Optimization and Incentives in Communication Networks

Libin Jiang

Optimization and Incentives in Communication Networks

by

Libin Jiang

B.E. (University of Science and Technology of China) 2003
M.Phil. (Chinese University of Hong Kong) 2005

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy
in

Engineering - Electrical Engineering and Computer Sciences

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Jean Walrand, Chair
Professor Venkat Anantharam
Professor Pravin Varaiya
Professor Shachar Kariv

Fall 2009

Optimization and Incentives in Communication Networks

Abstract

Optimization and Incentives in Communication Networks

by

Libin Jiang

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Jean Walrand, Chair

Performance optimization of communication networks involves challenges at both the engineering level and the human level. In the first part of the dissertation, we study a network security game where strategic players choose their investments in security. Since a player's investment can reduce the propagation of computer viruses, a key feature of the game is the positive externality exerted by the investment. With selfish players, unfortunately, the overall network security can be far from optimum. First, we characterize the price of anarchy (POA) in the strategic-form game under an "effective-investment" model and a "bad-traffic" model, and give insight on how the POA depends on the network topology, the cost functions of the players, and their mutual influence (or externality). We show that the POA in general cannot be offset by the improvement of security technology. Second, in a repeated game, users have more incentive to cooperate. We characterize the socially best outcome that can be supported by the repeated game, as compared to the social optimum. We also introduce a Folk Theorem which only requires local punishments and rewards, but supports the same payoff region as the usual Folk Theorem. Finally, with a social planner who implements a due-care scheme which mandates the minimal investments, we study how the performance bound improves. Although our primary focus is Internet security, many results are generally applicable to games with positive externalities.

In the second part of the dissertation, we consider the problem of achieving the maximum throughput and utility in a class of networks with resource-sharing constraints. This is a classical problem which had lacked an efficient distributed solution. First, we propose a fully distributed scheduling algorithm that achieves the maximum throughput. Inspired by CSMA (Carrier Sense Multiple Access) which is widely deployed in today's wireless networks, our algorithm is simple, asynchronous and easy to implement. Second, using a novel maximal-entropy technique, we combine the CSMA scheduling algorithm with congestion control to approach the maximum utility. Also, we further show that CSMA scheduling is a modular MAC-layer algorithm that can work with other protocols in the transport layer and network layer. Third, for wireless networks where packet collisions are unavoidable, we establish a general analytical model and extend the above algorithms to that case.

Stochastic Processing Networks (SPNs) model manufacturing, communication, and service systems. In manufacturing networks, for example, service activities require parts and resources to produce other parts. SPNs are more general than queueing networks and pose novel challenges to throughput-optimum scheduling. In the third part of the dissertation, we proposes a "deficit maximum weight" (DMW) algorithm to achieve throughput optimality and maximize the net utility of the production in SPNs.

Professor Jean Walrand
Dissertation Committee Chair

To my loving parents Chongwen and Jinfeng.

Contents

List of Figures

Acknowledgements

First, I would like to thank my advisor Jean Walrand for his guidance, support, and many opportunities he has provided to me through the years. His teaching and advice have opened doors for me, helped me grow both in technical skills and intuition. The discussions with him have inspired many ideas in my research on network algorithms and network economics. His kindness, patience and enthusiasm are essential to my progress. It is a wonderful experience working with him.

Also, I thank my co-advisor Venkat Anantharam for many stimulating discussions on game theory as applied to the network security investment problem. I am always inspired by the breadth of his knowledge, his sharp insight, and the rigor he put into our work.

I thank Prof. Pravin Varaiya and Prof. Shachar Kariv for serving on my dissertation committee, and for their interest and invaluable suggestions on my research. Prof. Kariv also taught me a lively course on game theory in the Economics department.

I thank my fellow graduate students Assane Gueye, Jiwoong Lee, Nikhil Shetty and Christophe Choumert (in no particular order:)) for being such good companies in my journey in Berkeley, and alumni Shyam Parekh, Antonis Dimakis, Rajarshi Gupta, Teresa Tung, Jeonghoon Mo for their guidance and advice.

I am very grateful to my collaborators outside Berkeley, including Prof. Srikant and Dr. Jian Ni at UIUC, Devavrat Shah and Jinwoo Shin at MIT, and Prof. Minghua Chen at CUHK. I thank Prof. Michael Neely at USC for his discussions. Also, a special thank to Prof. Soung Chang Liew at CUHK who supervised my master thesis. Prof Liew has always cared about my study and life in Berkeley, and kept me informed of his research. In fact, his work on CSMA modeling has inspired my interest in designing CSMA-based scheduling algorithms.

This work is supported by NSF under Grant NeTS-FIND 0627161, MURI grant BAA 07-036.18 and Vodafone-US Foundation Fellowship.

Chapter 1

Optimization and Incentives

Communication networks are composed of distributed, interdependent and sometimes selfish entities. Performance optimization of these networks involves challenges at both the engineering level and the human level. At the engineering level, the design objective (or "social welfare") of a network is its overall throughput, delay, fairness and security, etc. To optimize the social welfare in large networks, high-performance distributed algorithms need to be designed. On the other hand, at the human level the network entities are often controlled by humans, each with his own individual objective (i.e., "individual welfare"). The important role of human incentives needs to be understood from a game-theoretic and economic perspective.

Although the problems at the engineering level and the human level seem quite different, some fascinating connections have been revealed in recent years. It is known to economists that by deploying suitable incentivizing mechanisms such as pricing, the social welfare can be achieved in a game with selfish players. This idea has been used in communication networks to "regulate" the actions of different network entities (devices) using suitable "incentives" (in the form of feedback signals), such that a global performance objective is optimized in a distributed manner. On the other hand, if the devices are controlled by selfish humans, and there is no suitable regulation or mechanism in place, the network performance

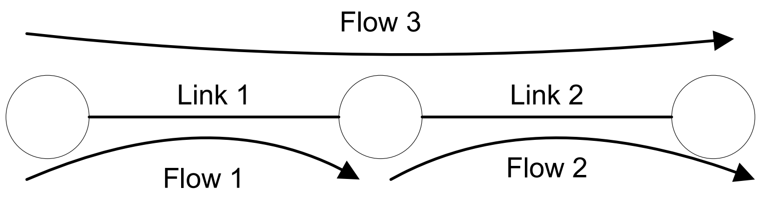


Figure 1.1: Global optimization

at the equilibrium is usually sub-optimum. Therefore, understanding and improving the performance in such scenarios becomes an important problem.

In this chapter, we give examples to illustrate the connections.

1.1 Global optimization and distributed algorithms

A well known global optimization problem in *both communication networks and economics* is a resource allocation problem. For example, consider three data flows, each going through a set of links (Fig. 1.1). Flow 1 goes through link 1, flow 2 goes through link 2, and flow 3 goes through both link 1 and link 2. Both link 1 and 2 have a capacity of 1. The rate at which flow i sends data is $x_i, i = 1, 2, 3$, and the total rate each link carries cannot exceed its capacity. That is, $x_1 + x_3 \leq 1$, $x_2 + x_3 \leq 1$. Each flow has an increasing and strictly concave "utility function" $U_i(x_i), i = 1, 2, 3$. The "congestion control problem" in communication networks is to determine the flow rates $\mathbf{x} := (x_1, x_2, x_3)$ such that the total utility is maximized. That is,

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i=1}^{3} U_i(x_i) \\ \text{s.t.} \quad & x_1 + x_3 \leq 1 \\ & x_2 + x_3 \leq 1. \end{aligned} \tag{1.1}$$

In economics, each link corresponds to a "resource", and each flow corresponds to an "activity". The activities consume different sets of resources as described by the above constraints. The solution of problem (1.1) is the optimum way to allocate the resources to the activities.

1.1.1 Distributed algorithms with economic interpretations

Two influential papers, [1] and [2], first described *distributed algorithms* in communication networks to solve problem (1.1) via a Lagrangian decomposition method. Although the method is standard in optimization theory, its application to communication networks has made a major impact on the understanding and design of network protocols.

The method is as follows. To solve (1.1), we form a Lagrangian

$$\begin{aligned}\mathcal{L}(\mathbf{x};\mu_1,\mu_2) &= \sum_{i=1}^{3} U_i(x_i) - \mu_1\cdot(x_1+x_3-1) - \mu_2\cdot(x_2+x_3-1)\\ &= [U_1(x_1)-\mu_1\cdot x_1] + [U_2(x_2)-\mu_2\cdot x_2]\\ &\quad +[U_3(x_3)-(\mu_1+\mu_2)\cdot x_3]\end{aligned}$$

where μ_1, μ_2 are the dual variables associated with the first two constraints (each for one link). They can be interpreted as the "*unit price*" of link 1 and link 2 to *"signal" their congestion level*. According to the theory of convex optimization [56], there exists $\mu_1^*, \mu_2^* \geq 0$ such that $\mathbf{x}^* := \arg\max_{\mathbf{x}} \mathcal{L}(\mathbf{x};\mu_1^*,\mu_2^*)$ is the optimum solution of problem (1.1). Since the terms in $\mathcal{L}(\mathbf{x};\mu_1,\mu_2)$ involving x_1, x_2, x_3 are separable, we have

$$\begin{aligned}x_1^* &= \arg\max_{x_1}\{U_1(x_1)-\mu_1^*\cdot x_1\}\\ x_2^* &= \arg\max_{x_2}\{U_2(x_2)-\mu_2^*\cdot x_2\}\\ x_3^* &= \arg\max_{x_3}\{U_3(x_3)-(\mu_1^*+\mu_2^*)\cdot x_3\} \qquad (1.2)\end{aligned}$$

The actual algorithms also include an iterative procedure for each link to locally find the prices μ_1^* and μ_2^*.

There are two important points about this solution.

1. Economic Interpretation: Once the proper prices μ_1^*, μ_2^* are set, the optimal $\mathbf{x}^*$ can be found if each flow chooses its rate according to the *total price along its path*, in order to maximizes its "payoff"—for example, $U_1(x_1) - \mu_1^* \cdot x_1$ is the "payoff" of flow 1 since $U_1(x_1)$ is its utility and $\mu_1^* \cdot x_1$ is its payment (although there is no actual payment involved).

In this sense, the price taken by one flow reflects the "*externality*" it causes to other flows. The reason why the maximum social welfare is achieved is that each flow takes into account the externality when maximizing its payoff (in other words, the "externality" is "internalized").

2. Distributed Implementation: Once the prices μ_1^*, μ_2^* are set, the global optimization (1.1) is reduced to a number of individual optimizations in (1.2). This leads to a distributed algorithms to find $\mathbf{x}^*$. Such algorithms are of particular interest in large scale communication networks, because the actions of each network device are simple and localized no matter how large the network is.

Since such algorithms can be "derived" from a global optimization problem, this method of algorithm design is called the "*optimization-based approach*".

1.1.2 Generalization

As a generalization of problem (1.1), consider n "players" in the network, where player i has a strategy $x_i, i = 1, 2, \ldots, n$. Define the vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and $\mathbf{x} \in \mathcal{X}$ where $\mathcal{X}$ is a "feasible set". Assume that player i's utility depends on $\mathbf{x}$ (not only x_i). Then, a global optimization problem is

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i=1}^{n} U_i(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{1.3}$$

After several early works including [1] [2], this problem has been studied in various contexts in communication networks, resulting in efficient and distributed algorithms to optimize various objectives such as throughput, delay, fairness and power consumptions [3].

1.2 Individual optimization and game theory

When the communication devices are controlled by humans, and without any "regulation" such as pricing in place, a "rational" player is only interested in maximizing his own utility. In this case, the individual optimization problem becomes important. Given $\mathbf{x}_{-i}$, player i solves

$$\begin{aligned} \max_{x_i} \quad & U_i(x_i, \mathbf{x}_{-i}) \\ \text{s.t.} \quad & x_i \in \mathcal{X}_i \end{aligned} \tag{1.4}$$

where $\mathcal{X}_i$ is the "action set" of player i.

If there is a $\bar{\mathbf{x}}$ such that $\bar{x}_i \in \arg\max_{x_i \in \mathcal{X}_i} U_i(x_i, \bar{\mathbf{x}}_{-i}), \forall i$, then $\bar{\mathbf{x}}$ is a "pure-strategy Nash Equilibrium" [4]. (Although a pure-strategy NE does not always exist, we use it here for introductory purposes.)

Since the externality is not internalized, in general $\sum_{i=1}^n U_i(\bar{\mathbf{x}}) < \sum_{i=1}^n U_i(\mathbf{x}^*)$ where $\mathbf{x}^*$ is the solution of (1.3). Then the following questions become interesting:

- Is the difference, or ratio between $\sum_{i=1}^n U_i(\mathbf{x}^*)$ and $\sum_{i=1}^n U_i(\bar{\mathbf{x}})$ bounded?
- If so, what is the largest difference or ratio? (The largest ratio has been named the "price of anarchy" (POA) in [8].)
- If POA is bad, how to improve it by modifying the game?

1.3 Organization

This dissertation addresses three problems, one at the human level and the other two at the engineering level. In the first problem, we consider *selfish* investment in network security, where each player chooses its investment in security to minimize his own cost. The main purpose of the study is to develop an analytical framework to understand the price of anarchy of this game, and study how to improve it using certain mechanisms or regulations. The second problem is achieving the maximum throughput and utility in a class of networks including wireless networks. Different from the first problem, the goal here is to maximize the social welfare through distributed algorithms. We use the optimization-based approach, but we need to incorporate several other elements (including time-reversible Markov chains, statistical mechanics and stochastic approximation) to design and analyze our algorithms.

The third problem is achieving the maximum throughput and utility in stochastic processing networks (SPNs). SPNs model manufacturing, communication, and service systems. They are more general than the queueing networks we consider in the second problem and pose unique challenges.

Part I

Selfish Investments in Network Security

Chapter 2

Introduction

Today's Internet suffers from various security problems. The best-known security problems are viruses and worms, which usually carry malicious codes and can cause considerable damage to the infected computers. The *key feature* of viruses and worms is that they *spread* across the network from infected computers to other computers. Viruses require user intervention to spread (such as opening an email or document which contains the virus), whereas worms spread automatically [5]

Another well known security risk is caused by the "Botnets". A Botnet is a collection of software robots (or "bots"), residing in infected computers, that can be controlled remotely by the Botnet operator to perform various malicious tasks such as installing Adware and Spyware in the infected computers, sending out spams to mail servers, and launching distributed Denial-of-Service attacks (DDoS) on certain websites. Like viruses and worms, many bots can automatically scan their environment and propagate themselves by exploiting system vulnerabilities.

Due to the "*contagious*" nature of the above risks, *Internet security does not only depend on the security investment made by individual users, but also on the interdependency among them.* If a careless user puts in little effort in protecting his computer system (e.g., installing anti-virus software and patching the system vulnerabilities), then it is easy for viruses, worms or bots to infect this computer and through it continue to infect or attack others.

On the contrary, if a user invests more to protect himself, then other users will also benefit since the chance of infection and attacks is reduced. Therefore, each user's investment exerts a "positive externality" on others.

Unfortunately, a selfish user (or "player") does not consider the above externality when choosing his investments in security, since he does not bear the full responsibility for his action. As a result, the overall network security is generally sub-optimum. We are interested in identifying and modeling the important factors that affect the extent of sub-optimality, that is, the "price of anarchy", and also consider possible ways to improve the outcome.

One important factor is the heterogeneity of user preferences. Internet users have different valuations of security and unit costs of investment. For example, government and commercial websites usually have higher valuations of security, since security breaches would lead to significant financial losses or other consequences. They are also more efficient in implementing security measures. On the other hand, a family computer user may care less about security, and also may be less efficient in improving it due to the lack of awareness and expertise. As a result, some players may choose to invest more, whereas others choose to "free ride", given that the security level is already "good" enough thanks to the investment of others. Due to the tendency of under-investment, the resulting outcome may be far worse for all users. This is the "free riding problem" as studied in, for example, [7].

Besides user preferences, the network topology, which is defined to describe the logical dependency among the players, is also important. The specific "dependency" studied here is the "importance" of a given user's investment to others. For example, assume that in a local network, user A directly connected to the Internet. All other users are connected to A and exchange a large amount of traffic with A. Clearly, the security level of A is particularly important for the local network since A has the largest influence on other users. If A has a low valuation of his own security, then it will invest little and the whole network suffers.

2.1 Overview of results

We first study how network topology, the preferences of users and their mutual influence affect network security in a non-cooperative setting. In the strategic-form game, we derive the "Price of Anarchy" (POA) [8] as a function of the above factors, where the POA here is defined as the worst-case ratio between the "social cost" at a Nash Equilibrium (NE) and the social optimum (SO). We show that the price of anarchy in general cannot be offset by the improvement of security technology. We also introduce the concept of "weighted POA" to get a richer characterization of the region of equilibrium payoffs. In a repeated game, users have more incentive to cooperate for their long-term interest. We study the "socially best" equilibrium in the repeated game, and compare it to SO. Not surprisingly, much better performance can be achieved in the repeated game. (The above results are based on [22; 23].)

Given that the POA is large in many scenarios, a natural question is how to improve the outcome of the game. A conceptually simple scheme with a regulator is called "due care" (see, for example, [7]). In the idealized case which we call "perfect due care", each player i is required to invest no less than x_i^*, the investment in the socially optimal configuration. Then, it can be shown that a NE is that each player i invest x_i^* which achieves the SO. However, since the regulator generally does not have a full knowledge of $\mathbf{x}^*$, we investigate a more general "due care" scheme where the regulator imposes a minimum investment vector $\mathbf{m}$ on the players where $\mathbf{m} \neq \mathbf{x}^*$ in general. We give the worst-case performance bound of the resulting NE and discuss how the bound could improve compared to the case without "due care".

During the study we have also developed a few interesting results for game theory itself. The "weighted POA" mentioned above is a general concept that can be applied to other games. We also developed a Folk theorem with local punishments and rewards, utilizing the structure of a class of games with positive externality.

2.2 Related works

In [6], Gordon and Loeb presented an economic model to determine the optimum investment of *a single player* to protect a given set of information. The model takes into account the vulnerability of the information and the potential loss if a security breach occurs. The externalities among different players, or the game theoretical aspects, were not considered.

Varian studied the network security problem using game theory in [7]. There, the effort of each player was assumed to be equally important to all other users (i.e., the symmetric case), and the network topology was not taken into account. Also, [7] is not focused on the efficiency analysis such as quantifying the POA.

In [14], Aspnes et al. formulated an "inoculation game" and studied its POA. There, each player in the network decides whether to install anti-virus software to avoid infection. Different from our work, [14] has assumed binary decisions (install or not install) and the same cost function for all players. Lelarge and Bolot [15] made a similar homogeneous assumption on the cost functions, and they obtained asymptotic results on the POA in random graphs when the number of players goes to infinity. Compared to these works, our results take into account heterogeneous cost functions, and apply to any given network topology with any number of players.

"Price of Anarchy" (POA) [8], measuring the performance of the worst-case equilibrium compared to the social optimum, has been studied in various games in recent years, most of them with "negative externality". Roughgarden et al. shows that the POA is generally unbounded in the "selfish routing game" [9; 10], where each user chooses some link(s) to send his traffic in order to minimize his congestion delay. Ozdaglar et al. derived the POA in a "price competition game" in [11] and [12], where a number of network service providers choose their prices to attract users and maximize their own revenues. In [13], Johari et al. studied the "resource allocation game", where each user bids for the resource to maximize his payoff, and showed that the POA is 3/4 assuming concave utility functions. In all the above games, there is "negative externality" among the players: for example in the "selfish routing game", if a user sends his traffic through a link, other users sharing that link will suffer larger delays.

On the contrary, in the network security game we study here, if a user increases his investment, the security level of other users will improve. In this sense, it falls into the category of games with positive externalities. In fact, many results here may be applicable to games with a similar nature. For example, assume that a number of service providers (SP) build networks which are interconnected. If a SP invests to upgrade her own network, the performance of the whole network improves and may bring more revenue to all SP's.

Chapter 3

Strategic-Form Games

3.1 Price of anarchy (POA) in the strategic-form game

3.1.1 General model

Assume there are n "players" where each player is normally an organization or enterprise. The security investment (or "effort", we use them interchangeably) of player i is $x_i \geq 0$. This includes investment in both finance (e.g., for purchasing/installing anti-virus software and firewall), time/energy (e.g., for system scanning, patching and maintenance) and education. The cost per unit of investment is $c_i > 0$. Denote $f_i(\mathbf{x})$ as player i's "security risk": the *expected* loss due to virus infections and attacks from the network, where $\mathbf{x}$ is the vector of investments by all players. $f_i(\mathbf{x})$ is decreasing[1] in each $x_j, j = 1, 2, \ldots, n$ (thus reflecting positive externality) and non-negative. We assume that it is convex and differentiable, and that $f_i(\mathbf{x} = \mathbf{0}) > 0$ is finite. Then the "cost function" of player i is

$$g_i(\mathbf{x}) := f_i(\mathbf{x}) + c_i x_i \tag{3.1}$$

Note that the function $f_i(\cdot)$ is generally different for different players.

The *strategic-form security game* Γ is formally defined as

$$\Gamma = (\mathcal{N}, \mathcal{X}, \mathbf{g}) \tag{3.2}$$

[1] "Decreasing" here means "non-increasing", different from "strictly decreasing".

where $\mathcal{N} = \{1, 2, \ldots, n\}$ is the set of players. $\mathcal{X}_i = \mathcal{R}_+$ is the action set of player i (i.e., his investment $x_i \in \mathcal{X}_i$), and $\mathcal{X} = \prod_{i=1}^n \mathcal{X}_i$ is the set of action profiles. $g_i : \mathcal{X} \to \mathcal{R}$, as defined in (3.1), is the cost function of player i, and $\mathbf{g} = (g_1, g_2, \ldots, g_n)$ is the cost functions for the game. In Γ, player i chooses his investment $x_i \geq 0$ to minimize $g_i(\mathbf{x})$. Also define $G(\mathbf{x})$ as the total cost (or "social cost") function:

$$G(\mathbf{x}) := \sum_{i=1}^{n} g_i(\mathbf{x}). \tag{3.3}$$

$G(\mathbf{x})$ serves as a global performance measure of a given profile $\mathbf{x}$.

Proposition 1. *As a simplification, we can assume that $c_i = 1, \forall i$ without loss of generality.*

Remark*: Given this, we will assume $c_i = 1, \forall i$ in most of our study.*

Proof. We show that given a game Γ as in (3.2), we can transform it to an equivalent game with unit cost 1.

To do this, we change a variable in the cost function (3.1), by defining $x'_i := c_i x_i, \forall i$. Denote $\mathbf{x}' = (x'_i) = (c_i x_i)$. Then,

$$\mathbf{x} = (x_i) = (x'_i / c_i). \tag{3.4}$$

Define the functions

$$\begin{aligned} \hat{f}_i(\mathbf{x}') &:= f_i(\mathbf{x}) \\ \hat{g}_i(\mathbf{x}') &:= \hat{f}_i(\mathbf{x}') + x'_i, \forall i \end{aligned} \tag{3.5}$$

where $\mathbf{x}$ is expressed in (3.4). Then, $\hat{g}_i(\mathbf{x}') = g_i(\mathbf{x}), \forall i$.

Clearly, the game $\hat{\Gamma} := (\mathcal{N}, \mathcal{X}, \hat{\mathbf{g}})$ (where $\hat{\mathbf{g}} = (\hat{g}_i)$) is equivalent to Γ. Also, in (3.5), the unit cost of investment x'_i is 1.

Note that $\hat{f}_i(\mathbf{x}')$ is non-increasing in $x'_j, j = 1, 2, \ldots, n$, and is convex and differentiable in $\mathbf{x}'$. Also, $\hat{f}_i(\mathbf{x}' = \mathbf{0}) > 0$ is finite. These properties are the same as those assumed for $f_i(\mathbf{x})$.

Therefore, for any game Γ, we can study it as the game $\hat{\Gamma}$ with unit cost 1 for each player. $\square$

3.1.2 A related empirical study

In this section we describe the main findings of a related empirical study in [16] on the security investments in Japanese enterprises. The purpose is to draw some correspondence between our general model and the empirical observations.

The data used in [16] is based on "Survey of actual condition of IT usage", conducted by METI (Ministry of Economy, Trade and Industry) of the Japanese government in March 2002 and 2003. The data set consists of responses from 3018 enterprises.

