorem 3 of [133]), the result then follows. ■

VITA

VITA

Xiaojun Lin received his B.S. in Electronics and Information Systems from Zhongshan University, Guangzhou, China, in 1994, and his M.S. in Electrical and Computer Engineering from Purdue University, West Lafayette, Indiana, in 2000.

From 1995 to 1998, he worked as an I/T Specialist in IBM China. In 2002, he spent three months in IBM T.J. Watson Research Labs. Since 2000, he has been working towards the Ph.D. degree in Electrical and Computer Engineering at Purdue University, West Lafayette, Indiana. In 2005, he received the Journal of Communications and Networks Best Paper of the Year Award.

His research interests are in resource allocation, control, routing, security, and cross-layer design for wireless and wireline communication networks.