First of all, it was found from the data that security investments are significantly affected by the *industry type*. Financial organizations seem to invest more compared to other organizations, because they have larger potential losses in the event of security breaches.

This corresponds to the "heterogeneity" of the players in our model, reflected in the cost function $g_i(\mathbf{x}) = f_i(\mathbf{x}) + c_i x_i$ where $f_i(\cdot)$ and c_i differ for different players. If the security risk function is large, the player tends to invest more.

Now consider the enterprises with the same industry type. For each enterprise i, the following variables are defined in [16]:

- $v_i = 1$ if she suffered from virus attacks in 2003, and $v_i = 0$ if not.
- Let E_i be the logarithm of the number of email accounts. Since e-mail attachments are a major virus source, E_i is used to reflect the vulnerability arising from inside users.
- B_i is the "system vulnerability score", which is inherently higher if enterprise i has a large coverage of systems and networks.
- $x_i = 1$ if the enterprise adopted security measures including "Defense measures", "Security policy" and "Human cultivation"[2], $x_i = 0$ otherwise.

[2] "Human cultivation" means the education and training of the members of the enterprise (including the employees and managers) to increase their awareness of security issues and develop good security practices.

Then, the data is fit into the following proposed model via "logistic regression".

$$v_i = \alpha \cdot E_i + \beta \cdot B_i + \delta \cdot x_i \tag{3.6}$$

Not surprisingly, it was found that $\alpha, \beta > 0$ and $\delta < 0$ [16] after the regression. That is, the security risk v_i is positively correlated to the number of email accounts and the system vulnerability, and negatively correlated to the security investment.

Comparison of the data analysis in [16] and our model

- One can view v_i in (3.6) as a simplistic version of our risk function $f_i(\cdot)$, since in their study v_i is either 1 or 0, which does not reflect the amount of loss incurred by viruses. Also, one can view v_i as a realization of an inherently random variable, and assume that the player would make decisions based on the *expected* loss due to attacks, which is the definition of $f_i(\cdot)$ and will made even more clear in section 3.1.4 and 3.1.5.
- The vulnerability level (affected by E_i and B_i) of each enterprise has been accounted for by the different risk functions $f_i(\cdot)$ in our model (which is another form of heterogeneity), and the effect of investment is modeled by the assumption that $f_i(\cdot)$ is decreasing in x_i.
- One aspect which was not considered in (3.6) is the positive externality of each player's investment to other players.

3.1.3 POA in the general model

First, we prove in section 7.1 the existence of pure-strategy Nash Equilibrium(s).

Proposition 2. *There exists some pure-strategy Nash Equilibrium (NE) in* Γ.

In the section we consider pure-strategy NE. Denote $\bar{\mathbf{x}}$ as the vector of investments at some NE, and $\mathbf{x}^*$ as the vector of investments at the social optimum (SO), i.e., $\mathbf{x}^* \in \arg\min_{\mathbf{x} \geq \mathbf{0}} G(\mathbf{x})$. Also denote the unit cost vector $\mathbf{c} = (c_1, c_2, \ldots, c_n)^T$.

We aim to find the POA, Q, which is defined as the largest possible $\rho(\bar{\mathbf{x}})$, where

$$\rho(\bar{\mathbf{x}}) := \frac{G(\bar{\mathbf{x}})}{G^*} = \frac{\sum_i g_i(\bar{\mathbf{x}})}{\sum_i g_i(\mathbf{x}^*)}$$

is the ratio between the social cost at the NE $\bar{\mathbf{x}}$ and at the social optimum. For convenience, sometimes we simply write $\rho(\bar{\mathbf{x}})$ as ρ if there is no confusion.

Before getting to the derivation, we illustrate the POA in a simple example. Assume there are 2 players, with their investments denoted as $x_1 \geq 0$ and $x_2 \geq 0$. The cost function is $g_i(\mathbf{x}) = f(y) + x_i, i = 1, 2$, where $f(y)$ is the security risk of both players, and $y = x_1 + x_2$ is the total investment. Assume that $f(y)$ is non-negative, decreasing, convex, and satisfies $f(y) \to 0$ when $y \to \infty$. The social cost is $G(\mathbf{x}) = g_1(\mathbf{x}) + g_2(\mathbf{x}) = 2 \cdot f(y) + y$.

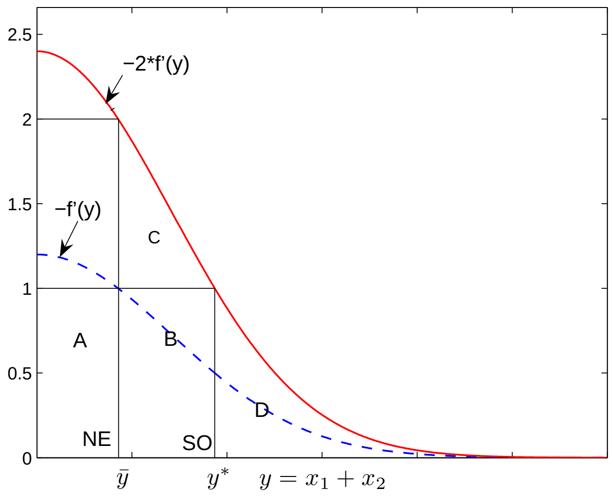


Figure 3.1: POA in a simple example

At a NE $\bar{\mathbf{x}}$, $\frac{\partial g_i(\bar{\mathbf{x}})}{\partial x_i} = f'(\bar{x}_1 + \bar{x}_2) + 1 = 0, i = 1, 2$. Denote $\bar{y} = \bar{x}_1 + \bar{x}_2$, then $-f'(\bar{y}) = 1$. This is shown in Fig 3.1. Then, the social cost $\bar{G} = 2 \cdot f(\bar{y}) + \bar{y}$. Note that $\int_{\bar{y}}^{\infty}(-f'(z))dz = f(\bar{y}) - f(\infty) = f(\bar{y})$ (since $f(y) \to 0$ as $y \to \infty$), therefore in Fig 3.1, $2 \cdot f(\bar{y})$ is the area $B + C + D$, and $\bar{G}$ is equal to the area of $A + (B + C + D)$.

At SO, on the other hand, the total investment y^* satisfies $-2f'(y^*) = 1$. Using a similar argument as before, $G^* = 2f(y^*) + y^*$ is equal to the area of $(A + B) + D$.

Then, the ratio $\bar{G}/G^* = [A + (B + C + D)]/[(A + B) + D] \leq (B + C)/B \leq 2$. We will show later that this upper bound is tight. So the POA is 2.

Now we analyze the POA with the general cost function (3.1). In some sense, it is a generalization of the above example.

Lemma 1. *For any NE $\bar{\mathbf{x}}$, $\rho(\bar{\mathbf{x}})$ satisfies*

$$\rho(\bar{\mathbf{x}}) \leq \max\{1, \max_k\{(-\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k})/c_k\}\} \tag{3.7}$$

Note that $(-\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k})$ is the marginal "benefit" to the security of all users by increasing x_k at the NE; whereas c_k is the marginal cost of increasing x_k. The second term in the RHS (right-hand-side) of (3.7) is the maximal ratio between these two.

Proof. At NE,

$$\begin{cases} \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_i} = -c_i & \text{if } \bar{x}_i > 0 \\ \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_i} \geq -c_i & \text{if } \bar{x}_i = 0 \end{cases} \tag{3.8}$$

By definition,

$$\rho(\bar{\mathbf{x}}) = \frac{G(\bar{\mathbf{x}})}{G^*} = \frac{\sum_i f_i(\bar{\mathbf{x}}) + \mathbf{c}^T\bar{\mathbf{x}}}{\sum_i f_i(\mathbf{x}^*) + \mathbf{c}^T\mathbf{x}^*}$$

Since $f_i(\cdot)$ is convex for all i. Then $f_i(\bar{\mathbf{x}}) \leq f_i(\mathbf{x}^*) + (\bar{\mathbf{x}} - \mathbf{x}^*)^T \nabla f_i(\bar{\mathbf{x}})$. So

$$\begin{aligned} G(\bar{\mathbf{x}}) &\leq (\bar{\mathbf{x}} - \mathbf{x}^*)^T \sum_i \nabla f_i(\bar{\mathbf{x}}) + \mathbf{c}^T\bar{\mathbf{x}} + \sum_i f_i(\mathbf{x}^*) \\ &= -\mathbf{x}^{*T} \sum_i \nabla f_i(\bar{\mathbf{x}}) + \bar{\mathbf{x}}^T[\mathbf{c} + \sum_i \nabla f_i(\bar{\mathbf{x}})] + \sum_i f_i(\mathbf{x}^*) \end{aligned}$$

Note that

$$\bar{\mathbf{x}}^T[\mathbf{c} + \textstyle\sum_i \nabla f_i(\bar{\mathbf{x}})] = \sum_i \bar{x}_i[c_i + \sum_k \frac{\partial f_k(\bar{\mathbf{x}})}{\partial x_i}]$$

There are two possibilities for every player i: (a) If $\bar{x}_i = 0$, then $\bar{x}_i[c_i + \sum_k \frac{\partial f_k(\bar{\mathbf{x}})}{\partial x_i}] = 0$. (b) If $\bar{x}_i > 0$, then $\frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_i} = -c_i$. Since $\frac{\partial f_k(\bar{\mathbf{x}})}{\partial x_i} \leq 0$ for all k, then $\sum_k \frac{\partial f_k(\bar{\mathbf{x}})}{\partial x_i} \leq -c_i$, so $\bar{x}_i[c_i + \sum_k \frac{\partial f_k(\bar{\mathbf{x}})}{\partial x_i}] \leq 0$.

As a result,

$$G(\bar{\mathbf{x}}) \leq -\mathbf{x}^{*T} \sum_i \nabla f_i(\bar{\mathbf{x}}) + \sum_i f_i(\mathbf{x}^*) \tag{3.9}$$

and

$$\rho(\bar{\mathbf{x}}) \leq \frac{-\mathbf{x}^{*T}\sum_i \nabla f_i(\bar{\mathbf{x}}) + \sum_i f_i(\mathbf{x}^*)}{\sum_i f_i(\mathbf{x}^*) + \mathbf{c}^T\mathbf{x}^*} \tag{3.10}$$

(i) If $x_i^* = 0$ for all i, then the RHS is 1, so $\rho(\bar{\mathbf{x}}) \leq 1$. Since ρ cannot be smaller than 1, we have $\rho = 1$.

(ii) If not all $x_i^* = 0$, then $\mathbf{c}^T\mathbf{x}^* > 0$. Note that the RHS of (3.10) is not less than 1, by the definition of $\rho(\bar{\mathbf{x}})$. So, if we subtract $\sum_i f_i(\mathbf{x}^*)$ (non-negative) from both the numerator and the denominator, the resulting ratio upper-bounds the RHS. That is,

$$\rho(\bar{\mathbf{x}}) \leq \frac{-\mathbf{x}^{*T}\sum_i \nabla f_i(\bar{\mathbf{x}})}{\mathbf{c}^T\mathbf{x}^*} \leq \max_k\{(-\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k})/c_k\}$$

where $\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k}$ is the k'th element of the vector $\sum_i \nabla f_i(\bar{\mathbf{x}})$.

Combining case (i) and (ii), the proof is completed. □

Lemma 2. *We can also bound the difference between $G(\bar{\mathbf{x}})$ and G^*. Using (3.9),*

$$\begin{aligned} G(\bar{\mathbf{x}}) - G^* &\leq -\mathbf{x}^{*T}\sum_i \nabla f_i(\bar{\mathbf{x}}) - \mathbf{c}^T\mathbf{x}^* \\ &\leq \{\max_k\{(-\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k})/c_k\} - 1\} \cdot (\mathbf{c}^T\mathbf{x}^*) \end{aligned} \tag{3.11}$$

Note that although Lemma 1 is quite general, the bound is not explicit since it involves $\bar{\mathbf{x}}$.

In the following, we give two models of the network security game which are special cases of the above general model. Each model defines a concrete form of $f_i(\cdot)$. They are formulated to capture the key features of the system while being amenable to mathematical analysis. We will give explicit expressions for the POA of the two models.

3.1.4 Effective-investment ("EI") model

Generalizing [7], we consider an "Effective-investment" (EI) model. In this model, the security risk of player i depends on an "effective investment", which we assume is a linear combination of the investments of himself and other players.

Specifically, let $p_i(\sum_{j=1}^n \alpha_{ji} z_j)$ be the probability that player i is infected by a virus (or suffers an attack), given the amount of effort every player puts in. The effort of player j, z_j, is weighted by α_{ji}, reflecting the "importance" of player j to player i. Let v_i be the cost of player i if he suffers an attack; and c_i be the cost per unit of effort by player i. Then, the total cost of player i is $g_i(\mathbf{z}) = v_i p_i(\sum_{j=1}^n \alpha_{ji} z_j) + c_i z_i$.

For convenience, we "normalize" the expression in the following way. Let the normalized effort be $x_i := c_i z_i, \forall i$. Then

$$\begin{aligned} g_i(\mathbf{x}) &= v_i p_i(\textstyle\sum_{j=1}^n \frac{\alpha_{ji}}{c_j} x_j) + x_i \\ &= v_i p_i(\frac{\alpha_{ii}}{c_i} \textstyle\sum_{j=1}^n \beta_{ji} x_j) + x_i \end{aligned}$$

where $\beta_{ji} := \frac{c_i}{\alpha_{ii}} \frac{\alpha_{ji}}{c_j}$ (so $\beta_{ii} = 1$). We call β_{ji} the "*relative importance*" of player j to player i.

Define the function $V_i(y) = v_i \cdot p_i(\frac{\alpha_{ii}}{c_i} y)$, where y is a dummy variable. Then $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$, where

$$f_i(\mathbf{x}) = V_i(\textstyle\sum_{j=1}^n \beta_{ji} x_j) \tag{3.12}$$

Assume that $p_i(\cdot)$ is decreasing, non-negative, convex and differentiable. Then $V_i(\cdot)$ also has these properties.

Proposition 3. *In the EI model defined above,*

$$\rho \leq \max_k \{1 + \sum_{i:i \neq k} \beta_{ki}\} := Q_{EI}. \tag{3.13}$$

Furthermore, the bound is tight.

Proof. Let $\bar{\mathbf{x}}$ be some NE. Denote $\mathbf{h} := \sum_i \nabla f_i(\bar{\mathbf{x}})$. Then the kth element of $\mathbf{h}$

$$\begin{aligned} h_k &= \textstyle\sum_i \frac{\partial V_i(\sum_{j=1}^n \beta_{ji} \bar{x}_j)}{\partial x_k} \\ &= \textstyle\sum_i \beta_{ki} \cdot V_i'(\sum_{j=1}^n \beta_{ji} \bar{x}_j) \end{aligned}$$

From (3.8), we have $\frac{\partial V_i(\sum_{j=1}^n \beta_{ji}\bar{x}_j)}{\partial x_i} = \beta_{ii} \cdot V_i'(\sum_{j=1}^n \beta_{ji}\bar{x}_j) = V_i'(\sum_{j=1}^n \beta_{ji}\bar{x}_j) \geq -1$. So $h_k \geq -\sum_i \beta_{ki}$. Plug this into (3.7), we obtain an upper bound of ρ:

$$\rho \leq \max\{1, \max_k\{-h_k\}\} \leq Q_{EI} := \max_k\{1 + \sum_{i:i\neq k} \beta_{ki}\} \tag{3.14}$$

which completes the proof. □

(3.14) gives some interesting insight into the game. Since $\beta_{ki}, i \neq k$ is player k's "relative importance" (or externality) to player i, then $1 + \sum_{i:i\neq k} \beta_{ki} = \sum_i \beta_{ki}$ is player k's relative importance to the society. (3.14) shows that the POA is bounded by the maximal social "importance" (or one plus the "total externality") among the players. Interestingly, the bound does not depend on the specific form of $V_i(\cdot)$ as long as it's convex, decreasing and non-negative.

It also provides a simple way to compute POA under the model. We define a "dependency graph" as in Fig. 3.2, where each vertex stands for a player, and there is a directed edge from k to i if $\beta_{ki} > 0$. In Fig. 3.2, player 3 has the highest social importance, and $\rho \leq 1 + (0.6 + 0.8 + 0.8) = 3.2$. In another special case, if for each pair (k, i), either $\beta_{ki} = 1$ or $\beta_{ki} = 0$, then the POA is bounded by the maximum out-degree of the graph plus 1. If all players are equally important to each other, i.e., $\beta_{ki} = 1, \forall k, i$, then $\rho \leq n$ (i.e., POA is the number of players). This also explains why the POA is 2 in the example considered in Fig 3.1.

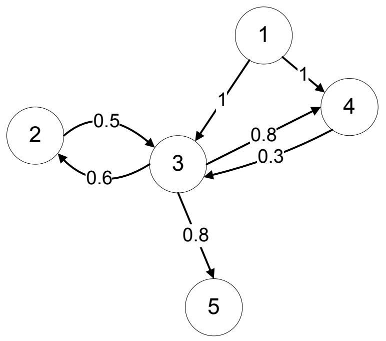


Figure 3.2: Dependency Graph and the Price of Anarchy (In this figure, $\rho \leq 1 + (0.6 + 0.8 + 0.8) = 3.2$)

The following is a worst case scenario that shows the bound is tight. Assume there are n players, $n \geq 2$. $\beta_{ki} = 1, \forall k, i$; and for all i, $V_i(y_i) = [(1-\epsilon)(1-y_i)]_+$, where $[\cdot]_+$ means positive part, $y_i = \sum_{j=1}^n \beta_{ji} x_j = \sum_{j=1}^n x_j$, $\epsilon > 0$ but is very small.[3]

Given $\mathbf{x}_{-i} = \mathbf{0}$, $g_i(\mathbf{x}) = [(1-\epsilon)(1-x_i)]_+ + x_i = (1-\epsilon) + \epsilon \cdot x_i$ when $x_i \leq 1$, so the best response for player i is to let $x_i = 0$. Therefore, $\bar{x}_i = 0, \forall i$ is a NE, and the resulting social cost $G(\bar{\mathbf{x}}) = \sum_i [V_i(0) + \bar{x}_i] = (1-\epsilon)n$. Since the social cost is $G(\mathbf{x}) = n \cdot [(1-\epsilon)(1-\sum_i x_i)]_+ + \sum_i x_i$, the social optimum is attained when $\sum_i x_i^* = 1$ (since $n(1-\epsilon) > 1$). Then, $G(\mathbf{x}^*) = 1$. Therefore $\rho = (1-\epsilon)n \to n$ when $\epsilon \to 0$. When $\epsilon = 0$, $\bar{x}_i = 0, \forall i$ is still a NE. In that case $\rho = n$.

3.1.5 Bad-traffic ("BT") Model

Next, we consider a model which is based on the amount of "bad traffic" (e.g., traffic that causes virus infection) from one player to another. Let r_{ki} be the total rate of traffic from k to i. How much traffic in r_{ki} will do harm to player i depends on the investments of both k and i. So denote by $\phi_{k,i}(x_k, x_i)$ the probability that player k's traffic does harm to player i. Clearly $\phi_{k,i}(\cdot,\cdot)$ is a non-negative, decreasing function. We also assume it is convex and differentiable. Then, the rate at which player i is infected by the traffic from player k is $r_{ki}\phi_{k,i}(x_k, x_i)$. Let v_i be player i's loss when it's infected by a virus, then $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$, where the investment x_i has been *normalized* such that its coefficient (the unit cost) is 1, and

$$f_i(\mathbf{x}) = v_i \sum_{k \neq i} r_{ki} \phi_{k,i}(x_k, x_i)$$

If the "firewall" of each player is symmetric (i.e., it treats the incoming and outgoing traffic in the same way), then it's reasonable to assume that $\phi_{k,i}(x_k, x_i) = \phi_{i,k}(x_i, x_k)$.

Proposition 4. *In the BT model,*

$$\rho \leq 1 + \max_{(i,j): i \neq j} \frac{v_i r_{ji}}{v_j r_{ij}} := Q_{BT}.$$

. The bound is also tight.

[3]Although $V_i(y_i)$ is not differentiable at $y_i = 1$, it can be approximated by a differentiable function arbitrarily closely, such that the result of the example is not affected.

Proof. Let $\mathbf{h} := \sum_i \nabla f_i(\bar{\mathbf{x}})$ for some NE $\bar{\mathbf{x}}$. Then the j-th element

$$\begin{aligned} h_j &= \sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_j} = \sum_{i \neq j} \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_j} + \frac{\partial f_j(\bar{\mathbf{x}})}{\partial x_j} \\ &= \sum_{i \neq j} v_i r_{ji} \frac{\partial \phi_{j,i}(\bar{x}_j, \bar{x}_i)}{\partial x_j} + v_j \sum_{i \neq j} r_{ij} \frac{\partial \phi_{i,j}(\bar{x}_i, \bar{x}_j)}{\partial x_j} \end{aligned}$$

We have

$$\begin{aligned} q_j &:= \frac{\sum_{i \neq j} \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_j}}{\frac{\partial f_j(\bar{\mathbf{x}})}{\partial x_j}} = \frac{\sum_{i \neq j} v_i r_{ji} \frac{\partial \phi_{j,i}(\bar{x}_j, \bar{x}_i)}{\partial x_j}}{v_j \sum_{i \neq j} r_{ij} \frac{\partial \phi_{i,j}(\bar{x}_i, \bar{x}_j)}{\partial x_j}} \\ &= \frac{\sum_{i \neq j} v_i r_{ji} \frac{\partial \phi_{j,i}(\bar{x}_j, \bar{x}_i)}{\partial x_j}}{\sum_{i \neq j} v_j r_{ij} \frac{\partial \phi_{j,i}(\bar{x}_j, \bar{x}_i)}{\partial x_j}} \leq \max_{i:i \neq j} \frac{v_i r_{ji}}{v_j r_{ij}} \end{aligned}$$

where the 3rd equality holds because $\phi_{i,j}(x_i, x_j) = \phi_{j,i}(x_j, x_i)$ by assumption.

From (3.8), we know that $\frac{\partial f_j(\bar{\mathbf{x}})}{\partial x_j} \geq -1$. So

$$h_j = (1 + q_j) \frac{\partial f_j(\bar{\mathbf{x}})}{\partial x_j} \geq -(1 + \max_{i:i \neq j} \frac{v_i r_{ji}}{v_j r_{ij}})$$

According to (3.7), it follows that

$$\rho \leq \max\{1, \max_j \{-h_j\}\} \leq Q_{BT} := 1 + \max_{(i,j):i \neq j} \frac{v_i r_{ji}}{v_j r_{ij}} \tag{3.15}$$

which completes the proof. □

Note that $v_i r_{ji}$ is the damage to player i caused by player j if player i is infected by all the traffic sent by j, and $v_j r_{ij}$ is the damage to player j caused by player i if player j is infected by all the traffic sent by i. Therefore, (3.15) means that the POA is upper-bounded by the "maximum imbalance" of the network. As a special case, if each pair of the network is "balanced", i.e., $v_i r_{ji} = v_j r_{ij}, \forall i, j$, then $\rho \leq 2$!

To show the bound is tight, we can use a similar example as in section 3.1.4. Let there be two players, and assume $v_1 r_{21} = v_1 r_{12} = 1$; $\phi_{1,2}(x_1, x_2) = (1 - \epsilon)(1 - x_1 - x_2)_+$. Then it becomes the same as the previous example when $n = 2$. Therefore $\rho \to 2$ as $\epsilon \to 0$. And $\rho = 2$ when $\epsilon = 0$.

Note that when the network becomes larger, the imbalance between a certain pair of players becomes less important. Thus ρ may be much less than the worst case bound in large networks due to the averaging effect.

3.2 Bounding the payoff regions using "weighted POA"

So far in the literature, the research on POA in various games has largely focused on the worst-case ratio between the social cost (or welfare) achieved at the Nash Equilibria and the social optimum. Note that the social cost (which is the summation of the individual costs of all players) only provides one-dimensional information. Therefore the POA is also one-dimensional information.

However, in any multi-player game, the players' payoffs form a vector which is multi-dimensional. Therefore, it is useful to have a richer characterization of the *region* of the payoff vectors. This region gives much more information because it characterizes the tradeoff between *efficiency* and *fairness* among different players.

This motivates us to introduce the concept of "weighted POA" which generalizes POA. With weighted POA, supposing that a NE payoff vector is known, one can bound the region of all feasible payoff vectors. This gives a better comparison between the NE payoff vector and the "Pareto frontier" of the feasible payoff region. Conversely, given any feasible payoff vector, one can bound the region of the possible payoff vectors at all Nash Equilibria.

The "weighted POA", $Q_{\mathbf{w}}$, is defined as the largest possible $\rho_{\mathbf{w}}(\bar{\mathbf{x}})$, where

$$\rho_{\mathbf{w}}(\bar{\mathbf{x}}) := \frac{G_{\mathbf{w}}(\bar{\mathbf{x}})}{G^*_{\mathbf{w}}} = \frac{\sum_i w_i \cdot g_i(\bar{\mathbf{x}})}{\sum_i w_i \cdot g_i(\mathbf{x}^*_{\mathbf{w}})}$$

Here, $\mathbf{w} \in \mathcal{R}^n_{++}$ is a weight vector, $\bar{\mathbf{x}}$ is the vector of investments at a NE of the original game; whereas $\mathbf{x}^*_{\mathbf{w}}$ minimizes a weighted social cost $G_{\mathbf{w}}(\mathbf{x}) := \sum_i w_i \cdot g_i(\mathbf{x})$.

Fig. 3.3 illustrates the concept of weighted POA in a 2-player game. The dash-dot red curve is the Pareto boundary of the feasible payoff region. And there is a unique NE whose payoff vector (or "cost vector") is marked by the circle. Then, the POA is equal to a/b. The weighted POA with weight vector $\mathbf{w} = (1, 0.5)$ is c/d. This is because the NE cost vector is on the line $g_1 + 0.5 \cdot g_2 = c$, and the cost vector which minimizes $G_{(2,1)}(\mathbf{x})$ is on the line $g_1 + 0.5 \cdot g_2 = d$.

To obtain $Q_{\mathbf{w}}$, consider a modified game

$$\hat{\Gamma} := (\mathcal{N}, \mathcal{X}, \hat{\mathbf{g}}) \tag{3.16}$$

where $\hat{\mathbf{g}} = (\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_n)$, and the cost function of player i is

$$\hat{g}_i(\mathbf{x}) := w_i \cdot g_i(\mathbf{x}) = w_i f_i(\mathbf{x}) + w_i \cdot c_i x_i := \hat{f}_i(\mathbf{x}) + \hat{c}_i x_i$$

Note that in the game $\hat{\Gamma}$, the NE strategies are the same as the original game Γ: given any $\mathbf{x}_{-i}$, player i's best response remains the same (since his cost function is only multiplied by a constant). So the two games are strategically equivalent, and thus have the same set of NE's. As a result, the weighted POA $Q_{\mathbf{w}}$ of the original game is exactly the POA in the modified game (Note the definition of $\mathbf{x}^*_{\mathbf{w}}$). Applying (3.7) to the game $\hat{\Gamma}$, we have

$$\begin{aligned}\rho_{\mathbf{w}}(\bar{\mathbf{x}}) &\leq \max\{1, \max_k\{(-\sum_i \frac{\partial \hat{f}_i(\bar{\mathbf{x}})}{\partial x_k})/\hat{c}_k\}\} \\ &= \max\{1, \max_k\{(-\sum_i \frac{w_i \partial f_i(\bar{\mathbf{x}})}{\partial x_k})/(w_k c_k)\}\} \qquad (3.17)\end{aligned}$$

Then, one can easily obtain the weighted POA for the EI model and the BT model.

Proposition 5. *In the EI model,*

$$\rho_{\mathbf{w}} \leq Q_{\mathbf{w},EI} := \max_k\{1 + \frac{\sum_{i:i\neq k} w_i \beta_{ki}}{w_k}\} \qquad (3.18)$$

In the BT model,

$$\rho_{\mathbf{w}} \leq Q_{\mathbf{w},BT} := 1 + \max_{(i,j):i\neq j} \frac{w_i v_i r_{ji}}{w_j v_j r_{ij}} \qquad (3.19)$$

We use Q_w to generally refer to $Q_{w,EI}$ or $Q_{w,BT}$, depending on the model.

Since $\rho_{\mathbf{w}}(\bar{\mathbf{x}}) = \frac{G_{\mathbf{w}}(\bar{\mathbf{x}})}{G^*_{\mathbf{w}}} = \frac{\sum_i w_i \cdot g_i(\bar{\mathbf{x}})}{\sum_i w_i \cdot g_i(\mathbf{x}^*_{\mathbf{w}})} \leq Q_{\mathbf{w}}$, we have $\sum_i w_i \cdot g_i(\mathbf{x}^*_{\mathbf{w}}) \geq \sum_i w_i \cdot g_i(\bar{\mathbf{x}})/Q_{\mathbf{w}}$. Notice that $\mathbf{x}^*_{\mathbf{w}}$ minimizes $G_{\mathbf{w}}(\mathbf{x}) = \sum_i w_i \cdot g_i(\mathbf{x})$, so for any feasible $\mathbf{x}$,

$$\sum_i w_i \cdot g_i(\mathbf{x}) \geq \sum_i w_i \cdot g_i(\mathbf{x}^*_{\mathbf{w}}) \geq \sum_i w_i \cdot g_i(\bar{\mathbf{x}})/Q_{\mathbf{w}}$$

Then we have the following.

Proposition 6. *Given any NE payoff vector* $\bar{\mathbf{g}}$*, then any feasible payoff vector* $\mathbf{g}$ *must be within the region*

$$\mathcal{B} := \{\mathbf{g} | \mathbf{w}^T\mathbf{g} \geq \mathbf{w}^T\bar{\mathbf{g}}/Q_{\mathbf{w}}, \forall \mathbf{w} \in \mathcal{R}^n_{++}\}$$

Conversely, given any feasible payoff vector $\mathbf{g}$*, any possible NE payoff vector* $\bar{\mathbf{g}}$ *is in the region*

$$\bar{\mathcal{B}} := \{\bar{\mathbf{g}} | \mathbf{w}^T \bar{\mathbf{g}} \leq \mathbf{w}^T \mathbf{g} \cdot Q_{\mathbf{w}}, \forall \mathbf{w} \in \mathcal{R}^n_{++}\}$$

In other words, the Pareto frontier of $\mathcal{B}$ lower-bounds the Pareto frontier of the feasible region of $\mathbf{g}$. (A similar statement can be made for $\bar{\mathcal{B}}$.)

As an illustrative example, consider the EI model, where the cost function of player i is of the form $g_i(\mathbf{x}) = V_i(\sum_{j=1}^n \beta_{ji} x_j) + x_i$. Assume there are two players in the game, and $\beta_{11} = \beta_{22} = 1$, $\beta_{12} = \beta_{21} = 0.2$. Also assume that $g_i(\mathbf{x}) = (1 - \sum_{j=1}^2 \beta_{ji} x_i)_+ + x_i$, for $i = 1, 2$. It is easy to verify that $\bar{x}_i = 0, i = 1, 2$ is a NE, and $g_1(\bar{\mathbf{x}}) = g_2(\bar{\mathbf{x}}) = 1$. One can further find that the boundary (Pareto frontier) of the feasible payoff region in this example is composed of the two axes and the following line segments (the computation is omitted):

$$\begin{cases} g_2 = -5 \cdot (g_1 - \frac{1}{1.2}) + \frac{1}{1.2} & g_1 \in [0, \frac{5}{6}] \\ g_2 = -0.2 \cdot (g_1 - \frac{1}{1.2}) + \frac{1}{1.2} & g_1 \in [0, 5] \end{cases}$$

which is the dashed line in Fig. 3.4.

By Proposition 6, for every weight vector $\mathbf{w}$, there is a straight line that lower-bounds the feasible payoff region. After plotting the lower bounds for many different $\mathbf{w}$'s, we obtain a bound for the feasible payoff region (Fig 3.4). Note that the bound only depends on the coefficients β_{ji}'s, but not the specific form of $V_1(\cdot)$ and $V_2(\cdot)$. We see that the feasible region is indeed within the bound.

3.3 Improvement of technology

Recall that in game $\Gamma = (\mathcal{N}, \mathcal{X}, \mathbf{g})$, the general cost function of player i is

$$g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i. \tag{3.20}$$

where we have assumed that the unit cost $c_i = 1$ without loss of generality.

Now assume that the security technology has improved. We would like to study how effective is technology improvement compared to the improvement of incentives. In the new

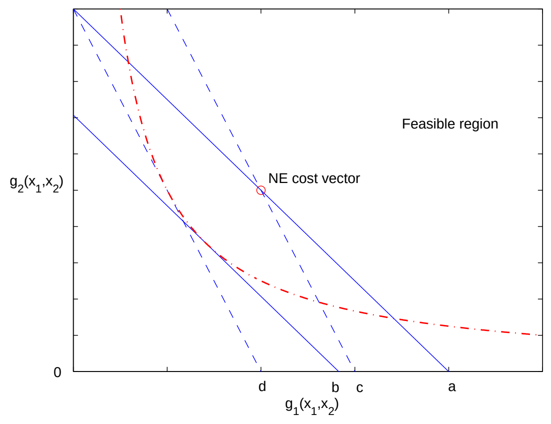


Figure 3.3: Illustration of weighted POA in a 2-player game. (Assume that the NE is unique. In this example, the POA is a/b; the weighted POA with weight vector $\mathbf{w} = (1, 0.5)$ is c/d.)

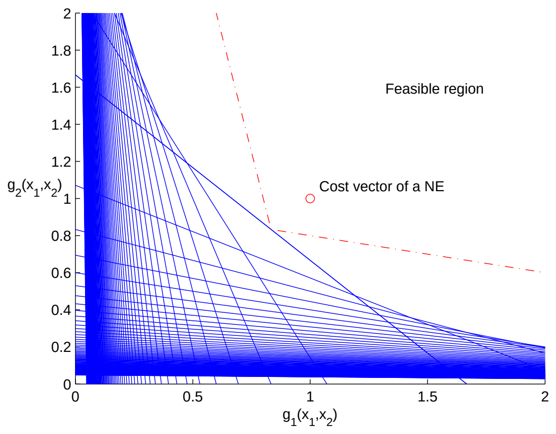


Figure 3.4: Bounding the feasible payoff region using weighted POA

game

$$\Gamma_{TI} := (\mathcal{N}, \mathcal{X}, \tilde{\mathbf{g}}) \tag{3.21}$$

where "TI" stands for "technology improvement", the new cost function of player i is

$$\tilde{g}_i(\mathbf{x}) = f_i(a \cdot \mathbf{x}) + x_i, a > 1. \tag{3.22}$$

This means that the effectiveness of the investment vector $\mathbf{x}$ has improved by a times (i.e., the risk decreases faster with $\mathbf{x}$ than before). Equivalently, if we define $\mathbf{x}' = a \cdot \mathbf{x}$, then (3.22) is $\tilde{g}_i(\mathbf{x}) = f_i(\mathbf{x}') + x'_i/a$, which means a decrease of unit cost if we regard $\mathbf{x}'$ as the investment.

Proposition 7. *Denote by G^* the optimal social cost in game Γ, and denote by $\tilde{G}^*$ the optimal social cost in game Γ_{TI}. Then,*

$$G^* \geq \tilde{G}^* \geq G^*/a. \tag{3.23}$$

That is, the optimal social cost decreases but cannot decrease more than a times.

Proof. First, for all $\mathbf{x}$, $\tilde{g}_i(\mathbf{x}) \leq g_i(\mathbf{x})$. Therefore $\tilde{G}^* \leq G^*$.

Let the optimal investment vector with the improved cost functions be $\tilde{x}^*$. We have $g_i(a \cdot \tilde{x}^*) = f_i(a \cdot \tilde{x}^*) + a \cdot \tilde{x}_i^*$. Also, $\tilde{g}_i(\tilde{x}^*) = f_i(a \cdot \tilde{x}^*) + \tilde{x}_i^*$. Then, $a \cdot \tilde{g}_i(\tilde{x}^*) = a \cdot f_i(a \cdot \tilde{x}^*) + a \cdot \tilde{x}_i^* \geq g_i(a \cdot \tilde{x}^*)$, because $f_i(\cdot)$ is non-negative and $a > 1$.

Therefore, we have $a \cdot \sum_i \tilde{g}_i(\tilde{x}^*) = a \cdot \tilde{G}^* \geq G(a \cdot \tilde{x}^*) \geq G(\mathbf{x}^*) = G^*$, since $\mathbf{x}^*$ minimizes $G(\mathbf{x}) = \sum_i g_i(\mathbf{x})$. This completes the proof. □

Here we have seen that the optimal social cost (after technology improved a times) is at least a fraction of $1/a$ of the optimal social cost before. On the other hand, we have the following about the POA after technology improvement.

Proposition 8. *Under the EI model and the BT model, the POA in game Γ_{TI} is the same as the POA in game Γ . (That is, the expressions of POA are the same as those given in Proposition 3 and 4.)*

Proof. The POA in the EI model only depends on the values of β_{ji}'s, which does not change with the new cost functions. To see this, note that

$$\begin{aligned} \tilde{g}_i(\mathbf{x}) &= f_i(a \cdot \mathbf{x}) + x_i \\ &= V_i(a \cdot \sum_j \beta_{ji} x_j) + x_i. \end{aligned}$$

Define the function $\tilde{V}_i(y) = V_i(a \cdot y), \forall i$, where y is a dummy variable, then $\tilde{g}_i(\mathbf{x}) = \tilde{V}_i(\sum_j \beta_{ji} x_j) + x_i$, where $\tilde{V}_i(\cdot)$ is still convex, decreasing and non-negative. So the β_{ji} values do not change. By Proposition 3, the POA remains the same.

In the BT model, define $\tilde{\phi}_{k,i}(x_k, x_i) := \phi_{k,i}(a \cdot x_k, a \cdot x_i)$, then $\tilde{\phi}_{k,i}(x_k, x_i)$ is still non-negative, decreasing and convex, and $\tilde{\phi}_{k,i}(x_k, x_i) = \tilde{\phi}_{i,k}(x_i, x_k)$. So by Proposition 4, the POA has the same expression as before. $\square$

To compare the effect of incentive improvement and technology improvement, consider the following two options to improve the network security.

1. With the current technology, deploy proper incentivizing mechanisms (i.e., "stick and carrot") to achieve the social optimum.
2. All players upgrade to the new technology, without solving the incentive problem.

With option 1, the resulting social cost is G^*. With option 2, (without even considering the cost of upgrading), the social cost is $\tilde{G}(\tilde{x}_{NE})$, where $\tilde{G}(\cdot) = \sum_i \tilde{g}_i(\cdot)$ is the social cost function in game Γ_{TI}, and $\tilde{x}_{NE}$ is a NE in Γ_{TI}. Define $\rho(\tilde{x}_{NE}) := \tilde{G}(\tilde{x}_{NE})/\tilde{G}^*$, then the ratio between the social costs with option 2 and option 1 is

$$\tilde{G}(\tilde{x}_{NE})/G^* = \rho(\tilde{x}_{NE}) \cdot \tilde{G}^*/G^* \geq \rho(\tilde{x}_{NE})/a$$

where the last step follows from Proposition 7. Also, by Proposition 8, in the EI or BT model, $\rho(\tilde{x}_{NE})$ is equal to the POA shown in Prop. 3 and 4 in the worst case. For example, assume the EI model with $\beta_{ij} = 1, \forall i, j$. Then in the worst case, $\rho(\tilde{x}_{NE}) = n$. When the number of players n is large, $\tilde{G}(\tilde{x}_{NE})/G^*$ may be much larger than 1.

From this discussion, we see that the technology improvement may not offset the negative effect of the lack of incentives, and solving the incentive problem may be more important than merely counting on new technologies.

3.4 Correlated equilibrium (CE)

Correlated equilibrium (CE) [18] is a more general notion of equilibrium which includes the set of NE. In some cases, CE has a "coordination effect" that results in better outcomes than all NE's [18]. We consider the performance bounds of CE in this section.

Conceptually, one may think of a CE as being implemented with the help of a mediator [19]. Let μ be a probability distribution over the strategy profiles $\mathbf{x}$. First the mediator selects a strategy profile $\mathbf{x}$ with probability $\mu(\mathbf{x})$. Then the mediator confidentially recommends to each player i the component x_i in this strategy profile. Each player i is free to choose whether to obey the mediator's recommendations. μ is a CE iff it would be a Nash equilibrium for all players to obey the mediator's recommendations. Note that given a recommended x_i, player i only knows $\mu(\mathbf{x}_{-i}|x_i)$ (i.e., the conditional distribution of other players' recommended strategies given x_i). Then in a CE, x_i should be a best response to the randomized strategies of other players with distribution $\mu(\mathbf{x}_{-i}|x_i)$. CE can also be implemented with a pre-play meeting of the players [17], where they decide the CE μ they will play. Later they use a device which generates strategy profiles $\mathbf{x}$ with the distribution μ and separately tells the i'th component, x_i, to player i.

For simplicity, we focus on CE whose support is on a discrete set of strategy profiles. We call such a CE a *discrete CE*. More formally, μ is a discrete CE iff (1) it is a CE; and (2) the distribution μ only assigns positive probabilities to $\mathbf{x} \in S_\mu$, where S_μ, the support of the distribution μ, is a discrete set of strategy profiles. That is, $S_\mu = \{\mathbf{x}^i \in \mathcal{R}_+^n, i = 1, 2, \ldots, M_\mu\}$, where $\mathbf{x}^i$ denotes a strategy profile, $M_\mu < \infty$ is the cardinality of S_μ and $\sum_{\mathbf{x}\in S_\mu} \mu(\mathbf{x}) = 1$. (But the action set of each player is still $\mathcal{R}_+$.)

First, we need to establish the existence of discrete CE's.

Proposition 9. *Discrete CE's exist in the security game since a pure-strategy NE is clearly a discrete CE, and pure-strategy NE exists (Proposition 2). Also, any randomization over multiple pure-strategy NE's is a discrete CE.*

Remark*: However, discrete CE is not confined to these two types. We will give an example later.*

We first write down the conditions for a discrete CE with the general cost function

$$g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i, \forall i. \tag{3.24}$$

If μ is a discrete CE, then for any x_i with a positive marginal probability (i.e., $(x_i, \tilde{\mathbf{x}}_{-i}) \in S_\mu$

for some $\tilde{\mathbf{x}}_{-i}$), x_i is a best response to the conditional distribution $\mu(\mathbf{x}_{-i}|x_i)$, i.e., $x_i \in \arg\min_{x'_i \in \mathcal{R}_+} \sum_{\mathbf{x}_{-i}} [f_i(x'_i, \mathbf{x}_{-i}) + x'_i] \mu(\mathbf{x}_{-i}|x_i)$. (Recall that player i can choose his investment from $\mathcal{R}_+$.) Since the objective function in the right-hand-side is convex and differentiable in x'_i, the first-order condition is

$$\begin{cases} \sum_{\mathbf{x}_{-i}} \frac{\partial f_i(x_i, \mathbf{x}_{-i})}{\partial x_i} \mu(\mathbf{x}_{-i}|x_i) + 1 = 0 & \text{if } x_i > 0 \\ \sum_{\mathbf{x}_{-i}} \frac{\partial f_i(x_i, \mathbf{x}_{-i})}{\partial x_i} \mu(\mathbf{x}_{-i}|x_i) + 1 \geq 0 & \text{if } x_i = 0 \end{cases} \tag{3.25}$$

where $\sum_{\mathbf{x}_{-i}} \frac{\partial f_i(x_i, \mathbf{x}_{-i})}{\partial x_i} \mu(\mathbf{x}_{-i}|x_i)$ can also be simply written as $E_\mu(\frac{\partial f_i(x_i, \mathbf{x}_{-i})}{\partial x_i}|x_i)$.

3.4.1 Example

Now we give an example of CE to illustrate the condition (3.25). It also demonstrates that a CE needs not be a NE or a randomization over multiple NE's

Consider the EI model with only 2 players, with cost functions $g_1(\mathbf{x}) = f(x_1 + \alpha \cdot x_2) + x_1$, and $g_2(\mathbf{x}) = f(x_2 + \alpha \cdot x_1) + x_2$, where $\alpha > 1, \mathbf{x} \geq 0$. (Note that the cost functions of the two players are symmetric.) We compute the pure NE's first. Assume that there exists $y_{NE} > 0$ such that $f'(y_{NE}) + 1 = 0$. Then the best response of player 1 to x_2 is $BR_1(x_2) = (y_{NE} - \alpha \cdot x_2)_+$, and the best response of player 2 to x_1 is $BR_2(x_1) = (y_{NE} - \alpha \cdot x_1)_+$. Then there are 3 pure-strategy NE's, shown in Fig. 3.5.

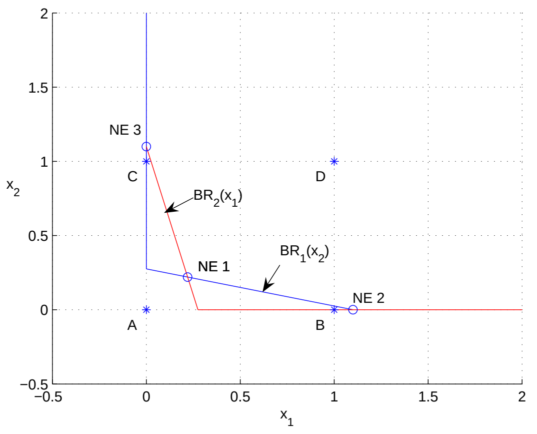


Figure 3.5: Pure-strategy NE's and a discrete CE

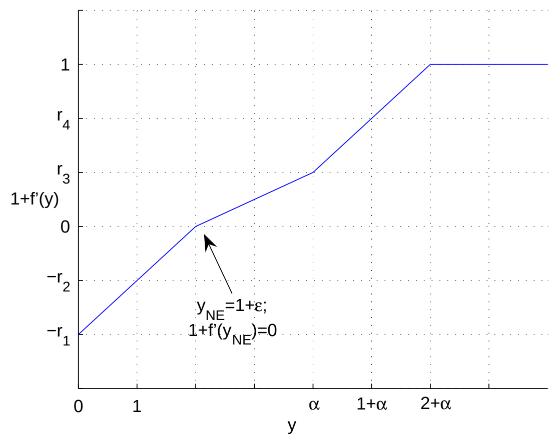


Figure 3.6: The shape of $1 + f'(y)$

Denote by A, B, C, D the profiles (0,0), (1,0), (0,1), (1,1) respectively (Fig. 3.5). We would like to construct a CE where only these profiles have positive probability and $\mu(A) : \mu(B) : \mu(C) : \mu(D) = 1 : \beta_1 : \beta_1 : \beta_1\beta_2$, where $\beta_1, \beta_2 > 1$.

Consider player 1 (the argument for player 2 is similar), we have $\frac{\partial g_1(\mathbf{x})}{\partial x_1} = f'(x_1+\alpha x_2)+1$. Assume that

$$\begin{aligned} \frac{\partial g_1(A)}{\partial x_1} &= f'(0) + 1 = -r_1 \\ \frac{\partial g_1(B)}{\partial x_1} &= f'(1) + 1 = -r_2 \\ \frac{\partial g_1(C)}{\partial x_1} &= f'(\alpha) + 1 = r_3 \\ \frac{\partial g_1(D)}{\partial x_1} &= f'(1+\alpha) + 1 = r_4 \end{aligned} \tag{3.26}$$

where $r_1, r_2, r_3, r_4 > 0$, $r_1 > r_2$, $r_3 < r_4$ (consistent to the convexity of $f(\cdot)$) and satisfy

$$r_1 = \beta_1 r_3 \text{ and } r_2 = \beta_2 r_4. \tag{3.27}$$

Proposition 10. *If (3.26) and (3.27) holds, then μ is a CE.*

Proof. By (3.26) and (3.27), we have

$$\begin{aligned}
& \mu(A|x_1 = 0)\frac{\partial g_1(A)}{\partial x_1} + \mu(C|x_1 = 0)\frac{\partial g_1(C)}{\partial x_1} \\
\propto\ & \mu(A)\frac{\partial g_1(A)}{\partial x_1} + \mu(C)\frac{\partial g_1(C)}{\partial x_1} \\
\propto\ & -r_1 + \beta_1 r_3 = 0
\end{aligned}$$

and

$$\begin{aligned}
& \mu(B|x_1 = 1)\frac{\partial g_1(B)}{\partial x_1} + \mu(D|x_1 = 1)\frac{\partial g_1(D)}{\partial x_1} \\
\propto\ & \mu(B)\frac{\partial g_1(B)}{\partial x_1} + \mu(D)\frac{\partial g_1(D)}{\partial x_1} \\
\propto\ & -\beta_1 r_2 + \beta_1\beta_2 r_4 = 0.
\end{aligned}$$

Therefore, by condition (3.25), it is the best response of player 1 to obey the recommended actions (0 or 1) from the distribution μ. Due to symmetry of the cost functions and the distribution μ, player 2 also obeys the recommended actions. Therefore μ is a CE. □

Clearly, there exist functions $f(y)$ that satisfy (3.26) and (3.27). Fig. 3.6 shows $1+f'(y)$ of such a function.

3.4.2 How good can a CE get?

The next question we would like to understand is: does there always exist a CE that achieves the social optimum in the security game? In other words, can we always "coordinate" the players' actions using CE in order to achieve the SO? The answer is given below.

Proposition 11. *In general, there does not exist a CE that achieves the social optimum in the security game.*

Proof. Suppose that there is a unique $\mathbf{x}^* > 0$ that minimizes the social cost. If a CE achieves SO, then the CE should have probability 1 on the profile $\mathbf{x}^*$. In other words, each time, the mediator chooses $\mathbf{x}^*$ and recommends x_i^* to player i. Then, we have

$$\sum_k \frac{\partial f_k(\mathbf{x}^*)}{\partial x_i} = -1$$

Since $\sum_k \frac{\partial f_k(\mathbf{x}^*)}{\partial x_i} \leq \frac{\partial f_i(\mathbf{x}^*)}{\partial x_i}$, we have $\frac{\partial g_i(\mathbf{x}^*)}{\partial x_i} = \frac{\partial f_i(\mathbf{x}^*)}{\partial x_i} + 1 \geq 0$. If the inequality is strict, then player i has incentive to invest less than x_i^*. Therefore in general, CE cannot achieve SO in this game. □

Proposition 12. *But, a CE can be "better" than all pure-strategy NE's in the security game.*

Remark: Similar results hold for games with finite action sets [18]. Note that the security game we study here is different in that the action set of each player is $\mathcal{R}_+$ which is not a finite set.

Consider the example in the last section where $1 + f'(y)$ is shown in Fig. 3.6. For simplicity, we assume that $1+f'(y)$ is piecewise linear, and satisfies $1+f'(1+\epsilon) = 0$ (where $\epsilon > 0$), $1 + f'(2+\alpha) = 1$, and $f(\infty) = 0$. Then $f(1+\alpha), f(\alpha), f(1)$ and $f(0)$ can be computed according to Fig. 3.6.

In the CE μ, the expected cost of player 1 is

$$\begin{aligned} & E_\mu(g_1(\mathbf{x})) \\ = \; & \frac{1}{1+2\beta_1+\beta_1\beta_2}\{f(0) + \beta_1[f(1)+1] + \\ & \beta_1 f(\alpha) + \beta_1\beta_2[f(1+\alpha)+1]\} \end{aligned}$$

and by symmetry, $E_\mu(g_2(\mathbf{x})) = E_\mu(g_1(\mathbf{x}))$. So the expected social cost is $E_\mu(g_1(\mathbf{x})+g_2(\mathbf{x})) = 2E_\mu(g_1(\mathbf{x}))$.

Also, since $1 + f'(1+\epsilon) = 0$, we have $y_{NE} = 1+\epsilon$. From here the social costs of all three pure-strategy NE's in Fig. 3.5 can be obtained.

Let $\alpha = 5$, $\beta_1 = 8, \beta_2 = 4$, $r_1 = 2.4, r_2 = 2, r_3 = 0.3, r_4 = 0.5$, $\epsilon = 1$. Then, it can be computed that the expected social cost at the CE is $E_\mu(g_1(\mathbf{x}) + g_2(\mathbf{x})) = 4.351$. And the social costs at the NE 1, NE 2 and NE 3 in Fig. 3.5 are 7.467, 5.4 and 5.4 respectively. Therefore the CE is "better" than all pure-strategy NE's.

3.4.3 The worst-case discrete CE

In contrast to the last section, now we consider the POA of discrete CE, which is defined as the performance ratio of the worst discrete CE compared to the SO. In the EI model and BT model, we show that the POA of discrete CE is identical to that of pure-strategy NE derived before, although the set of discrete CE's is larger than the set of pure-strategy NE's in general.

First, the following lemma can be viewed as a generalization of Lemma 1.

Lemma 3. *With the general cost function (3.24), the POA of discrete CE, denoted as ρ_{CE}, satisfies*

$$\rho_{CE} \leq \max_{\mu \in \mathcal{C}_D} \{\max\{1, \max_k [E_\mu(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_k})]\}\}$$

where $\mathcal{C}_D$ is the set of discrete CE's, the distribution μ defines a discrete CE, and the expectation is taken over the distribution μ.

The proof of Lemma 3 (shown in section 7.4) is similar to that of Lemma 1, although the distribution μ can be complicated.

Proposition 13. *In the EI model and the BT model, the POA of discrete CE is the same as the POA of pure-strategy NE. That is, in the EI model,*

$$\rho_{CE} \leq \max_k \{1 + \sum_{i:i \neq k} \beta_{ki}\},$$

and in the BT model,

$$\rho_{CE} \leq (1 + \max_{(i,j):i \neq j} \frac{v_i r_{ji}}{v_j r_{ij}}).$$

The proof is in section 7.5.

Chapter 4

Extensive-Form Games

4.1 Repeated game

In the repeated game, the strategic-form game Γ is repeated in each "period". As in Γ, we assume that in each period, some cost on security investment is incurred for player i and he has a security risk $f_i(\cdot)$ which depends on all players' investments in this period. In this case, the players have more incentives to cooperate for their long term interests. In this section we consider the performance gain provided by the repeated game.

We make the following standard definitions in the repeated game.

The game Γ is repeated in period $0, 1, 2, \ldots$. Let x_i^t be the action of player i in period t, and let $\mathbf{x}^t$ be the vector of actions of all players in period t. Define h^{t+1}, the history at the end of period t, to be the sequence of actions in the previous periods:

$$h^{t+1} := (\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^t).$$

And h^0 is an empty history. Let the set of possible histories at the end of period t be H^t.

A pure strategy for player i is a sequence of maps $\{s_i^t\}_{t=0,1,\ldots}$, where each s_i^t maps H^t to his feasible action set $\mathcal{R}_+$. Then, $s_i^t(h^t)$ is player i's action in period t specified by his strategy, given the history h^t. For simplicity, we write $s_i^t(h^t)$ as $s_i(h^t)$. Also, let the vector $\mathbf{s}(h^t) = (s_i(h^t))_{i=1}^n$.

We say that player i has *deviated* in period t if $x_i^t \neq s_i(h^t)$.

Players discount the future. Specifically, player i's cost in the repeated game is

$$g_i^\infty := (1-\delta)\sum_{i=0}^{\infty}[g_i(\mathbf{x}^t)\cdot\delta^t]$$

where $\delta \in (0,1)$ is the "discount factor".

Define $\underline{g_i}$, the "reservation cost" of player i, as

$$\underline{g_i} := \min_{x_i \geq 0} g_i(\mathbf{x}) \text{ given that } x_j = 0, \forall j \neq i$$

and we denote $\underline{x_i}$ as a minimizer. $\underline{g_i} = g_i(x_i = \underline{x_i}, \mathbf{x}_{-i} = \mathbf{0})$ is the minimal cost achievable by player i when other players are punishing him by making minimal investments 0. Let the reservation cost vector $\underline{\mathbf{g}} := (\underline{g_i})_{i=1}^n$.

By Proposition 1, we assume that $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$ without loss of generality. Also, we make the following additional assumptions in this section:

1. $f_i(\mathbf{x})$ (and $g_i(\mathbf{x})$) is *strictly* convex in x_i if $\mathbf{x}_{-i} = \mathbf{0}$. So $\underline{x_i}$ is unique.
2. $\frac{\partial g_i(\mathbf{0})}{\partial x_i} < 0$ for all i. So, $\underline{x_i} > 0$.
3. For each player, $f_i(\mathbf{x})$ is strictly decreasing with x_j for some $j \neq i$. That is, positive externality exists.

By assumption 2 and 3, we have $g_i(\underline{\mathbf{x}}) < g_i(x_i = \underline{x_i}, \mathbf{x}_{-i} = \mathbf{0}) = \underline{g_i}, \forall i$. Therefore there exists $\mathbf{x}' \geq \mathbf{0}$ so that $\mathbf{g}(\mathbf{x}') < \underline{\mathbf{g}}$ (where the strict inequality is element-wise throughout the chapter).

4.1.1 A performance bound on the best SPE

The Folk Theorem by Fudenberg and Maskin (Theorem 5.4 in [17]) provides a Subgame Perfect Equilibrium (SPE) in repeated games with discounted costs (as we assumed above) to support any feasible cost vector $\mathbf{g} < \underline{\mathbf{g}}$, when the discount factor δ is sufficiently close to 1. Assume that $\mathbf{g}(\mathbf{x}') = \mathbf{g}$ where $\mathbf{x}' \geq \mathbf{0}$. In particular, they defined the strategies for the players so that in the SPE outcome, each player i sticks to the investment x'_i, so that $g_i^\infty = g_i < \underline{g_i}$.[1]

Therefore, the set of SPE is quite large in general. By negotiating with each other, the players can agree on some SPE. In this section, we are interested in the performance of the "socially best SPE" that can be supported, that is, the SPE with the minimum social cost (denoted as G_E). Such a SPE is "optimal" for the society, provided that it is also rational for individual players. We will compare it to the social optimum by considering the "performance ratio" $\gamma = G_E/G^*$, where G^* is the optimal social cost, and

$$\begin{aligned} G_E = \quad & \inf_{\mathbf{x} \geq \mathbf{0}} \quad \textstyle\sum_i g_i(\mathbf{x}) \\ & \text{s.t.} \qquad g_i(\mathbf{x}) < \underline{g_i}, \forall i \end{aligned} \tag{4.1}$$

Since $g_i(\cdot)$ is convex by assumption, due to continuity,

$$\begin{aligned} G_E = \quad & \min_{\mathbf{x} \geq \mathbf{0}} \quad \textstyle\sum_i g_i(\mathbf{x}) \\ & \text{s.t.} \qquad g_i(\mathbf{x}) \leq \underline{g_i}, \forall i \end{aligned} \tag{4.2}$$

where $g_i(\mathbf{x}) \leq \underline{g_i}$ is the rationality constraint for each player i. Denote by $\mathbf{x}_E$ a solution of (4.2). Then $\sum_i g_i(\mathbf{x}_E) = G_E$.

Recall that $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$, where the investment x_i has been normalized such that its coefficient (unit cost) is 1. Then, to solve (4.2), we form a partial Lagrangian

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \lambda') \quad &:= \quad \textstyle\sum_k g_k(\mathbf{x}) + \sum_k \lambda'_k [g_k(\mathbf{x}) - \underline{g_k}] \\ &= \quad \textstyle\sum_k (1 + \lambda'_k) g_k(\mathbf{x}) - \sum_k \lambda'_k \underline{g_k} \end{aligned}$$

and pose the problem $\max_{\lambda' \geq \mathbf{0}} \min_{\mathbf{x} \geq \mathbf{0}} \mathcal{L}(\mathbf{x}, \lambda')$.

Let λ be the vector of dual variables when the problem is solved (i.e., when the optimal solution $\mathbf{x}_E$ is reached). Then differentiating $\mathcal{L}(\mathbf{x}, \lambda')$ in terms of x_i, we have the optimality

[1]In the game defined above, x_i and $g_i(\mathbf{x})$ have no upper-bound. So the proof of SPE is somewhat different. However, one can show that the one-stage deviation principle still holds (under the strategies), which can then be used to establish the SPE similar to the proof of Theorem 5.4 in [17].

condition

$$\begin{cases} \sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] = 1+\lambda_i & \text{if } x_{E,i} > 0 \\ \sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] \leq 1+\lambda_i & \text{if } x_{E,i} = 0 \end{cases} \tag{4.3}$$

Proposition 14. *The performance ratio γ is upper-bounded by $\gamma = G_E/G^* \leq \max_k\{1+\lambda_k\}$. (The proof is given in section 7.2.)*

This result can be understood as follows: if $\lambda_k = 0$ for all k, then all the incentive-compatibility constraints are not active at the optimal point of (4.2). So, individual rationality is not a constraining factor for achieving the social optimum. In this case, $\gamma = 1$, meaning that the best SPE achieves the social optimal. But if $\lambda_k > 0$ for some k, the individual rationality of player k prevent the system from achieving the social optimum. Larger λ_k leads to a poorer performance bound on the best SPE relative to SO.

Proposition 14 gives an upper bound on γ assuming the general cost function $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$. Although it is applicable to the two specific models introduced before, it is not explicitly related to the network parameters. In the following, we give an explicit bound for the EI model.

Proposition 15. *In the EI model where $g_i(\mathbf{x}) = V_i(\sum_{j=1}^n \beta_{ji}x_j) + x_i$, γ is bounded by*

$$\gamma \leq \min\{\max_{i,j,k} \frac{\beta_{ik}}{\beta_{jk}}, Q\}$$

where $Q = \max_k\{1 + \sum_{i:i\neq k} \beta_{ki}\}$.

The part $\gamma \leq Q$ is straightforward: since the set of SPE includes all NE's, the best SPE must be better than the worst NE. The other part is derived from Proposition 14 (its proof is included in section 7.3).

Note that the inequality $\gamma \leq \max_{i,j,k} \frac{\beta_{ik}}{\beta_{jk}}$ may not give a tight bound, especially when β_{jk} is very small for some j, k. But in the following simple example, it is tight and shows that the best SPE achieves the social optimum. Assume n players, and $\beta_{ij} = 1, \forall i, j$. Then, the POA in the strategic-form game is $\rho \leq Q = n$ according to (3.14). In the repeated game, however, the performance ratio $\gamma \leq \max_{i,j,m} \frac{\beta_{im}}{\beta_{jm}} = 1$ (i.e., the social optimum is achieved). This illustrates the performance gain resulting from the repeated game.

It should be noted that, however, although repeated games can provide much better performance, they usually require more communication and coordination among the players than strategic-form games.

4.1.2 Folk Theorem with local punishments and rewards

In this section, our study is more on the game theory itself.

In the usual Folk theorem, the strategy structure proposed by Fudenberg and Maskin (Theorem 5.4 in [17]) requires that, in general, all players change their actions to punish a misbehaving player and then reward those who have punished him correctly.

In a large network (or a game with many players), however, it increases the coordination overhead to require that all players be involved in the punishment and rewarding actions. In this section, we are interested in constructing a SPE which only relies on "local" punishments and rewards, only involving the player who deviates and his "neighbors" (to be defined more precisely below). Also, it seems more reasonable to allow *simultaneous*, but local punishments and rewards in large networks, which is another major difference from the aforementioned strategy structure. We show that this SPE supports the same payoff (or cost) region as the usual Folk Theorem.

Still, our SPE requires that all actions are observable to all players as in the usual Folk Theorem. In that sense, the SPE is not "fully localized". However, the result in this section, we believe, is an interesting first step to explore whether there exists a SPE which only requires local observations and local reactions, but can still achieve the same payoff region as the usual Folk Theorem.

We make the following mild assumptions.

1. For any player i, the cost function $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$ is continuous in x_i. The action $x_i \in [0, x_{max}]$. [2] And $0 \leq f_i(\mathbf{x}) < \infty, \forall \mathbf{x}$.

2. x_{max} is large enough: $x_{max} \geq f_i(\mathbf{0}), \forall i$.

[2]Here, we define the upper bound x_{max} so that the one-stage deviation principle can be directly used, which simplifies the proof.

3. $g_i(\mathbf{x})$ only depends on x_i and x_j for all $j \in \mathcal{N}(i)$. More formally, let $\mathcal{S}(i) = \{i\} \cup \mathcal{N}(i)$ and $\mathcal{U}(i)$ be the set of all other players not in $\mathcal{S}(i)$. Then $\frac{\partial g_i(\mathbf{x})}{\partial x_k} = 0, \forall k \in \mathcal{U}(i), \forall \mathbf{x}$.

4. Assume $i \in \mathcal{N}(j)$ iff $j \in \mathcal{N}(i)$. (Therefore, the dependent relationships among the players are symmetric and can be represented by an undirectional graph, such as the ones in Fig 4.1.)

5. There are positive externalities from the neighbors, i.e., $\frac{\partial g_i(\mathbf{x})}{\partial x_j} \leq 0, \forall j \in \mathcal{N}(i), \forall \mathbf{x}$.

Theorem 1. Folk Theorem with local punishments and rewards. *Under the above assumptions, in a repeated game with observable actions*[3]*, any feasible vector* $\mathbf{g} < \underline{\mathbf{g}}$ *(i.e.,* $g_i < \underline{g_i}, \forall i$*) can be supported by a SPE with only "local" punishments and rewards, if the discount factor* δ *is large enough.*

"Local punishments and rewards" mean that, if node i deviates in a period, only players within his L-hop neighborhood need to change their actions in response to the deviation, where L is a constant.

The key to the proof is the construction of punishments and rewards that only involve local players.

Assume that the profile $\mathbf{x}' = (x'_i)_{i=1}^n$ achieves the cost vector $\mathbf{g} < \underline{\mathbf{g}}$. That is, $g_i(\mathbf{x}') = g_i, \forall i$.

(i) Local punishment: Player i is punished by his neighbors: Any player $j \in \mathcal{N}(i)$ lets $x_j = 0$, and player i lets $x_i = \underline{x_i}$ to get his reservation cost $\underline{g_i}$. (Recall that $g_i(\mathbf{x})$ does not depend on $x_k, \forall k \in \mathcal{U}(i)$.)

(ii) Local rewarding: Player i rewards his neighbors. Any player $j \in \mathcal{N}(i)$ lets $x_j = x'_j$, and player i lets $x_i = \hat{x}_i > x'_i$. For $j \in \mathcal{N}(i)$, define

$$\hat{g_j}(i) := g_j(\hat{x}_i, \mathbf{x}'_{-i}) \leq g_j(x'_i, \mathbf{x}'_{-i}) = g_j \tag{4.4}$$

by assumptions 5.

[3] *That is, the actions of each player can be observed by all other players.*

$\theta_i(h^\tau)$	A	$B(i,\cdot)$	$B(j,\cdot), j \neq i$	$C(i)$	$C(j), j \neq i$
$s_i(h^\tau)$	x_i'	$\underline{x_i}$	0	$\hat{x}_i$	x_i'

Table 4.1: Mapping from the "state" to the action

Also, $\hat{x}_i$ is chosen such that

$$\hat{g}_i(i) := g_i(\hat{x}_i, \mathbf{x}'_{-i}) = g_i + \Delta < \underline{g_i} \tag{4.5}$$

for some $\Delta > 0$. (Such a Δ exists since $g_i < \underline{g_i}$.) Lemma 4 below shows that such a $\hat{x}_i$ exists.

Lemma 4. *There exists $\hat{x}_i > x_i'$ such that (4.5) holds.*

Proof. From assumption 2 and the form of $g_i(\mathbf{x})$, we know that $g_i(x_i = x_{max}, \mathbf{x}_{-i} = \mathbf{x}'_{-i}) \geq x_{max} \geq f_i(\mathbf{0}) = g_i(\mathbf{0}) \geq \underline{g_i}$. Since $g_i(\mathbf{x})$ is continuous in x_i, $g_i(x_i = x_{max}, \mathbf{x}_{-i} = \mathbf{x}'_{-i}) \geq \underline{g_i}$ (just proved) and $g_i(\mathbf{x}') = g_i < \underline{g_i}$, we know that there exists an $\hat{x}_i > x_i'$ such that (4.5) holds. $\square$

Proof of Theorem 1:

We first describe the strategy, and then show that it is a SPE.

I. Strategy

To define the strategy, we need to specify what is $s_i(h^\tau)$ given h^τ, $\tau = 0, 1, 2, \ldots$. For this, we define $\theta_i(h^\tau)$ as the "state" of player i in period τ given the history h^τ. The possible states are given in Table 4.1, where the second argument of $B(i, \cdot)$ is a counter which will be made clear later. Also, $s_i(h^\tau)$ is directly obtained from $\theta_i(h^\tau)$ according to Table 4.1.

Write $\theta(h^\tau) := (\theta_i(h^\tau))_{i=1}^n$. The remaining task is to specify $\theta(h^\tau)$.

$\theta(h^\tau)$ is computed recursively: first, let $\theta_i(h^0) = A, \forall i$; then, for $t = 0, 1, 2 \ldots, \tau - 1$, compute $\theta(h^{t+1})$ from $\theta(h^t)$ and $\mathbf{x}^t$ (i.e., the actions in period t) by the following three steps.

Step 1: Punishment actions

Denote the set of players who have deviated in period t as $\mathcal{D}^t$. Define the following subset of $\mathcal{D}^t$:

$$\mathcal{P}^t := \{j \in \mathcal{D}^t | \mathcal{S}(j) \cap \mathcal{S}(j') = \emptyset, \forall j' \in \mathcal{D}^t, j' \neq j\}. \tag{4.6}$$

$\mathcal{P}^t$ is the set of players who will be "punished" for their deviations in period t. Definition (4.6) ensures the following.

Lemma 5. *If there is a single player j who deviates in period t, then $j \in \mathcal{P}^t$.*

For any player $j \in \mathcal{P}^t$, let $\theta_i(h^{t+1}) = B(j, N_j), \forall i \in \mathcal{S}(j)$, where N_j is chosen such that

$$\min_{\mathbf{x}} g_j(\mathbf{x}) + N_j \cdot \underline{g_j} > \max_{\mathbf{x}} g_j(\mathbf{x}) + N_j \cdot \hat{g}_j(j) \tag{4.7}$$

where the *min* and *max* are taken over the domain $\mathbf{x} \in [0, x_{max}]^n$. Such an N_j exists since $\hat{g}_j(j) < \underline{g_j}$ according to (4.5).

According to Table 4.1, if $j \in \mathcal{P}^t$, we have $s_j(h^{t+1}) = \underline{x_j}$ and $s_k(h^{t+1}) = 0, \forall k \in \mathcal{N}(j)$. This means that the neighbors of player j should begin to "punish" player j in period $t+1$.

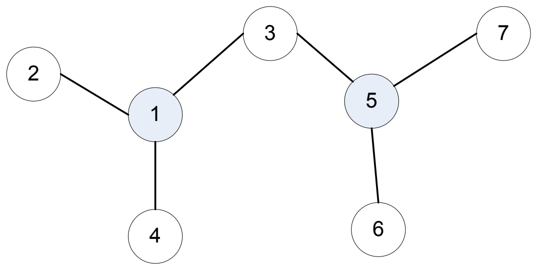
Step 2: Reverting actions

For $j \in \mathcal{P}^t$, define the set

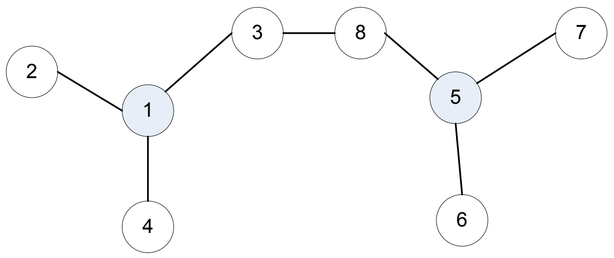
$$\mathcal{I}_j^t := \{i | i \neq j; \theta_i(h^t) = B(i, r) \text{ where } r \leq N_i; \mathcal{S}(i) \cap \mathcal{S}(j) \neq \emptyset\}. \tag{4.8}$$

For all $k \in \mathcal{S}(i) \backslash \mathcal{S}(j)$ where $i \in \mathcal{I}_j^t, j \in \mathcal{P}^t$, let $\theta_k(h^{t+1}) = A$ (i.e., the state of player k is "reverted" to A from $B(i, r)$). Therefore, $s_k(h^{t+1}) = x'_k$.

For example, in Fig 4.1 (a), $\mathcal{S}(1) = \{1, 2, 3, 4\}$, $\mathcal{S}(5) = \{3, 5, 6, 7\}$. Suppose that players in $\mathcal{S}(1)$ are in state $B(1, r)$ (where $1 < r \leq N_1$) in period t (i.e., player 2, 3, 4 are "punishing" player 1 due to his earlier deviation). And suppose that player 5 is the only player who deviates in period t. Then, $\mathcal{P}^t = \{5\}$, $\mathcal{I}_5^t = \{1\}$. So, in period $t+1$, the states of player 3, 5, 6, 7 are $B(5, N_5)$; and the state of player 1, 2, 4 is A. This means that the punishment of player 1 is ended if some player in $\mathcal{S}(1)$ needs to begin punishing another player. In Fig 4.1 (b), however, the set $\mathcal{S}(1)$ and $\mathcal{S}(5)$ are disjoint. Under the same assumptions for period t,



(a) A scenario where simultaneous punishmentments are not allowed



(b) A scenario where simultaneous punishmentments are allowed

Figure 4.1: Illustration of simultaneous punishmentments

then in period $t+1$, the states of all players in $\mathcal{S}(1)$ are $B(1, r-1)$ and the states of all players in $\mathcal{S}(5)$ are $B(5, N_5)$. In other words, player 1 and 5 are punished simultaneously.

Note that these "reactions" (i.e., changes of states in response to deviations) in step 1 and 2 are *local* in the sense that a deviator only affects the states of players within 3 hops.

Step 3: Actions of other players

For any player j whose state $\theta_j(h^{t+1})$ has not been assigned in step 1 or 2, do the following.

(i) If $\theta_j(h^t) = B(i, r)$ for some i, let

$$\theta_j(h^{t+1}) = \begin{cases} B(i, r-1) & \text{if } r > 1 \\ C(i) & \text{if } r = 1 \end{cases}.$$

(ii) If $\theta_j(h^t) = A$ or $C(i)$ for some i, then let $\theta_j(h^{t+1}) = \theta_j(h^t)$, i.e., the state is unchanged.

II. SPE

We now show that the above strategy is a SPE, i.e., unilateral deviation in any subgame will not yield any benefit for the deviator. Using the one-stage-deviation principle, we only need to consider the case when one player deviates in a single period and follows the strategy afterwards.

Now, given any history $h^{t+1} = (\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^t)$. In the following, we consider a deviation by player i in period $t+1$ who follows the strategy afterwards. Other players always follow their strategies after period t. *We say that a player "deviates from state S" if his state is S in period $t+1$ but his action does not conform to it.*

Lemma 6. *Given any history h^{t+1}, with or without the above one-stage deviation by player i, eventually the states of all players become either A or C. Therefore $x_i \geq x_i', \forall i$ eventually.*

Proof. Note that in each period the state of any player is among A, B or C. Starting with period $t+2$, all players follow their strategies. Eventually, all players with state B change to state C. □

Assume that player i's state is A in period $t+1$ (i.e., $\theta_i(h^{t+1}) = A$). If he conforms to it (and other players follow their strategies), he will never be punished and will remain in state A. According to Lemma 6, he will eventually receive a cost which is not higher than g_i. If he deviates, by Lemma 5 he will be punished in state $B(i, \cdot)$ in the following N_i periods; and will receive a cost of $\hat{g}_i(i) := g_i + \Delta$ later in state $C(i)$ which lasts for ever. If the discount factor δ is large enough, the cost at the "tail" dominates, so there is no benefit for him to deviate.

Assume that player i's state is $B(j,r), j \neq i$ in period $t+1$ for some r. If he conforms, he will eventually receive a cost not more than $\hat{g}_i(j) \leq g_i$ in state $C(j)$ (after the states of all players become A or C, by Lemma 6). If he deviates, he will eventually receive a cost $\hat{g}_i(i) := g_i + \Delta$ in state $C(i)$. So there is no incentive to deviate.

Assume that player i's state is $B(i,r)$ in period $t+1$ for some r. If he conforms, his cost is $\underline{g_i}$ when his state is $B(i, \cdot)$ (for less than T_i periods), and $\hat{g}_i(i) := g_i + \Delta$ after his state changes to $C(i)$. If he deviates in period $t+1$, his cost is at least $\underline{g_i}$ in this period (since $\underline{x_i}$ is his best response when the actions of its neighbors are 0), and he will then receive a cost of $\underline{g_i}$ for T_i periods and later receive a cost of $\hat{g}_i(i) := g_i + \Delta$ in state $C(i)$. Since $\underline{g_i} > \hat{g}_i(i)$, there is no benefit to deviate.

Assume that player i's state is $C(j), j \neq i$ in period $t+1$. If he conforms, eventually his cost is at most $\hat{g}_i(j) \leq g_i$ (after the states of all players become A or C, by Lemma 6). Otherwise, he eventually has a cost of $\hat{g}_i(i) := g_i + \Delta$ in state $C(i)$. So he should conform.

Assume that player i's state is $C(i)$ in period $t+1$. If he conforms, his cost is $\hat{g}_i(i) := g_i + \Delta$. Otherwise, his expected cost would be at least

$$g_{i,D} := (1-\delta)\min_{\mathbf{x}} g_i(\mathbf{x}) + \delta(1-\delta^{N_i})\underline{g_i} + \delta^{N_i+1}\hat{g}_i(i).$$

Note that

$$\begin{aligned}
& g_{i,D} - \hat{g}_i(i) \\
= \; & (1-\delta)\min_{\mathbf{x}} g_i(\mathbf{x}) + \delta(1-\delta^{N_i})\underline{g_i} - (1-\delta^{N_i+1})\hat{g}_i(i) \\
= \; & (1-\delta)[\min_{\mathbf{x}} g_i(\mathbf{x}) - (g_i+\Delta)] + \delta(1-\delta^{N_i})[\underline{g_i} - \hat{g}_i(i)] \\
\geq \; & (1-\delta)[\min_{\mathbf{x}} g_i(\mathbf{x}) - \max_{\mathbf{x}} g_i(\mathbf{x})] + \delta(1-\delta^{N_i})[\underline{g_i} - \hat{g}_i(i)] \\
= \; & (1-\delta)\{[\min_{\mathbf{x}} g_i(\mathbf{x}) - \max_{\mathbf{x}} g_i(\mathbf{x})] + \\
& (\delta + \delta^2 + \cdots + \delta^{N_i})[\underline{g_i} - \hat{g}_i(i)]\}
\end{aligned}$$

When δ approaches 1, this expression approaches $(1-\delta)\{[\min_{\mathbf{x}} g_i(\mathbf{x}) - \max_{\mathbf{x}} g_i(\mathbf{x})] + N_i[\underline{g_i} - \hat{g}_i(i)]\}$, which is strictly positive by (4.7). So $g_{i,D} > \hat{g}_i(i)$: there is no incentive for player i to deviate.

Chapter 5

Improving the Outcome with a "Due Care" Regulation

Recall again that in the strategic-form game $\Gamma = (\mathcal{N}, \mathcal{X}, \mathbf{g})$, the general cost function is $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i, \forall i$.

In the "due care" scheme, a social planner requires that player i invest at least m_i.

For example, UC Berkeley has a minimum security standard for devices connected to the campus network [21]. Requirements include software patch updates, anti-virus software, host-based firewall software, etc. Devices that do not meet the standard are not allowed to connect to the network.

Definition: More formally, the game with a "due care" level of $\mathbf{m}$ is defined as

$$\Gamma_{DC}(\mathbf{m}) = (\mathcal{N}, \mathcal{X}_{DC}, \mathbf{g})$$

where "DC" means "due care", $\mathbf{m} = (m_1, m_2, \dots, m_n) \geq 0$ is the vector of minimum investments, $\mathcal{X}_{DC} = \prod_{i=1}^{n} \mathcal{X}_{DC,i}$ and the action set $\mathcal{X}_{DC,i} = [m_i, +\infty)$.

5.1 Improved bounds

Proposition 16. *Assume that* $\mathbf{x}^* \geq 0$ *achieves the social optimum in game* Γ *(and the optimal social cost is* $G^* = \sum_i g_i(\mathbf{x}^*)$*). Also assume that* $\mathbf{m} \leq \mathbf{x}^*$ *element-wise. Let* $\bar{\mathbf{x}}_{DC}$ *be a NE in* $\Gamma_{DC}(\mathbf{m})$*, i.e., the new game with the due care level* $\mathbf{m}$*. Then,*

$$\sum_i g_i(\bar{\mathbf{x}}_{DC}) - G^* \leq (Q-1)(\mathbf{1}^T\mathbf{x}^* - M) \tag{5.1}$$

where $M = \sum_i m_i$*, and* Q *refers to* Q_{EI} *or* Q_{BT} *depending on the model.*

Remark: In comparison, in the original game Γ we have $G(\bar{\mathbf{x}}) - G^* \leq (Q-1)\cdot(\mathbf{1}^T\mathbf{x}^*)$ by (3.11) where $\bar{\mathbf{x}}$ is a NE in Γ, and the bound is tight (i.e., there exist parameters of the game such that the equality holds).

Therefore, (5.1) shows that the difference bound has improved in $\Gamma_{DC}(\mathbf{m})$. The implication is that if a social planner employs a "due care" scheme which requires that each player i invest at least $m_i \leq x_i^*$, then the outcome could improve.

Proof. We now show that (5.1) can be obtained by utilizing the previous results in a modified game.

Define $x_i' := x_i - m_i \geq 0$. Consider the new game Γ' where the strategies are $\mathbf{x}' \geq 0$, and the cost function is

$$\hat{g}_i(\mathbf{x}') := \hat{f}_i(\mathbf{x}') + x_i', \forall i$$

where

$$\hat{f}_i(\mathbf{x}') := f_i(\mathbf{m} + \mathbf{x}').$$

Notice that $\hat{f}_i(\mathbf{x}')$ is still convex in $\mathbf{x}'$ and non-increasing with every x_i'. Also,

$$g_i(\mathbf{x}' + \mathbf{m}) = \hat{g}_i(\mathbf{x}') + m_i, \forall i \tag{5.2}$$

$$\sum_i g_i(\mathbf{x}' + \mathbf{m}) = \sum_i \hat{g}_i(\mathbf{x}') + M. \tag{5.3}$$

Clearly, the social optimum in Γ' is achieved by $\mathbf{x}^* - \mathbf{m} \geq \mathbf{0}$.

Since $\bar{\mathbf{x}}_{DC}$ is a NE in $\Gamma_{DC}(\mathbf{m})$, therefore $\bar{\mathbf{x}}' := \bar{\mathbf{x}}_{DC} - \mathbf{m}$ is a NE in Γ'. Then, using (3.11), we have

$$\sum_i \hat{g}_i(\bar{\mathbf{x}}') - \sum_i \hat{g}_i(\mathbf{x}^* - \mathbf{m}) \le (Q-1)(\mathbf{1}^T\mathbf{x}^* - M). \tag{5.4}$$

Finally, using (5.3) and (5.4), we have (5.1). $\square$

Corollary 1. *In the special case when $\mathbf{m} = \mathbf{x}^*$ (i.e., when each player is required to invest at least x_i^*), (5.1) implies that $\sum_i g_i(\bar{\mathbf{x}}' + \mathbf{m}) - G^* \le 0$, so the SO is achieved in the NE. We call this setup as the "perfect due care".*

In practice, on the other hand, the social planner normally does not know $\mathbf{x}^*$. However, he could use an estimation of $\mathbf{x}^*$ to set $\mathbf{m}$. Ideally, m_i can be different for different players depending on their social importance. But we show that even with a uniform minimum investment requirement for all players, the difference bound could still improve.

Assume that $m_i = m, \forall i$. If $m = \min_j x_j^* > 0$, then by (5.1), the difference bound improves.

In general, $\mathbf{m} \le \mathbf{x}^*$ does NOT hold, so (5.1) does not apply. In this case, let $\mathcal{I}_+(\mathbf{m}) := \{i | x_i^* > m_i\}$, $D_+(\mathbf{m}) := \sum_{i \in \mathcal{I}_+}(x_i^* - m_i)$; $\mathcal{I}_-(\mathbf{m}) := \{i | x_i^* < m_i\}$, $D_-(\mathbf{m}) := -\sum_{i \in \mathcal{I}_-}(x_i^* - m_i)$; and let $\mathcal{I}_0(\mathbf{m}) := \{i | x_i^* = m_i\}$. Then we have the following.

Proposition 17. *Assume that $\bar{\mathbf{x}}_{DC}$ is a NE in $\Gamma_{DC}(\mathbf{m})$ where $\mathbf{m} \le \mathbf{x}^*$ does not necessarily hold. Then*

$$\Delta(\mathbf{m}) := \sum_i g_i(\bar{\mathbf{x}}_{DC}) - G^* \le (Q-1) \cdot D_+(\mathbf{m}) + D_-(\mathbf{m}). \tag{5.5}$$

Remark: Note that this result includes Prop. 16 as a special case where $\mathcal{I}_-(\mathbf{m}) = \emptyset$, $D_-(\mathbf{m}) = 0$ and $D_+(\mathbf{m}) = \mathbf{1}^T\mathbf{x}^* - M$.

Proof. Denote $\bar{\mathbf{x}}' := \bar{\mathbf{x}}_{DC} - \mathbf{m}$. By (3.11), we have

$$\begin{aligned}\Delta(\mathbf{m}) &= \sum_i \hat{g}_i(\bar{\mathbf{x}}') - \sum_i \hat{g}_i(\mathbf{x}^* - \mathbf{m}) \\ &\leq -(\mathbf{x}^* - \mathbf{m})^T \sum_i \nabla \hat{f}_i(\bar{\mathbf{x}}') - \mathbf{1}^T(\mathbf{x}^* - \mathbf{m}) \\ &\leq \sum_{j \in \mathcal{I}_+(m)} [(x_j^* - m_j)(-\sum_i \frac{\partial \hat{f}_i(\bar{\mathbf{x}}')}{\partial x_j'})] - [D_+(m) - D_-(m)]\end{aligned}$$

Let Q_j be an upper bound of $-\sum_i \frac{\partial \hat{f}_i(\bar{\mathbf{x}}')}{\partial x_j'}$. (In the EI model, $Q_j = 1 + \sum_{i:i \neq j} \beta_{j,i}$. In the BT model, $Q_j = 1 + \max_{i:i\neq j} \frac{v_i r_{ji}}{v_j r_{ij}}$.) Then,

$$\Delta(\mathbf{m}) \leq \sum_{j \in \mathcal{I}_+(m)} (Q_j - 1)(x_j^* - m_j) + D_-(m). \tag{5.6}$$

Since $Q := \max_j Q_j$, we have (5.5) which is a looser bound. □

Assume that $m_i = m \geq 0, \forall i$, i.e., the social planner sets the same minimum investment for all players. We would like to choose m to minimize the upper bound of $\Delta(\mathbf{m}) = \Delta(m\mathbf{1})$ in (5.5). Denote the RHS of (5.5) as $B(\mathbf{m}) = B(m\mathbf{1})$. Then the left-sided derivative of $B(m\mathbf{1})$ is

$$\frac{d^-\Delta(m\mathbf{1})}{dm} = -(Q-1)\cdot[|\mathcal{I}_+(m\mathbf{1})| + |\mathcal{I}_0(m\mathbf{1})|] + |\mathcal{I}_-(m\mathbf{1})|,$$

and the right-sided derivative is

$$\frac{d^+\Delta(m\mathbf{1})}{dm} = -(Q-1)\cdot|\mathcal{I}_+(m\mathbf{1})| + [|\mathcal{I}_-(m\mathbf{1})| + |\mathcal{I}_0(m\mathbf{1})|].$$

Note that $|\mathcal{I}_+(m\mathbf{1})| + |\mathcal{I}_0(m\mathbf{1})| = \sum_{i=1}^n I(x_i^* \geq m)$ where $I(\cdot)$ is the indication function. And $|\mathcal{I}_-(m\mathbf{1})| + |\mathcal{I}_0(m\mathbf{1})| = \sum_{i=1}^n I(x_i^* \leq m)$. Also note that $Q \geq 1$. Therefore, both the left-sided and right-sided derivatives are non-decreasing functions of m. So, $B(m\mathbf{1})$ is convex in m.

When $m = 0$, $d^-\Delta(m\mathbf{1})/dm = -(Q-1)\cdot n \leq 0$. Therefore, $B(m\mathbf{1})$ has a minimum achieved at $m = m_0$ where m_0 satisfies that

$$\frac{d^-\Delta(m_0\mathbf{1})}{dm} \leq 0, \frac{d^+\Delta(m_0\mathbf{1})}{dm} \geq 0. \tag{5.7}$$

Remark: Clearly, letting $m = m_0$ in general leads to a better difference bound than $m = \min_j x_j^*$.

5.2 Numerical example

Next, we give a numerical example to illustrate how the "due care" scheme changes the outcome.

In the EI model, assume that there are $n = 6$ players. Assume that the cost function of player i is

$$\begin{aligned} g_i(\mathbf{x}) &= f_i(\mathbf{x}) + x_i \\ &= (1 - \sum_k \beta_{k,i} x_k)_+ + x_i. \end{aligned}$$

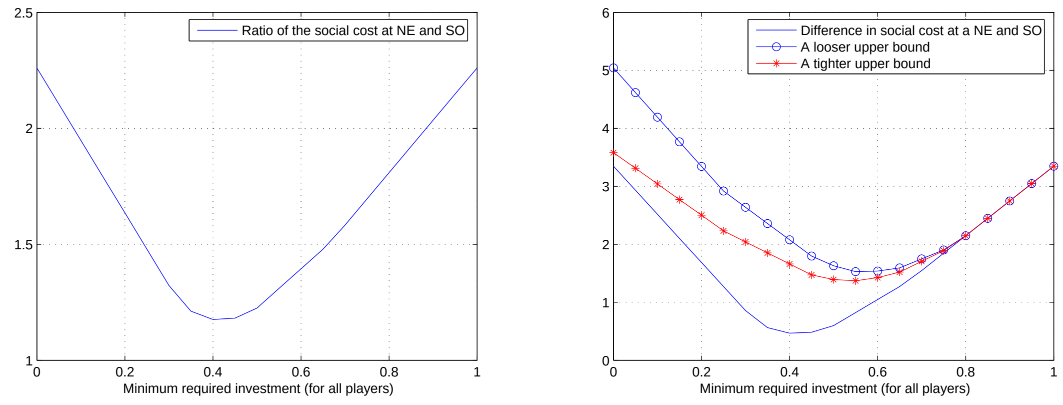
Define the matrix $B := (\beta_{i,j})_{1 \le i,j \le n}$ as

$$B = \begin{pmatrix} 1 & 0 & 0.7 & 0 & 0.3 & 0.3 \\ 0 & 1 & 1 & 0 & 0.5 & 0.4 \\ 0.5 & 0.6 & 1 & 0 & 0.3 & 0.5 \\ 0 & 1 & 0 & 1 & 0.6 & 0 \\ 0 & 0 & 0 & 0.6 & 1 & 0 \\ 0 & 0.6 & 0 & 0.4 & 0 & 1 \end{pmatrix}$$

which is arbitrarily chosen (with $\beta_{i,i} = 1, \forall i$). From B, it is easy to compute that the POA is $Q_{EI} = 2.9$.

Then, we set the minimum investments vector $\mathbf{m} = m\mathbf{1}$ where $m = 0, 0.05, 0.1, \ldots, 0.95, 1$. For each value of m, we numerically find a NE, compute its social cost, and compare it with the SO. Fig. 5.1 shows the ratio and difference between the social costs at the NE and the SO, for different values of m. In Fig. 5.1, we have also plotted the looser and tighter upper bounds on the difference as in (5.5) and (5.6).

As expected, there is an optimum level of m. If $m = 0$, then the game is reduced to the case without "due care"; if m is too large, then the investments are unnecessarily large which increases the social cost. We also see that if the regulator sets the same, but properly chosen m for all players, the social cost could still decrease significantly.



(a) Ratio

(b) Difference

Figure 5.1: The effect of the “due care” scheme

Chapter 6

Conclusion

6.1 Summary

We have studied the equilibrium performance of the network security game. Our model explicitly considered the network topology, the different cost functions of the players, and their relative importance to each other. We showed that in the strategic-form game, the POA can be very large and tends to increase with the network size, and the dependency and imbalance among the players. This indicates severe efficiency problems in selfish investment. We have compared the benefits of improving security technology and improving incentives. In particular, we show that the POA of pure-strategy NE is invariant with the improvement of technology, under the EI model and the BT model. So, improving technology alone may not offset the efficiency loss due to the lack of incentives. As a generalization of our study, we have considered the performance of correlated equilibrium (CE). We have shown that although CE cannot achieve SO in general, it can be better than NE's in some cases. In terms of the worst-case bounds, the POAs of discrete CE are the same as the POAs of pure-strategy NE under the EI model and the BT model.

Not surprisingly, the best equilibrium in the repeated games usually gives much better performance, and it's possible to achieve the social optimum if that does not conflict with individual interests. Implementing the strategies supporting a SPE in a repeated game, however, needs more communication and coordination among the players. To reduce the coordination overhead, we have introduced a new Folk Theorem which only requires local punishments and rewards, but supports the same payoff region as the usual Folk Theorem.

Finally, we have investigated how the worst-case social cost in the strategic-form game could improve with a "due care" scheme where the regulator does not have full knowledge of the network.

6.2 Future work

Usually, the more information the social planner has, the more efficient mechanisms she can deploy, as seen in the "due care" scheme. However, collecting more information means less privacy for the players, which could become a bottleneck to implementing efficient schemes. Similar situations arise for other proposed mechanisms such as cyber-insurance. Therefore, improving network security is complex economic and technological problem that also involves social aspects. A further investigation into these issues, and their impact on the efficiency and implementability of different mechanisms is an interesting direction for future research.

In our study, we have largely assumed that the players in the game are Internet users, such as enterprises and organizations who invest to protect their security. In fact, the role of Internet Service Providers (ISPs) is also important. ISPs know more about the internal structure and operations of their networks than the users, therefore they are arguably in a better position to deploy security measures, such as (i) setting up better filters and intrusion detection systems to block suspicious traffic in their networks, and even block suspicious websites; (ii) setting up more strict security policies (including some security mandates to their users). Unfortunately, the ISPs may not have enough incentives to invest more in security for a number of reasons. First, the investments in security by the ISPs do not necessarily translate to higher revenues. For example, blocking certain traffic or websites may reduce the usability of the Internet for the users, and raising the security standard for the users also causes burdens to some of them. These factors could affect the user subscription to the ISP. Second, the ISPs do not suffer the full consequence of attacks. The increased traffic caused by most attacks or spams can be carried by the ISP's network without too much additional cost, and the damages to end users' computer systems are largely not suffered by the ISPs. Due to the above incentive problems of the ISPs, one would expect that some regulations imposed on the ISPs would benefit the network security.

Note that in principle, our "effective-investment" (EI) model can also include the ISPs as players, whose investments benefit other players. Therefore, it is a quite flexible model to take into account these effects.

Finally, the attackers' incentive is another interesting aspect. The most direct way to discourage the attackers is to catch them. This effort has been under way and intensifying, although it is generally not easy due to the anonymity and lack of traceability in the Internet. On the other hand, if the defenders invest more to make the network more secure, the attackers may be more discouraged to attack, at least when an immediate technology upgrade is not available on the attackers' side.

Chapter 7

Skipped Proofs

7.1 Proof of Proposition 2

Consider player i's set of best responses, $BR_i(\mathbf{x}_{-i})$, to $\mathbf{x}_{-i} \geq \mathbf{0}$. Define $x_{i,max} := [f_i(\mathbf{0}) + \epsilon]/c_i$ where $\epsilon > 0$, then due to convexity of $f_i(\mathbf{x})$ in x_i, we have

$$\begin{aligned} & f_i(x_i = 0, \mathbf{x}_{-i}) - f_i(x_i = x_{i,max}, \mathbf{x}_{-i}) \\ \geq \;& x_{i,max} \cdot (-\frac{\partial f_i(x_{i,max}, \mathbf{x}_{-i})}{\partial x_i}) \\ = \;& \frac{f_i(\mathbf{0}) + \epsilon}{c_i}(-\frac{\partial f_i(x_{i,max}, \mathbf{x}_{-i})}{\partial x_i}) \end{aligned}$$

. Since $f_i(x_i = 0, \mathbf{x}_{-i}) \leq f_i(\mathbf{0})$, and $f_i(x_i = x_{i,max}, \mathbf{x}_{-i}) \geq 0$, it follows that

$$f_i(\mathbf{0}) \geq \frac{f_i(\mathbf{0}) + \epsilon}{c_i}(-\frac{\partial f_i(x_{i,max}, \mathbf{x}_{-i})}{\partial x_i})$$

which means that $\frac{\partial f_i(x_{i,max}, \mathbf{x}_{-i})}{\partial x_i} + c_i > 0$. So, $BR_i(\mathbf{x}_{-i}) \subseteq [0, x_{i,max}]$.

Let $x_{max} = \max_i x_{i,max}$. Consider a modified game where the strategy set of each player is restricted to $[0, x_{max}]$. Since the set is compact and convex, and the cost function is convex, therefore this is a convex game and has some pure-strategy NE [20], denoted as $\bar{\mathbf{x}}$.

Given $\bar{\mathbf{x}}_{-i}$, $\bar{x}_i$ is also a best response in the strategy set $[0, \infty)$, because the best response cannot be larger than x_{max} as shown above. Therefore, $\bar{\mathbf{x}}$ is also a pure-strategy NE in the original game.

7.2 Proof of Proposition 14

Consider the following convex optimization problem parametrized by $\mathbf{t} = (t_1, t_2, \ldots, t_n)$, with optimal value $V(\mathbf{t})$:

$$\begin{aligned} V(\mathbf{t}) = \quad \min\nolimits_{\mathbf{x}\geq \mathbf{0}} \quad & \textstyle\sum_i g_i(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq t_i, \forall i \end{aligned} \tag{7.1}$$

When $\mathbf{t} = \underline{\mathbf{g}}$, it is the same as problem (4.2) that gives the social cost of the best SPE; when $\mathbf{t} = \mathbf{g}^*$, it gives the same solution as the social optimum. According to the theory of convex optimization ([56], page 250), the "value function" $V(\mathbf{t})$ is convex in $\mathbf{t}$. Therefore,

$$V(\underline{\mathbf{g}}) - V(\mathbf{g}^*) \leq \nabla V(\underline{\mathbf{g}})(\underline{\mathbf{g}} - \mathbf{g}^*)$$

Also, $\nabla V(\underline{\mathbf{g}}) = -\lambda$, where λ is the vector of dual variables when the problem with $\mathbf{t} = \underline{\mathbf{g}}$ is solved. So,

$$\begin{aligned} G_E &= V(\underline{\mathbf{g}}) \\ &\leq V(\mathbf{g}^*) + \lambda^T(\mathbf{g}^* - \underline{\mathbf{g}}) \\ &= G^* + \lambda^T(\mathbf{g}^* - \underline{\mathbf{g}}) \\ &\leq G^* + \lambda^T \mathbf{g}^* \end{aligned}$$

Then

$$\gamma = \frac{G_E}{G^*} \leq 1 + \frac{\lambda^T \mathbf{g}^*}{\mathbf{1}^T \mathbf{g}^*} \leq \max_k \{1 + \lambda_k\}$$

which completes the proof.

7.3 Proof of Proposition 15

It is useful to first give a sketch of the proof before going to the details. Roughly, the KKT condition [56] (for the best SPE), as in equation (4.3), is $\sum_k (1 + \lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] = 1 + \lambda_i, \forall i$ (except for some "corner cases" which will be taken care of by Lemma 8). Without considering the corner cases, we have the following by inequality (7.2):

$$\begin{aligned}\gamma &\leq \max_{i,j}\frac{1+\lambda_i}{1+\lambda_j} = \max_{i,j}\frac{\sum_k(1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}]}{\sum_k(1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}]}\\ &\leq \max_{i,j,k}\{\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}/\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}\}\end{aligned}$$

which is Proposition 18. Then by plugging in $f_k(\cdot)$ of the EI model, Proposition 15 immediately follows.

Now we begin the detailed proof.

As assumed in section 4.1, $\mathbf{g}(\mathbf{x}) < \underline{g}$ is feasible.

Lemma 7. *If $\mathbf{g}(\mathbf{x}) < \underline{\mathbf{g}}$ is feasible, then at the optimal solution of problem (4.2), at least one dual variable is 0. That is, $\exists i_0$ such that $\lambda_{i_0} = 0$.*

Proof. Suppose $\lambda_i > 0, \forall i$. Then all constraints in (4.2) are active. As a result, $G_E = \sum_k \underline{g_k}$.

Since $\exists\mathbf{x}$ such that $\mathbf{g}(\mathbf{x}) < \underline{\mathbf{g}}$, then for this $\mathbf{x}$, $\sum_k g_k(\mathbf{x}) < \sum_k \underline{g_k}$. $\mathbf{x}$ is a feasible point for (4.2), so $G_E \leq \sum_k g_k(\mathbf{x}) < \sum_k \underline{g_k}$, which contradicts $G_E = \sum_k \underline{g_k}$. □

From Proposition 14, we need to bound $\max_k\{1+\lambda_k\}$. Since $1+\lambda_i \geq 1, \forall i$, and $1+\lambda_{i_0} = 1$ (by Lemma 7), it is easy to see that

$$\gamma \leq \max_k\{1+\lambda_k\} = \max_{i,j}\frac{1+\lambda_i}{1+\lambda_j} \tag{7.2}$$

Before moving to Proposition 18, we need another observation:

Lemma 8. *If for some i, $\sum_k(1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] < 1+\lambda_i$, then $\lambda_i = 0$.*

Proof. From (4.3), it follows that $x_{E,i} = 0$. Since $\sum_k(1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] < 1+\lambda_i$, and every term on the left is non-negative, we have

$$(1+\lambda_i)[-\frac{\partial f_i(\mathbf{x}_E)}{\partial x_i}] < 1+\lambda_i$$

That is, $\frac{\partial f_i(\mathbf{x}_E)}{\partial x_i} + 1 = \frac{\partial g_i(\mathbf{x}_E)}{\partial x_i} > 0$. Since $f_i(\mathbf{x})$ is convex in x_i, and $x_{E,i} = 0$, then

$$g_i(\underline{x_i}, \mathbf{x}_{E,-i}) \geq g_i(x_{E,i}, \mathbf{x}_{E,-i}) + \frac{\partial g_i(\mathbf{x}_E)}{\partial x_i}(\underline{x_i} - 0) > g_i(\mathbf{x}_E)$$

where we have used the fact that $\underline{x_i} > 0$.

Note that $g_i(\underline{x_i}, \mathbf{x}_{E,-i}) \leq g_i(\underline{x_i}, \mathbf{0}_{-i}) = \underline{g_i}$. Therefore,

$$g_i(\mathbf{x}_E) < \underline{g_i}$$

So $\lambda_i = 0$. □

Proposition 18. *With the general cost function $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$, γ is upper-bounded by*

$$\gamma \leq \min\{\max_{i,j,k}\{\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i} / \frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}\}, Q\}$$

where Q is the POA derived before for Nash Equilibria in the strategic-form game (i.e., $\rho \leq Q$), and $\mathbf{x}_E$ achieves the optimal social cost in the set of SPE.

Proof. First of all, since any NE is Pareto-dominated by $\underline{\mathbf{g}}$, the best SPE is at least as good as NE. So $\gamma \leq Q$.

Consider $\pi_{i,j} := \frac{1+\lambda_i}{1+\lambda_j}$. (a) If $\lambda_i = 0$, then $\pi_{i,j} \leq 1$. (b) If $\lambda_i, \lambda_j > 0$, then according to Lemma 8, we have $\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] = 1 + \lambda_i$ and $\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}] = 1 + \lambda_j$. Therefore

$$\pi_{i,j} = \frac{\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}]}{\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}]} \leq \max_k\{\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i} / \frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}\}$$

(c) If $\lambda_i > 0$ but $\lambda_j = 0$, then from Lemma 8, $\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}] = 1 + \lambda_i$ and $\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}] \leq 1 + \lambda_j$. Therefore,

$$\pi_{i,j} \leq \frac{\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i}]}{\sum_k (1+\lambda_k)[-\frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}]} \leq \max_k\{\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i} / \frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}\}$$

Considering the cases (a), (b) and (c), and from equation (7.2), we have

$$\gamma \leq \max_{i,j} \pi_{i,j} \leq \max_{i,j,k}\{\frac{\partial f_k(\mathbf{x}_E)}{\partial x_i} / \frac{\partial f_k(\mathbf{x}_E)}{\partial x_j}\}$$

which completes the proof. □

Proposition 18 applies to any game with the cost function $g_i(\mathbf{x}) = f_i(\mathbf{x}) + x_i$, where $f_i(\mathbf{x})$ is non-negative, decreasing in each x_i, and satisfies the assumption (1)-(3) at the beginning of section 4.1. This includes the EI model and the BT model introduced before. It is not easy to find an explicit form of the upper bound on γ in Proposition 18 for the BT model. However, for the EI model, we have the simple expression shown in Proposition 15:

$$\gamma \leq \min\{\max_{i,j,k} \frac{\beta_{ik}}{\beta_{jk}}, Q\}$$

where $Q = \max_k\{1 + \sum_{i:i\neq k} \beta_{ki}\}$.

Proof. The part $\gamma \leq Q$ is straightforward: since the set of SPE includes all NE's, the best SPE must be better than the worst NE. Also, since $\frac{\partial f_k(\mathbf{x}_E)}{x_i} = \beta_{ik} V_k'(\sum_m \beta_{mk} x_{E,m})$, and $\frac{\partial f_k(\mathbf{x}_E)}{x_j} = \beta_{jk} V_k'(\sum_m \beta_{mk} x_{E,m})$, using Proposition 18, we have $\gamma \leq \max_{i,j,k} \frac{\beta_{ik}}{\beta_{jk}}$. $\square$

7.4 Proof of Lemma 3

Proof. The performance ratio between the discrete CE $\mu(\mathbf{x})$ and the social optimal is

$$\rho(\mu) \quad := \quad \frac{G(\mu)}{G^*} = \frac{E[\sum_i (f_i(\mathbf{x}) + x_i)]}{\sum_i [f_i(\mathbf{x}^*) + x_i^*]}$$

where the expectation (and all other expectations below) is taken over the distribution μ.

Since $f_i(\cdot)$ is convex for all i. Then for any $\mathbf{x}$, $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*) + (\mathbf{x} - \mathbf{x}^*)^T \nabla f_i(\mathbf{x})$. So

$$\begin{aligned} & \rho(\mu) \\ \leq \quad & \frac{E[(\mathbf{x} - \mathbf{x}^*)^T \sum_i \nabla f_i(\mathbf{x}) + \mathbf{1}^T \mathbf{x}] + \sum_i f_i(\mathbf{x}^*)}{\sum_i f_i(\mathbf{x}^*) + \mathbf{1}^T \mathbf{x}^*} \\ = \quad & \frac{E\{-\mathbf{x}^{*T} \sum_i \nabla f_i(\mathbf{x}) + \mathbf{x}^T[\mathbf{1} + \sum_i \nabla f_i(\mathbf{x})]\} + \sum_i f_i(\mathbf{x}^*)}{\sum_i f_i(\mathbf{x}^*) + \mathbf{1}^T \mathbf{x}^*} \end{aligned}$$

Note that

$$\mathbf{x}^T[\mathbf{1} + \sum_i \nabla f_i(\mathbf{x})] = \sum_i x_i [1 + \sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i}].$$

For every player i, for each x_i with positive probability, there are two possibilities: (a) If $x_i = 0$, then $x_i[1 + \sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i}] = 0, \forall \mathbf{x}$; (b) If $x_i > 0$, then by (3.25), $E(\frac{\partial f_i(\mathbf{x})}{\partial x_i} | x_i) = -1$.

Since $\frac{\partial f_k(\mathbf{x})}{\partial x_i} \leq 0$ for all k, then $E(\sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i}|x_i) \leq -1$. Therefore for both (a) and (b), we have $E[x_i(1+\sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i})|x_i] = x_i \cdot E[1+\sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i}|x_i] \leq 0$. So,

$$\begin{aligned} & E\{\sum_i [x_i(1+\sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i})]\} \\ = \quad & \sum_i E\{E[x_i(1+\sum_k \frac{\partial f_k(\mathbf{x})}{\partial x_i})|x_i]\} \leq 0. \end{aligned}$$

As a result,

$$\rho(\mu) \leq \frac{-E[\mathbf{x}^{*T}\sum_i \nabla f_i(\mathbf{x})] + \sum_i f_i(\mathbf{x}^*)}{\sum_i f_i(\mathbf{x}^*) + \mathbf{1}^T\mathbf{x}^*}. \tag{7.3}$$

Consider two cases:

(i) If $x_i^* = 0$ for all i, then the RHS is 1, so $\rho(\mu) \leq 1$. Since $\rho(\mu)$ cannot be smaller than 1, we have $\rho(\mu) = 1$.

(ii) If not all $x_i^* = 0$, then $\mathbf{1}^T\mathbf{x}^* > 0$. Note that the RHS of (7.3) is not less than 1, by the definition of $\rho(\mu)$. So, if we subtract $\sum_i f_i(\mathbf{x}^*)$ (non-negative) from both the numerator and the denominator, the resulting ratio upper-bounds the RHS. That is,

$$\rho(\mu) \leq \frac{-E[\mathbf{x}^{*T}\sum_i \nabla f_i(\mathbf{x})]}{\mathbf{1}^T\mathbf{x}^*} \leq \max_k\{E(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_k})\}$$

where $\sum_i \frac{\partial f_i(\bar{\mathbf{x}})}{\partial x_k}$ is the k'th element of the vector $\sum_i \nabla f_i(\bar{\mathbf{x}})$.

Combining cases (i) and (ii), we have

$$\rho(\mu) \leq \max\{1, \max_k E(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_k})\}.$$

Then, ρ_{CE} is upper-bounded by $\max_{\mu \in \mathcal{C}_D} \rho(\mu)$. □

7.5 Proof of Proposition 13

Proof. Since μ is a discrete CE, by (3.25), for any x_i with positive probability, $E(-\frac{\partial f_i(\mathbf{x})}{\partial x_i}|x_i) \leq 1$. Therefore $E(-\frac{\partial f_i(\mathbf{x})}{\partial x_i}) \leq 1$.

In the EI model, we have

$$-\frac{\partial f_i(\mathbf{x})}{\partial x_k} = \beta_{ki}[-\frac{\partial f_i(\mathbf{x})}{\partial x_i}].$$

Therefore

$$E(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_k}) = E(-\sum_i \beta_{ki} \frac{\partial f_i(\mathbf{x})}{\partial x_i}) \le \sum_i \beta_{ki}.$$

So, $\rho_{CE} \le \max_k\{1 + \sum_{i:i\neq k} \beta_{ki}\}$.

In the BT model, similar to the proof in Proposition 4, it's not difficult to see that the following holds for any $\mathbf{x}$:

$$[-\sum_{i:i\neq j} \frac{\partial f_i(\mathbf{x})}{\partial x_j}]/[-\frac{\partial f_j(\mathbf{x})}{\partial x_j}] \quad \le \quad \max_{i:i\neq j} \frac{v_i r_{ji}}{v_j r_{ij}}.$$

Then,

$$-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_j} \le (1 + \max_{i:i\neq j} \frac{v_i r_{ji}}{v_j r_{ij}})[-\frac{\partial f_j(\mathbf{x})}{\partial x_j}].$$

If μ is a discrete CE, then $E(-\frac{\partial f_j(\mathbf{x})}{\partial x_j}) \le 1, \forall j$. Therefore $E(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_j}) \le (1 + \max_{i:i\neq j} \frac{v_i r_{ji}}{v_j r_{ij}})$. So,

$$\rho_{CE} \le \max_j E(-\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_j}) \le (1 + \max_{(i,j):i\neq j} \frac{v_i r_{ji}}{v_j r_{ij}}).$$

□

Part II

CSMA Scheduling and Congestion Control

Chapter 8

Introduction

In multi-hop wireless networks, it is an important problem to efficiently utilize the network resources and provide fairness to competing data flows. These objectives require the cooperation of different network layers (See Fig. 8.1). Assuming that the routing (in the network layer) is fixed, the *transport layer* needs to inject the right amount of traffic into the network based on the congestion level and the *MAC layer* (i.e., Medium Access Control layer) needs to serve the traffic efficiently to achieve high throughput. Through a utility maximization framework [24], this problem can be naturally decomposed into congestion control at the transport layer and scheduling at the MAC layer. The utility maximization problem is an instance of (1.3):

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i=1}^{m} U_i(x_i) \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{8.1}$$

where m is the number of data flows, x_m is the rate of flow m, $\mathcal{X}$ is the "capacity region" of the network.

It turns out that MAC-layer scheduling is the bottleneck of the problem [24]. In particular, it is not easy to achieve the maximal throughput through distributed scheduling, which in turn prevents full utilization of the wireless network. Scheduling is challenging since the conflicting relationships between different links can be complicated.

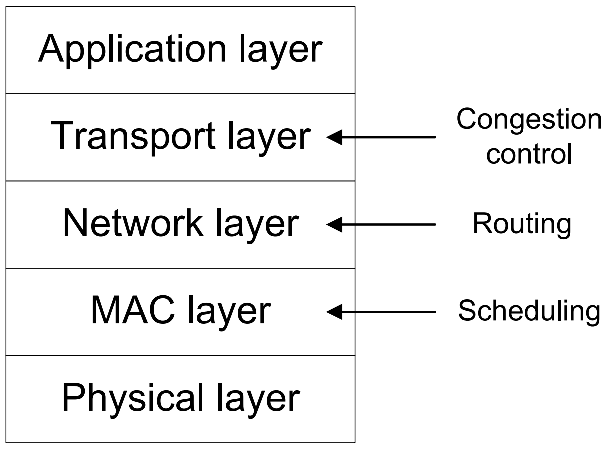


Figure 8.1: Network layers

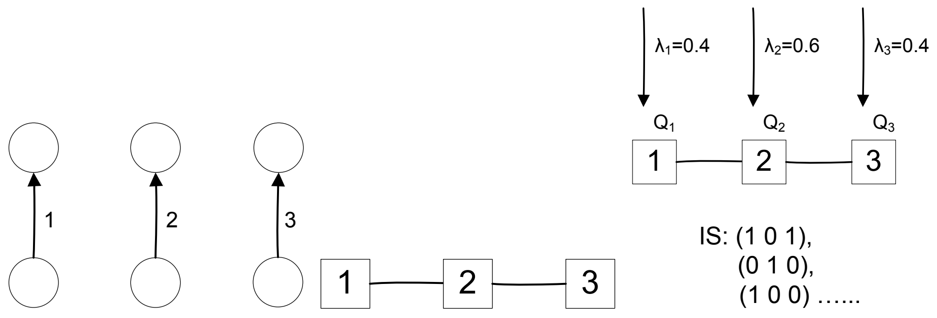
8.1 Interference model and the scheduling problem

First we describe the general interference model we will consider. Assume there are K *links* in the network, where each *link* is an (ordered) transmitter-receiver pair. The network is associated with a conflict graph (or "CG") $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of vertices (each of them represents a link) and $\mathcal{E}$ is the set of edges. Two links cannot transmit at the same time (i.e., "conflict") iff there is an edge between them.

For example, consider the network in Fig 8.2 (a). Assume that link 1 and 2 cannot transmit at the same time since they are too close, and assume the same for link 2 and 3. Then, the network's conflict graph (CG) is shown in Fig 8.2 (b), where each square represents a link and the edge between a pair of links indicates interference.

An independent set (IS) in G is a set of links that can transmit at the same time without any interference. Let $\mathcal{X}$ be the set of all IS's of G (not confined to "maximal independent sets"), and let $N = |\mathcal{X}|$ be the number of IS's. Denote the i'th IS as $x^i \in \{0,1\}^K$, a 0-1 vector that indicates which links are transmitting in this IS. The k'th element of x^i, $x^i_k = 1$ if link k is transmitting, and $x^i_k = 0$ otherwise. We also refer to x^i as a "transmission state", and x^i_k as the "transmission state of link k".

For example, in the CG in Fig 8.2 (b), the following states constitute independent sets: (i) No link transmits; (ii) Only one link transmits; (iii) Two links, link 1 and 3, transmit at



(a) Network (b) Conflict Graph (CG) (c) Arrival rates

Figure 8.2: Example

the same time. We have listed a few IS's in Fig 8.2 (c), denoted by the 0-1 vectors defined above.

Assume i.i.d. traffic arrival at each link k with arrival rate λ_k. $\lambda_k \leq 1$ is also *normalized* with respect to the link capacity 1, and thus can be viewed as the fraction of time when link k needs to be active to serve the arrival traffic. And denote the vector of arrival rates as $\lambda \in R_+^K$. Further assume that $\lambda_k > 0, \forall k$ without loss of generality, since the link(s) with zero arrival rate can be removed from the problem.

Definition 1. *(i) λ is said to be* feasible *if and only if $\lambda = \sum_{i=1}^N \bar{p}_i \cdot x^i$ for some probability distribution $\bar{\mathbf{p}} \in \mathcal{R}_+^N$ satisfying $\bar{p}_i \geq 0$ and $\sum_{i=1}^N \bar{p}_i = 1$. That is, λ is a convex combination of the IS's, such that it is possible to serve the arriving traffic with some transmission schedule. Denote the set of feasible λ by $\bar{\mathcal{C}}$.*

(ii) λ is said to be *strictly feasible* iff it can be written as $\lambda = \sum_{i=1}^N \bar{p}_i \cdot x^i$ where $\bar{p}_i > 0$ and $\sum_{i=1}^N \bar{p}_i = 1$. Denote the set of strictly feasible λ by $\mathcal{C}$.

For example, the arrival rates $\lambda = (0.4, 0.6, 0.4)$ is feasible since $\lambda = 0.4 * (1, 0, 1) + 0.6 * (0, 1, 0)$. However, it is not strictly feasible because the IS $(0, 0, 0)$ must have a probability of 0. On the other hand, $\lambda = (0.4, 0.5, 0.4)$ is strictly feasible.

We show the following relationship in section 9.7.1.

Proposition 19. *The set $\mathcal{C}$ is exactly the interior of $\bar{\mathcal{C}}$.*

Remark*: The interior of $\bar{\mathcal{C}}$ is defined as int $\bar{\mathcal{C}} := \{\lambda \in \bar{\mathcal{C}} | \mathcal{B}(\lambda, d) \subseteq \bar{\mathcal{C}}$ for some $d > 0\}$, where $\mathcal{B}(\lambda, d) = \{\lambda' | \, ||\lambda' - \lambda||_2 \leq d\}$.*

Now we define what is a scheduling algorithm and when it is called "throughput-optimum".

Definition 2. *A **scheduling algorithm** decides which links should transmit at any time instance t, given the history of the system (possibly including the history of queue lengths, arrival processes, etc) up to time t.*

*A scheduling algorithm is **throughput optimum** if it can support any strictly feasible arrival rates $\lambda \in \mathcal{C}$ (in other words, it can stabilize the queues whenever possible). Equivalently, we also say that such an algorithm achieves the **maximal throughput**.*

We say that a scheduling algorithm is **distributed** if each link only uses information within its one-hop neighborhood. We are primarily interested in designing a distributed scheduling algorithm that is throughput optimum.

8.2 Related works

8.2.1 Scheduling algorithms

There have been many scheduling algorithms proposed in the literature. Most of them divide the time into equal-length "slots". In each slot $t, t = 0, 1, 2, \ldots$, the algorithm chooses an IS $\mathbf{x}^*(t) \in \mathcal{X}$ to be active (that is, each link in the IS transmits a unit-length packet). The scheduling decision is usually based on $\mathbf{Q}(t) \in \mathcal{R}_+^K$, the queue lengths of different links in that slot (with higher priority usually given to longer queues). Then the queue lengths in the next slot are

$$\mathbf{Q}(t+1) = [\mathbf{Q}(t) - \mathbf{x}^*(t)]_+ + \mathbf{a}(t)$$

where $\mathbf{a}(t) \in \mathcal{R}_+^K$ is the vector of the number of arrived packets in slot t. It is assumed that $E(\mathbf{a}(t)) = \lambda$.

Maximal-weight scheduling

A classical throughput-optimum algorithm is maximal-weight scheduling (MWS) [47]. (This algorithm has also been applied to achieve 100% throughput in input-queued switches [48].) With MWS, in each slot, an IS with the maximal "weight" is scheduled, where the "weight" of an IS is the summation of the queue lengths of the active links in the IS. That is, the scheduled IS in slot t is

$$\mathbf{x}^*(t) \in \arg\max_{\mathbf{x}\in\mathcal{X}} w(\mathbf{x}, t)$$

where $w(\mathbf{x}, t)$ is the weight of the IS $\mathbf{x}$ in slot t:

$$w(\mathbf{x}^i, t) := \sum_k [x_k^i Q_k(t)].$$

For example, assume that $\mathbf{Q}(t) = (5, 4, 3)$ in the 3-link network in Fig 8.2 (b), then $\mathbf{x}^*(t) = (1, 0, 1)$.

However, implementing MWS in general G is quite difficult for two reasons. (i) MWS is inherently a centralized algorithm and is not amenable to distributed implementation; (ii) finding a maximal-weighted IS (in each slot) is NP-complete in general and is hard even for centralized algorithms. Therefore, MWS is not suitable for distributed wireless networks.

Low-complexity but sub-optimal algorithms

Due to the above disadvantages of MWS, a number of low-complexity, but sub-optimal scheduling algorithms have been proposed. The Maximal Scheduling algorithm (MS) was proposed in [33] and was also studied in the context of 802.11-like protocol [34]. In each slot, MS chooses links with non-empty queues until no further link can be chosen without interference. For example, assume that $\mathbf{Q}(t) = (5, 4, 0)$ in Fig 8.2 (b). Then MS either activates link 1 or link 2 (but not both) in slot t.

In LQF [35; 36; 37; 38], $\mathbf{x}^*(t)$ is constructed by iteratively choosing the longest queue. (Therefore LQF can be viewed as a greedy algorithm.) For example, assume that $\mathbf{Q}(t) = (5, 4, 3)$ in Fig 8.2 (b). To determine $\mathbf{x}^*(t)$, LQF does the following. First, since link 1 has the longest queue, let $x_1^*(t) = 1$. After choosing link 1, link 2 cannot be chosen, therefore $x_2^*(t) = 0$. In the remaining link(s), link 3 has the longest queue, so let $x_3^*(t) = 1$. Repeat the process until all links are decided.

Although the above algorithms have low computational complexity, they can only achieve a fraction of the capacity region $\mathcal{C}$ in general. The size of the fraction depends on the network topology and interference relationships. Since LQF takes into account the queue lengths, it can usually achieve higher throughput than MS and also has good delay performance. In fact, it has been shown that LQF is throughput optimum if the network topology satisfies a "local pooling" condition [35], or if the network is small [38]. In general topologies, however, LQF is not throughput optimum, and the fraction of $\mathcal{C}$ achievable can be computed as in [36].

Throughput-optimum algorithms for restrictive interference models

A few recent works proposed throughput-optimal algorithms for certain interference models. For example, Eryilmaz et al. [28] proposed a polynomial-complexity algorithm for the "two-hop interference model"[1]. Modiano et al. [29] introduced a gossip algorithm for the "node-exclusive model"[2]. The extensions to more general interference models, as discussed in [28] and [29], usually involves extra challenges. Sanghavi et al. [30] introduced an algorithm that can approach the throughput capacity (with increasing overhead) for the node-exclusive model.

Random Access algorithms

Recently, a number of researchers realized that random access algorithms, despite their simplicity, can achieve high throughput in wireless networks. Random access algorithms differ significantly from the synchronous time-slotted model adopted in many existing scheduling algorithms described above. Of particular interest is the CSMA/CA algorithm (Carrier

[1]In this model, a transmission over a link from node m to node n is successful iff none of the one-hop neighbors of m and n is in any conversation at the time.

[2]In this model, a transmission over a link from node m to node n is successful iff neither m nor n is in another conversation at the time.

Sense Multiple Access / Collision Avoidance) widely deployed in the current IEEE 802.11 wireless networks. In CSMA/CA (or "CSMA" for simplicity), each link keeps sensing the medium when it is not transmitting. If the link senses that transmission signal (or power) of some conflicting link(s), the link does not start a transmission to avoid a collision. Otherwise, the link starts a transmission after a randomly chosen waiting time (or "backoff time"). The original purpose of the random backoff time is to prevent several links from starting transmitting at the same time which leads to collisions. As a result, the algorithm is asynchronous and distributed in nature, where each link decides its action based on its own observations of its neighbors. With these advantages, a question is whether this type of algorithm can achieve the maximum throughput.

In [53], Durvy and Thiran showed that asynchronous CSMA can achieve a high level of spatial reuse, via the study of an idealized CSMA model without collisions. In [40], Marbach et al. considered a model of CSMA with collisions. It was shown that under a restrictive "node-exclusive" interference model, CSMA can be made *asymptotically* throughput-optimal in the limiting regime of large networks with a small sensing delay. (Note that when the sensing delay goes to 0, collisions asymptotically disappear.) In [41], Proutiere et al. developed asynchronous random-access-based algorithms whose throughput performance, although not optimum, is no less than some maximal scheduling algorithms, e.g. Maximum Size scheduling algorithms.

However, none of these works have established the throughput optimality of CSMA under a general interference model, nor have they designed specific algorithms to achieve the optimality.

8.2.2 Joint scheduling and congestion control

In the scheduling problem, it is usually assumed that the (random) arrival processes to the links are given[3], and the purpose of the scheduling algorithm is to support the arrivals (such that the queues are stable). In the protocol stack of wireless networks, the scheduling algorithm is located in the medium access control (MAC) layer. The transport layer, on

[3]But the average arriving rates are general unknown.

the other hand, controls the arrival processes. These two layers need to work together to achieve high throughput as well as certain fairness (in terms of maximum total utility) among different data flows which traverse the network. This is the joint scheduling and congestion control problem.

Interestingly, solving the utility maximization problem ([24]) naturally leads to a simple congestion control algorithm at the transport layer and the maximal-weight scheduling (MWS) at the MAC layer. Unfortunately, as mentioned in section 8.2.1, implementing MWS is not practical in distributed networks. This motivated the study of combining imperfect scheduling with congestion control: Reference [39] investigated the impact of imperfect scheduling on network utility maximization. Related to this area, there is research on utility maximization given a certain MAC layer protocol, for example [31] and [32] which considered the slotted-ALOHA random access protocol at the MAC layer. Due to the inherent inefficiency of slotted-ALOHA, however, these proposals cannot achieve the maximum utility that is achievable with perfect scheduling.

8.3 Overview of results

8.3.1 Throughput-Optimum CSMA Scheduling

Our first contribution is to introduce a *distributed* adaptive CSMA (Carrier Sensing Multiple Access) algorithm for a general interference model. It is inspired by CSMA but may be applied to more general resource sharing problems (i.e., not limited to wireless networks). We show the algorithm (and its variants) can achieve the maximal throughput, in an idealized-CSMA model without packet collisions and also in a CSMA model with collisions. The algorithm may not be directly comparable to the throughput-optimal algorithms mentioned above since it utilizes the carrier-sensing capability. But it does have a few distinct features:

- Each node only uses its local information (e.g., its backlog). No explicit control messages are required among the nodes.

- It is based on CSMA random access, which is similar to the IEEE 802.11 protocol and is easy to implement.

- Time is not divided into synchronous slots. Thus no synchronization of transmissions is needed.

In [42], Rajagopalan and Shah independently proposed a throughput-optimal algorithm similar to ours in the context of optical networks. However, there are some notable differences. First, in Theorem 2, we show that any $\lambda \in \mathcal{C}$ can be supported by a fixed set of parameters in CSMA, and our algorithms are designed to find or approximate these parameters. This observation was not made in [42]. Instead, the stationary distribution of the network dynamics in [42] (similar to CSMA) are used to approximate the maximal-weight schedule. Also, utility maximization (discussed below) was not considered in [42].

Next we give an overview on how the adaptive CSMA scheduling works. We first discuss the case with the idealized-CSMA without packet collisions (this case will be covered in Chapter 9 which is based on [26; 67]). The case with collisions will be explained later.

Consider a wireless network where some links interfere. Packets arrive at the transmitters of the links with certain rates. Consider an "idealized-CSMA" protocol [51; 52] that works as follows. The different transmitters choose independent exponentially-distributed backoff times. A transmitter decrements its backoff timer when it senses the channel idle and starts transmitting when its timer runs out. The packet transmission times are also exponentially distributed. (The process defines a "CSMA Markov chain".) The assumption in [51; 52] is that a transmitter hears any transmitter of a link that would interfere with it. That is, there are no hidden nodes. Moreover, the transmitters hear a conflicting transmission instantaneously. Accordingly, there are no collisions in the idealized-CSMA. In practice, other protocols with RTS/CTS (i.e., the control packets named "Request-to-Send"/"Clear-to-Send") can be used to address the hidden node problems [51; 55].

The "adaptive CSMA" scheduling algorithm works in a very intuitive way: Each link adjusts its transmission aggressiveness ("TA") based on its backlog. **If its backlog increases, then the link transmits more aggressively, and vice versa.** For simplicity, we temporarily make an idealized time-scale-separation approximation, that is, as the links change their TA, the CSMA Markov chain instantaneously reaches its stationary distribution. Under this approximation, we can show that the following simple algorithm is throughput-optimal.

- The transmitter of link k sets its mean backoff time to be $\exp\{-\alpha \cdot Q_k\}$ where Q_k is the backlog of link k and $\alpha > 0$ is any constant.

Since it takes some time to reach the stationary distribution, the approximation does not hold in practice. Therefore, the actual algorithm is somewhat different from this. However, the main idea remains the same. (Chapter 9 will give the details.)

In the case with collisions, to achieve throughput-optimality, we need to limit the impact of collisions. This leads us to the following design choices.

- Each link fixes the mean backoff time, but adjusts the mean transmission time according to its backlog. If the backlog increases, it increases the mean transmission time, and vice versa. (This is because if we adjust the backoff time, then the collisions would be excessive when the backoff time is small.)
- Also, we adopt the RTS/CTS mode of IEEE 802.11 standard. That is, before a link transmits a data packet, it first transmits a short control packet (RTS, or "Request-to-Send" packet). If the RTS is successful (i.e., it is acknowledged by a CTS, or a "Clear-to-Send" packet, from the intended receiver), then the data packet is transmitted. Otherwise, the link will wait for a random time and transmits the RTS again. The benefit of RTS/CTS mode is that, in the absence of hidden nodes, collisions only happen among the short control packets, but not the longer data packets.

With the above designs, the impact of collisions is limited: the cost due to collisions does not increase with the lengths of data packets. Thus, with long enough data packets, throughput-optimality can be approached, if we can establish the throughput-optimality in the idealized-CSMA model.

We will formally prove these results in Chapter 10, which is based on [72; 73].

In both the cases without or with collisions, the main thread of our development includes the following three steps.

- Analyzing the stationary distribution of the Markov chain defined by the CSMA protocol with given parameters (TA). The stationary distributions in both cases (without or with collisions) have a product-form, due to their time-reversible or quasi-reversible structure.
- Then showing that there exist a proper choice of the parameters such that the average service rates induced by the product-form distribution can support any strictly feasible arrival rates on all links. (But our algorithms do not require apriori knowlege of the arrival rates.) This has an interesting connection with Markov Random Fields.
- Based on these insights, designing adaptive distributed algorithms to stabilize the queues. The algorithms essentially solve a convex optimization problem with noisy gradients. (Therefore we use tools in stochastic approximation.)

8.3.2 Joint CSMA scheduling and congestion control

Our second contribution is to combine the proposed scheduling algorithm with end-to-end congestion control using a novel "maximal entropy" technique, to achieve fairness among competing flows as well as maximal throughput (sections 9.2, 9.3).

The resulting algorithm is very simple. In addition to the CSMA scheduling algorithm at the MAC layer, the source of each data flow performs the following congestion control function: **it decreases its input rate if the queue at the source builds up, and vice versa.**

We further show that the proposed CSMA scheduling not only can be combined with congestion control at the transport layer, it is in fact a modular MAC-layer algorithm that can work with various protocols in other layers. In section 9.7.4, we derive and present the joint algorithms after combining it with optimal routing, anycast and multicast at the network layer.

Chapter 9

Distributed CSMA Scheduling and Congestion Control

9.1 Adaptive CSMA scheduling for maximal throughput

9.1.1 An idealized CSMA protocol and the average throughput

We use an idealized model of CSMA as in [50; 51; 52]. This model makes two simplifying assumptions. First, it assumes that if two links conflict – because their simultaneous transmissions would result in incorrectly received packets – then each of the two links hears when the other one transmits. Second, the model assumes that this sensing is instantaneous, so that collisions do not occur. The first assumption implies that there are no hidden nodes (HN). This is possible if the range of carrier-sensing is large enough [55].[1]) The second assumption is violated in actual systems because of the finite speed of light and of the time needed to detect a received power.

[1]A related problem that affects the performance of wireless networks is the exposed-node (EN) problem. EN occurs when two links could transmit together without interference, but they can sense the transmission of each other. As a result, their simultaneous transmissions are unnecessarily forbidden by CSMA. Reference [55] proposed a protocol to address HN and EN problems in a systematic way. We assume here that the HN and EN are negligible with the use of such a protocol. Note that however, although EN problem may *reduce* the capacity region, it does not affect the applicability of our model, since we can define an edge between two links in the CG as long as they can sense the transmission of each other, even if this results in EN.

Despite these assumptions, we will use the idealized-CSMA model in this chapter for the following reasons:

(1) We find that the idealized-CSMA model is an easier starting point before analyzing the case with collisions. In fact, it has captured the essence of CSMA. Indeed, in Chapter 10, we will present another model that explicitly considers collisions in wireless network. That model is a natural generalization of idealized-CSMA. Also, our distributed scheduling and congestion control algorithms based on idealized-CSMA can be naturally extended to that model.

(2) Collisions and the HN problem only occur in wireless networks, but not in a more general class of networks such as stochastic processing networks (SPN) [25]. Designing scheduling algorithms based on idealized-CSMA is certainly relevant and interesting for those networks.

SPN is a general model for manufacturing, communication, or service systems. Consider the following "task processing" problem in SPN for an example. There are K different types of tasks and a finite set of resources $\mathcal{B}$. To perform a type-k task, one needs a subset $\mathcal{B}_k \subseteq \mathcal{B}$ of resources and these resources are then monopolized by the task while it is being performed. Note that two tasks can be performed simultaneously iff they use disjoint subsets of resources. Clearly this can be accommodated in our model in section 8.1 by associating each type of tasks to a "link".

The idealized-CSMA works as follows. If the transmitter of link k senses the transmission of any conflicting link (i.e., any link m such that $(k, m) \in \mathcal{E}$), then it keeps silent. If none of its conflicting links is transmitting, then the transmitter of link k waits (or backs-off) for a random period of time that is exponentially distributed with mean $1/R_k$ and then starts its transmission[2]. If some conflicting link starts transmitting during the backoff, then link k suspends its backoff and resumes it after the conflicting transmission is over. The transmission time of link k is exponentially distributed with mean 1. (The assumption on exponential distribution can be relaxed [52].) Assuming that the sensing time is

[2]If more than one backlogged links share the same transmitter, the transmitter maintains independent backoff timers for these links.

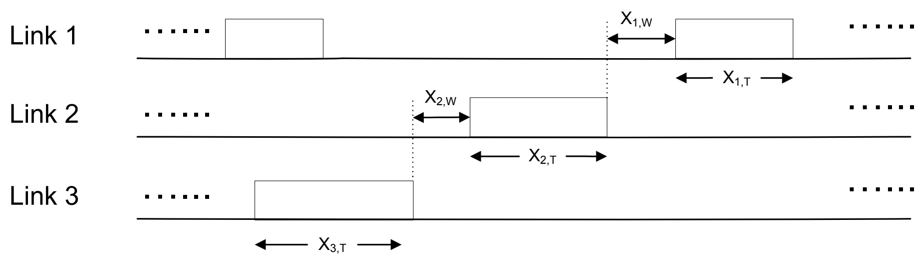


Figure 9.1: Timeline of the idealized CSMA

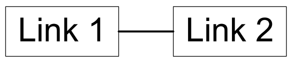
negligible, given the continuous distribution of the backoff times, the probability for two conflicting links to start transmission at the same time is zero. So collisions do not occur in idealized-CSMA.

For the 3-link network in Fig. 8.2 (b), the timeline of the above idealized CSMA protocol is shown in Fig. 9.1, where $X_{k,W}$ and $X_{k,T}$ are the random waiting time and random transmission time of link k. Note that every time when link k transmits a packet, $X_{k,W}$ and $X_{k,T}$ are newly generated, independent of the past.

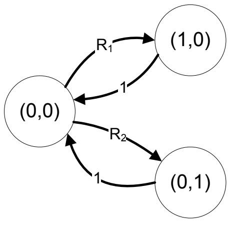
It is not difficult to see that the transitions of the transmission states form a Continuous Time Markov Chain, which is called the *CSMA Markov Chain*. The state space of the Markov chain is $\mathcal{X}$. Denote link k's neighboring set by $\mathcal{N}(k) := \{m : (k,m) \in \mathcal{E}\}$. If in state $x^i \in \mathcal{X}$, link k is not active ($x^i_k = 0$) and all of its conflicting links are not active (i.e., $x^i_m = 0, \forall m \in \mathcal{N}(k)$), then state x^i transits to state $x^i + \mathbf{e}_k$ with rate R_k, where $\mathbf{e}_k$ is the K-dimension vector whose k'th element is 1 and all other elements are 0's. Similarly, state $x^i + \mathbf{e}_k$ transits to state x^i with rate 1. However, if in state x^i, any link in its neighboring set $\mathcal{N}(k)$ is active, then state $x^i + \mathbf{e}_k$ does not exist (i.e., $x^i + \mathbf{e}_k \notin \mathcal{X}$).

Fig 9.2 gives an example network whose CG is shown in (a). There are two links, with an edge between them, which means that they cannot transmit together. Fig 9.2 (b) shows the corresponding CSMA Markov Chain. State (0,0) means that no link is transmitting, state (1,0) means that only link 1 is transmitting, and (0,1) means that only link 2 is transmitting. The state (1,1) is not feasible.

Let $r_k = \log(R_k)$. We call r_k the "transmission aggressiveness" (TA) of link k. For a given positive vector $\mathbf{r} = \{r_k, k = 1, \ldots, K\}$, the CSMA Markov chain is irreducible.



(a) Conflict graph



(b) CSMA Markov Chain

Figure 9.2: Example: Conflict graph and corresponding Markov Chain.

Designate the stationary distribution of its feasible states x^i by $p(x^i; \mathbf{r})$. We have the following result.

Lemma 9. *([50; 51; 52]) The stationary distribution of the CSMA Markov chain has the following product-form:*

$$p(x^i; \mathbf{r}) = \frac{\exp(\sum_{k=1}^{K} x_k^i r_k)}{C(\mathbf{r})} \tag{9.1}$$

where

$$C(\mathbf{r}) = \sum_j \exp(\sum_{k=1}^{K} x_k^j r_k) \ . \tag{9.2}$$

Note that the summation $\sum_j$ is over all feasible states x^j.

Proof. We verify that the distribution (9.1)-(9.2) satisfies the detailed balance equations (see [49]). Consider states x^i and $x^i + \mathbf{e}_k$ where $x_k^i = 0$ and $x_m^i = 0, \forall m \in \mathcal{N}(k)$. From (9.1), we have

$$\frac{p(x^i + \mathbf{e}_k; \mathbf{r})}{p(x^i; \mathbf{r})} = \exp(r_k) = R_k$$

which is exactly the detailed balance equation between state x^i and $x^i + \mathbf{e}_k$. Such relations hold for any two states that differ in only one element, which are the only pairs that correspond to nonzero transition rates. It follows that the distribution is invariant. □

Note that the CSMA Markov chain is time-reversible since the detailed balance equations hold. In fact, the Markov chain is a reversible "spatial process" and its stationary distribution (9.1) is a Markov Random Field ([49], page 189; [54]). (This means that the state of every link k is conditionally independent of all other links, given the transmission states of its conflicting links.)

Later, we also write $p(x^i;\mathbf{r})$ as $p_i(\mathbf{r})$ for simplicity. These notations are interchangeable throughout the chapter. And let $\mathbf{p}(\mathbf{r}) \in \mathcal{R}_+^N$ be the vector of all $p_i(\mathbf{r})$'s. In Fig 9.2, for example, the probabilities of state (0,0), (1,0) and (0,1) are $1/(1+R_1+R_2)$, $R_1/(1+R_1+R_2)$ and $R_2/(1+R_1+R_2)$ in the stationary distribution.

It follows from Lemma 1 that $s_k(\mathbf{r})$, the probability that link k transmits, is given by

$$s_k(\mathbf{r}) = \sum_i [x_k^i \cdot p(x^i;\mathbf{r})] \ . \tag{9.3}$$

Without loss of generality, assume that each link k has a capacity of 1. That is, if link k transmits data all the time (without contention from other links), then its service rate is 1 (unit of data per unit time). Then, $s_k(\mathbf{r})$ is also the *normalized* throughput (or service rate) with respect to the link capacity.

Even if the waiting time and transmission time are not exponential distributed but have the same means $1/R_k$ and 1 (in fact, as long as the ratio of their means is $1/R_k$), reference [52] shows that the stationary distribution (9.1) still holds. That is, the stationary distribution is *insensitive* to the distributions of the waiting time and transmission time.

9.1.2 Adaptive CSMA for maximal throughput

CSMA *could* achieve the maximal throughput

For a $\lambda \in \mathcal{C}$, let $\bar{\mathbf{p}}$ be a probability distribution such that $\lambda = \sum_{i=1}^N \bar{p}_i x^i$. (Note that $\bar{\mathbf{p}}$ may not be unique, in which case we arbitrarily choose one such distribution.) Define the following function (the "log-likelihood function" [59] if we estimate the parameter $\mathbf{r}$ assuming that we observe $\bar{p}_i$). Note that $\bar{\mathbf{p}}$ only shows up in the derivation of our algorithm, but the information of $\bar{\mathbf{p}}$ is not needed in the algorithm itself.

$$\begin{aligned} F(\mathbf{r};\lambda) &:= \sum_i \bar{p}_i \log(p_i(\mathbf{r})) \\ &= \sum_i \bar{p}_i[\sum_{k=1}^K x_k^i r_k - \log(C(\mathbf{r}))] \\ &= \sum_k \lambda_k r_k - \log(\sum_j \exp(\sum_{k=1}^K x_k^j r_k)) \end{aligned}$$

where $\lambda_k = \sum_i \bar{p}_i x_k^i$ is the arrival rate at link k. (Note that the function $F(\mathbf{r}; \lambda)$ depends on λ, but does not involve $\bar{\mathbf{p}}$ anymore.)

Consider the following optimization problem

$$\sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda) \quad . \tag{9.4}$$

Since $\log(p_i(\mathbf{r})) \leq 0$, we have $F(\mathbf{r}; \lambda) \leq 0$. Therefore $\sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$ exists. Also, $F(\mathbf{r}; \lambda)$ is concave in $\mathbf{r}$ [56]. We show that the following proposition holds.

Proposition 20. *If* $\sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$ *is attainable (i.e., there exists* finite $\mathbf{r}^* \geq 0$ *such that* $F(\mathbf{r}^*; \lambda) = \sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$*), then* $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$*. That is, the service rate is not less than the arrival rate when* $\mathbf{r} = \mathbf{r}^*$.

Proof. Let $\mathbf{d} \geq 0$ be a vector of dual variables associated with the constraints $\mathbf{r} \geq 0$ in problem (9.4), then the Lagrangian is $\mathcal{L}(\mathbf{r}; \mathbf{d}) = F(\mathbf{r}; \lambda) + \mathbf{d}^T \mathbf{r}$. At the optimal solution $\mathbf{r}^*$, we have

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathbf{r}^*; \mathbf{d}^*)}{\partial r_k} &= \lambda_k - \frac{\sum_j x_k^j \exp(\sum_{k=1}^K x_k^j r_k^*)}{C(\mathbf{r}^*)} + d_k^* \\ &= \lambda_k - s_k(\mathbf{r}^*) + d_k^* = 0 \end{aligned} \tag{9.5}$$

where $s_k(\mathbf{r})$, according to (9.3), is the service rate (at stationary distribution) given $\mathbf{r}$. Since $d_k^* \geq 0$, $\lambda_k \leq s_k(\mathbf{r}^*)$. $\square$

Equivalently, problem (9.4) is the same as minimizing the Kullback–Leibler divergence (KL divergence) between the two distributions $\bar{\mathbf{p}}$ and $\mathbf{p}(\mathbf{r})$:

$$\inf_{\mathbf{r} \geq \mathbf{0}} D_{KL}(\bar{\mathbf{p}} || \mathbf{p}(\mathbf{r})) \tag{9.6}$$

where the KL divergence

$$\begin{aligned} D_{KL}(\bar{\mathbf{p}} || \mathbf{p}(\mathbf{r})) : &= \textstyle\sum_i [\bar{p}_i \log(\bar{p}_i / p_i(\mathbf{r}))] \\ &= \textstyle\sum_i [\bar{p}_i \log(\bar{p}_i)] - F(\mathbf{r}; \lambda). \end{aligned}$$

That is, we choose $\mathbf{r} \geq \mathbf{0}$ such that $\mathbf{p}(\mathbf{r})$ is the “closest” to $\bar{\mathbf{p}}$ in terms of the KL divergence.

The above result is related to the theory of Markov Random Fields [59] in that, when we minimize the KL divergence between a given joint distribution $\mathbf{p}_I$ and a product-form joint distribution $\mathbf{p}_{II}$, then depending on the structure of $\mathbf{p}_{II}$, certain marginal distributions induced by the two joint distributions are equal (i.e., a moment-matching condition). In our case, the time-reversible CSMA Markov chain gives the product-form distribution. Also, the arrival rate and service rate on link k are viewed as two marginal probabilities. They are not always equal, but satisfy the desired inequality in Proposition 20, due to the constraint $\mathbf{r} \geq \mathbf{0}$ which is important in our design.

The following condition, proved in section 9.7.2, ensures that $\sup_{\mathbf{r}\geq 0} F(\mathbf{r}; \lambda)$ is attainable.

Proposition 21. *If the arrival rate λ is strictly feasible, then $\sup_{\mathbf{r}\geq 0} F(\mathbf{r}; \lambda)$ is attainable.*

Combining Propositions 20 and 21, we have the following desirable result.

Theorem 2. *For any strictly feasible λ there exists a finite $\mathbf{r}^*$ such that $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$.*

Remark: To see why strict feasibility is necessary, consider the network in Fig. 9.2. If $\lambda_1 = \lambda_2 = 0.5$ (not strictly feasible), then the service rates $s_1(\mathbf{r}) = s_2(\mathbf{r}) \to 0.5$ when $r_1 = r_2 \to \infty$, but they cannot reach 0.5 for finite values of $\mathbf{r}$.

An idealized distributed algorithm

Since $\partial F(\mathbf{r}; \lambda)/\partial r_k = \lambda_k - s_k(\mathbf{r})$, a simple gradient algorithm to solve (9.4) is

$$r_k(j+1) = [r_k(j) + \alpha(j) \cdot (\lambda_k - s_k(\mathbf{r}(j)))]_+, \forall k \tag{9.7}$$

where $j = 0, 1, 2, \ldots$, and $\alpha(j)$ is some (small) step sizes. Since this is an algorithm to maximize a concave function, it is well known that with suitable decreasing step sizes $\alpha(j)$, $\mathbf{r}(j)$ converges to $\mathbf{r}^*$; and with a constant step size $\alpha(j) = \alpha, \forall j$ where α is small enough, $\mathbf{r}(j)$ converges to a neighborhood of $\mathbf{r}^*$ (and in general oscillates around the neighborhood) [56].

The most important property of algorithm (9.7), however, is that it is easy for *distributed* implementation in wireless networks, because link k can adjust r_k based on its *local information*: arrival rate λ_k and service rate $s_k(\mathbf{r}(j))$. (If the arrival rate is larger than the service rate, then r_k should be increased, and vice versa.) No information about the arrival rates and service rates of other links is needed. Note that however, the arrival and service rates are generally random variables in actual networks, unlike in (9.7). Therefore (9.7) is only an idealized algorithm which cannot be used directly.

Proposed distributed algorithm

We propose the following algorithm based on the above insight. Let link k adjust r_k at time t_j, $j = 1, 2, \ldots$. Let $t_0 = 0$ and the *update interval* $T(j) := t_j - t_{j-1}, j = 1, 2, \ldots$. Define "period j" as the time between t_{j-1} and t_j, and $\mathbf{r}(j)$ as the value of $\mathbf{r}$ set at time t_j. Let $\lambda'_k(j)$ and $s'_k(j)$ be, respectively, the empirical average arrival rate and service rate at link k between time t_j and t_{j+1}. That is, $s'_k(j) := \int_{t_j}^{t_{j+1}} x_k(\tau)d\tau/T(j+1)$, where $x_k(\tau) \in \{0, 1\}$ is the state of link k at time instance τ. Note that $\lambda'_k(j)$ and $s'_k(j)$ are generally random variables. We design the following distributed algorithm.

Algorithm 1: Adjusting the TA (transmission aggressiveness) in CSMA

At time t_{j+1} where $j = 0, 1, 2, \ldots$, let

$$r_k(j+1) = [r_k(j) + \alpha(j) \cdot (\lambda'_k(j) - s'_k(j))]_D, \forall k \tag{9.8}$$

where $\alpha(j) > 0$ is the step size, and $[\cdot]_D$ means the projection to the set $D := [0, r_{max}]$ where $r_{max} > 0$. We allow $r_{max} = +\infty$, in which case the projection is the same as $[\cdot]_+$.[3]

Clearly, each link k only uses its local information in the algorithm.

In section 9.6, we will discuss how Algorithm 1 can support the arrival rates and stabilize the queues under suitable settings of $\alpha(j), T(j)$ and r_{max}. In other words, we will explain its convergence and stability properties. (We defer this part because it is relatively technical.)

[3]A subtle point: If in period $j+1$ (for any j), the queue of link k' becomes empty, then link k' still transmits dummy packets with TA $r_{k'}(j)$ until t_{j+1}. This ensures that the (ideal) average service rate is still $s_k(\mathbf{r}(j))$ for all k. (The transmitted dummy packets are counted in the computation of $s'_k(j)$.)

We note that in a related work [46], Liu et al. carried out a convergence analysis, using a differential-equation method, of a utility maximization algorithm extended from [26] (see section 9.3 for the algorithm). However, queueing stability was not established in [46].

9.1.3 Discussion

(1) It has been believed that optimal scheduling is NP complete in general. This complexity is reflected in the mixing time of the CSMA Markov chain (i.e., the time for the Markov chain to approach its stationary distribution). In [67], the upper-bound used to quantify the mixing time is exponential in K. However, the bound may not be tight in typical wireless networks. For example, in a network where all links conflict, the CSMA Markov chain mixes much faster than the bound.

(2) There is some resemblance between the above algorithm (in particular the CSMA Markov chain) and simulated annealing (SA) [45]. SA is an optimization technique that utilizes time-reversible Markov chains to find a maximum of a function. SA can be used, for example, to find the Maximal-Weighted IS (MWIS) which is needed in Maximal-Weight Scheduling. However, note that our algorithm does not try to find the MWIS via SA. Instead, the stationary distribution of the CSMA Markov chain with a properly-chosen $\mathbf{r}^*$ is sufficient to support any vector of strictly feasible arrival rates (Theorem 2).

9.2 The primal-dual relationship

In the previous section we have described the adaptive CSMA algorithm to support any strictly-feasible arrival rates. For joint scheduling and congestion control, however, directly using the above expression of service rate (9.3) will lead to a non-convex problem. This section takes another look at the problem and also helps to avoid the difficulty.

Rewrite (9.4) as

$$\begin{array}{ll} \max_{\mathbf{r},\mathbf{h}} & \{\sum_k \lambda_k r_k - \log(\sum_j \exp(h_j))\} \\ \text{s.t.} & h_j = \sum_{k=1}^K x_k^j r_k, \forall j \\ & r_k \geq 0, \forall k. \end{array} \tag{9.9}$$

For each $j = 1, 2, \ldots, N$, associate a dual variable u_j to the constraint $h_j = \sum_{k=1}^K x_k^j r_k$. Write the vector of dual variables as $\mathbf{u} \in \mathcal{R}_+^N$. Then it is not difficult to find the dual problem of (9.9) as follows. (The computation was given in [27], but is omitted here.)

$$\begin{array}{ll} \max_{\mathbf{u}} & -\sum_i u_i \log(u_i) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \geq \lambda_k, \forall k \\ & u_i \geq 0, \sum_i u_i = 1. \end{array} \tag{9.10}$$

where the objective function is the entropy of the distribution $\mathbf{u}$, $H(\mathbf{u}) := -\sum_i u_i \log(u_i)$. [4]

Also, if for each k, we associate a dual variable r_k to the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k$ in problem (9.10), then one can compute that the dual problem of (9.10) is the original problem $\max_{\mathbf{r}\geq\mathbf{0}} F(\mathbf{r}; \lambda)$ (This is shown in section 9.7.2 as a by-product of the proof of Proposition 21). This is not surprising, since in convex optimization, the dual problem of dual problem is often the original problem.

What is interesting is that both $\mathbf{r}$ and $\mathbf{u}$ have concrete physical meanings. We have seen that r_k is the TA of link k. Also, u_i can be regarded as the probability of state x^i. This observation will be useful in later sections. A convenient way to guess this is by observing the constraint $\sum_i (u_i \cdot x_k^i) \geq \lambda_k$. If u_i is the probability of state x^i, then the constraint simply means that the service rate of link k, $\sum_i (u_i \cdot x_k^i)$, is larger than the arrival rate.

Proposition 22. *Given some (finite) TA's of the links (that is, given the dual variable* $\mathbf{r}$ *of problem (9.10)), the stationary distribution of the CSMA Markov chain maximizes the partial Lagrangian* $\mathcal{L}(\mathbf{u}; \mathbf{r}) = -\sum_i u_i \log(u_i) + \sum_k r_k (\sum_i u_i \cdot x_k^i - \lambda_k)$ *over all possible distributions* $\mathbf{u}$. *Also, Algorithm (9.7) can be viewed as a subgradient algorithm to update the dual variable* $\mathbf{r}$ *in order to solve problem (9.10).*

[4] In fact, there is a more general relationship between ML estimation problem such as (9.4) and Maximal-Entropy problem such as (9.10) [59] [60]. In [27], on the other hand, problem (9.10) was motivated by the "statistical entropy" of the CSMA Markov chain.

Proof. Given some finite dual variables $\mathbf{r}$, a partial Lagrangian of problem (9.10) is

$$\mathcal{L}(\mathbf{u};\mathbf{r}) = -\sum_i u_i \log(u_i) + \sum_k r_k(\sum_i u_i \cdot x_k^i - \lambda_k). \tag{9.11}$$

Denote $\mathbf{u}^*(\mathbf{r}) = \arg\max_{\mathbf{u}} \mathcal{L}(\mathbf{u};\mathbf{r})$, where $\mathbf{u}$ is a distribution. Since $\sum_i u_i = 1$, if we can find some w, and $\mathbf{u}^*(\mathbf{r}) > 0$ (i.e., in the interior of the feasible region) such that

$$\frac{\partial \mathcal{L}(\mathbf{u}^*(\mathbf{r});\mathbf{r})}{\partial u_i} = -\log(u_i^*(\mathbf{r})) - 1 + \sum_k r_k x_k^i = w, \forall i,$$

then $\mathbf{u}^*(\mathbf{r})$ is the desired distribution. The above conditions are

$$u_i^*(\mathbf{r}) = \exp(\sum_k r_k x_k^i - w - 1), \forall i. \text{ and } \sum_i u_i^*(\mathbf{r}) = 1.$$

By solving the two equations, we find that $w = \log[\sum_j \exp(\sum_k r_k x_k^j)] - 1$ and

$$u_i^*(\mathbf{r}) = \frac{\exp(\sum_k r_k x_k^i)}{\sum_j \exp(\sum_k r_k x_k^j)}, \forall i \tag{9.12}$$

satisfy the conditions.

Note that in (9.12), $u_i^*(\mathbf{r})$ is exactly the stationary probability of state x^i in the CSMA Markov chain given the TA $\mathbf{r}$ of all links. That is, $u_i^*(\mathbf{r}) = p(x^i;\mathbf{r}), \forall i$ (cf. (9.1)). So Algorithm (9.7) is a subgradient algorithm to search for the optimal dual variable. Indeed, given $\mathbf{r}$, $u_i^*(\mathbf{r})$ maximizes $\mathcal{L}(\mathbf{u};\mathbf{r})$; then, $\mathbf{r}$ can be updated by the subgradient algorithm (9.7), which is the deterministic version of Algorithm 1. The whole system is trying to solve problem (9.10) or (9.4). □

Let $\mathbf{r}^*$ be the optimal vector of dual variables of problem (9.10). From the above computation, we see that $\mathbf{u}^*(\mathbf{r}^*) = \mathbf{p}(\mathbf{r}^*)$, the optimal solution of (9.10), is a product-form distribution. Also, $\mathbf{p}(\mathbf{r}^*)$ can support the arrival rates λ because it is feasible to (9.10). This is another way to look at Theorem 2.

9.3 Joint scheduling and congestion control

Now, we combine congestion control with the CSMA scheduling algorithm to achieve fairness among competing flows as well as the maximal throughput. Here, the input rates are distributedly adjusted by the source of each flow.

9.3.1 Formulation and algorithm

Assume there are M flows, and let m be their index ($m = 1, 2, \ldots, M$). Define $a_{mk} = 1$ if flow m uses link k, and $a_{mk} = 0$ otherwise. Let f_m be the rate of flow m, and $v_m(f_m)$ be the "utility function" of this flow, which is assumed to be increasing and strictly concave. Assume all links have the same PHY data rates (it is easy to extend the algorithm to different PHY rates).

Assume that each link k maintains a separate queue for each flow that traverses it. Then, the service rate of flow m by link k, denoted by s_{km}, should be no less than the incoming rate of flow m to link k. For flow m, if link k is its first link (i.e., the source link), we say $\delta(m) = k$. In this case, the constraint is $s_{km} \geq f_m$. If $k \neq \delta(m)$, denote flow m's upstream link of link k by $up(k, m)$, then the constraint is $s_{km} \geq s_{up(k,m),m}$, where $s_{up(k,m),m}$ is equal to the incoming rate of flow m to link k. We also have $\sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}, \forall k$, i.e., the total service rate of link k is not less than the sum of all flow rates on the link.

Then, consider the following optimization problem:

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{s},\mathbf{f}} & -\sum_i u_i \log(u_i) + \beta \sum_{m=1}^{M} v_m(f_m) \\ \text{s.t.} & s_{km} \geq 0, \forall k, m : a_{mk} = 1 \\ & s_{km} \geq s_{up(k,m),m}, \forall m, k : a_{mk} = 1, k \neq \delta(m) \\ & s_{km} \geq f_m, \forall m, k : k = \delta(m) \\ & \sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}, \forall k \\ & u_i \geq 0, \sum_i u_i = 1. \end{array} \tag{9.13}$$

where $\beta > 0$ is a constant weighting factor.

Notice that the objective function is not exactly the total utility, but it has an extra term $-\sum_i u_i \log(u_i)$. As will be further explained in section 9.3.2, when β is large, the "importance" of the total utility dominates the objective function of (9.13). (This is similar in spirit to the weighting factor used in [44].) As a result, the solution of (9.13) approximately achieves the maximal utility. Associate dual variables $q_{km} \geq 0$ to the 2nd and 3rd lines of constraints of (9.13). Then a partial Lagrangian (subject to $s_{km} \geq 0$,

$\sum_i u_i \cdot x_k^i \geq \sum_{m:a_{mk}=1} s_{km}$ and $u_i \geq 0, \sum_i u_i = 1$) is

$$\begin{aligned} & \mathcal{L}(\mathbf{u}, \mathbf{s}, \mathbf{f}; \mathbf{q}) \\ = \quad & -\textstyle\sum_i u_i \log(u_i) + \beta \sum_{m=1}^M v_m(f_m) \\ & + \textstyle\sum_{m,k:a_{mk}=1, k \neq \delta(m)} q_{km}(s_{km} - s_{up(k,m),m}) \\ & + \textstyle\sum_{m,k:,k=\delta(m)} q_{km}(s_{km} - f_m) \\ = \quad & -\textstyle\sum_i u_i \log(u_i) \\ & +\beta \textstyle\sum_{m=1}^M v_m(f_m) - \sum_{m,k:k=\delta(m)} q_{km} f_m \\ & + \textstyle\sum_{k,m:a_{mk}=1} [s_{km} \cdot (q_{km} - q_{down(k,m),m})] \end{aligned} \tag{9.14}$$

where $down(k, m)$ means flow m's downstream link of link k (Note that $down(up(k, m), m) = k$). If k is the last link of flow m, then define $q_{down(k,m),m} = 0$.

Fix the vectors $\mathbf{u}$ and $\mathbf{q}$ first, we solve for s_{km} in the sub-problem

$$\begin{aligned} \max_{\mathbf{s}} \quad & \textstyle\sum_{k,m:a_{mk}=1} [s_{km} \cdot (q_{km} - q_{down(k,m),m})] \\ \text{s.t.} \quad & s_{km} \geq 0, \forall k, m : a_{mk} = 1 \\ & \textstyle\sum_{m:a_{mk}=1} s_{km} \leq \sum_i (u_i \cdot x_k^i), \forall k. \end{aligned} \tag{9.15}$$

The solution is easy to find (similar to [24] and related references therein): at link k, denote $z_k := \max_{m:a_{mk}=1}(q_{km} - q_{down(k,m),m})$. (i) If $z_k > 0$, then for a $m' \in \arg\max_{m:a_{mk}=1}(q_{km} - q_{down(k,m),m})$, let $s_{km'} = \sum_i (u_i \cdot x_k^i)$ and let $s_{km} = 0, \forall m \neq m'$. In other words, link k serves a flow with the maximal back-pressure $q_{km} - q_{down(k,m),m}$. (ii) If $z_k \leq 0$, then let $s_{km}(j) = 0, \forall m$, i.e., link k does not serve any flow. Since the value of $q_{down(k,m),m}$ can be obtained from a one-hop neighbor, this algorithm is distributed. (In practice, the value of $q_{down(k,m),m}$ can be piggybacked in the ACK packet in IEEE 802.11.)

Plugging the solution of (9.15) back into (9.14), we get

$$\begin{aligned} \mathcal{L}(\mathbf{u}, \mathbf{f}; \mathbf{q}) \quad = \quad & [-\textstyle\sum_i u_i \log(u_i) + \sum_k (z_k)_+ (\sum_i u_i \cdot x_k^i)] \\ & +[\beta \textstyle\sum_{m=1}^M v_m(f_m) - \sum_{m,k:k=\delta(m)} q_{km} f_m] \end{aligned}$$

where z_k is the maximal back-pressure at link k. So a distributed algorithm to solve (9.13) is as follows. Denote by Q_{km} the actual queue length of flow m at link k. For simplicity, assume that $v'_m(0) \leq V < \infty, \forall m$, i.e., the derivative of all utility functions at 0 is bounded by some $V < \infty$.

Algorithm 2: Joint scheduling and congestion control

Initially, assume that all queues are empty (i.e., $Q_{km}(0) = 0, \forall k, m$), and let $q_{km}(0) = 0, \forall k, m$. As before, the update interval $T(j) = t_j - t_{j-1}$ and $t_0 = 0$. Here we use constant step sizes and update intervals $\alpha(j) = \alpha, T(j) = T, \forall j$. The variables $\mathbf{q}, \mathbf{f}, \mathbf{r}$ are iteratively updated at time t_j, $j = 1, 2, \ldots$. Let $\mathbf{q}(j), \mathbf{f}(j), \mathbf{r}(j)$ be their values set at time t_j. Denote by $s^{'}_{km}(j)$ the empirical average service rate of flow m at link k in period $j+1$ (i.e., the time between t_j and t_{j+1}).

- Scheduling: In period $j+1$, link k lets its TA be $r_k(j) = [z_k(j)]_+$ in the CSMA operation, where $z_k(j) = \max_{m:a_{mk}=1}(q_{km}(j) - q_{down(k,m),m}(j))$. (The rationale is that, given $\mathbf{z}(j)$, the $\mathbf{u}^*$ that maximizes $\mathcal{L}(\mathbf{u}, \mathbf{f}; \mathbf{q}(j))$ over $\mathbf{u}$ is the stationary distribution of the CSMA Markov Chain with $r_k(j) = [z_k(j)]_+$, similar to the proof of Proposition 22.) Choose a flow $m' \in \arg\max_{m:a_{mk}=1}(q_{km}(j) - q_{down(k,m),m}(j))$. When link k gets the opportunity to transmit, (i) if $z_k(j) > 0$, it serves flow m'; (Similar to Algorithm 1, the dummy packets transmitted by link k, if any, are counted in $s^{'}_{km'}(j)$.) (ii) if $z_k(j) \leq 0$, then it transmits dummy packets. These dummy packets are not counted, i.e., let $s^{'}_{km}(j) = 0, \forall m$. Also, they are not put into any actual queue at the receiver of link k. (A simpler alternative is that link k keeps silent if $z_k(j) \leq 0$. That case can be similarly analyzed following the method in section 9.7.3.)

- Congestion control: For each flow m, if link k is its source link, the transmitter of link k lets the flow rate in period $j+1$ be $f_m(j) = \arg\max_{\hat{f}_m \in [0,1]}\{\beta \cdot v_m(\hat{f}_m) - q_{km}(j) \cdot \hat{f}_m\}$. (This maximizes $\mathcal{L}(\mathbf{u}, \mathbf{f}; \mathbf{q}(j))$ over $\mathbf{f}$.)

- The dual variables q_{km} (maintained by the transmitter of each link) are updated (similar to a subgradient algorithm). At time t_{j+1}, let $q_{km}(j+1) = [q_{km}(j) - \alpha \cdot s^{'}_{km}(j))]_+ + \alpha \cdot s^{'}_{up(k,m),m}(j)$ if $k \neq \delta(m)$; and $q_{km}(j+1) = [q_{km}(j) - \alpha \cdot s^{'}_{km}(j))]_+ + \alpha \cdot f_m(j)$ if $k = \delta(m)$. (By doing this, approximately $q_{km} \propto Q_{km}$.)

Remark 1: As $T \to \infty$ and $\alpha \to 0$, Algorithm 2 approximates the "ideal" algorithm that solves (9.13), due to the convergence of the CSMA Markov chain in each period. A bound of the achievable utility of Algorithm 2, compared to the optimal total utility $\bar{W}$ defined in

(9.16) is given in section 9.7.3. The bound, however, is not very tight, since our simulation shows good performance without a very large T or a very small α.

Remark 2: In section 9.7.4, we show that by using similar techniques, the adaptive CSMA algorithm can be combined with optimal routing, anycast or multicast with network coding. So it is a modular MAC-layer protocol which can work with other protocols in the transport layer and the network layer.

Remark 3: Coincidentally, the authors of [68] implemented a protocol similar to Algorithm 2 using 802.11e hardware and it shows superior performance compared to normal 802.11. There, according to the backpressure, a flow chooses from a discrete set of contention windows, or "CW's" (where a smaller CW corresponds to a larger TA). We note that, however, different from our work, [68] only focuses on implementation study, without theoretical analysis. Therefore, the potential optimality of CSMA is not shown in [68]. Also, the CW's there are set in a more heuristic way.

9.3.2 Approaching the maximal utility

We now show that the solution of (9.13) approximately achieves the maximal utility when β is large. Denote the maximal total utility achievable by $\bar{W}$, i.e.,

$$\bar{W} := \max_{\mathbf{u},\mathbf{s},\mathbf{f}} \textstyle\sum_m v_m(f_m) \tag{9.16}$$

subject to the same constraints as in (9.13). Assume that $\mathbf{u} = \bar{\mathbf{u}}$ when (9.16) is solved. Also, assume that in the optimal solution of (9.13), $\mathbf{f} = \hat{\mathbf{f}}$ and $\mathbf{u} = \hat{\mathbf{u}}$. We have the following bound.

Proposition 23. *The difference between the total utility ($\sum_{m=1}^{M} v_m(\hat{f}_m)$) resulting from solving (9.13) and the maximal total utility $\bar{W}$ is bounded. The bound of difference decreases with the increase of β. In particular,*

$$\bar{W} - (K \cdot \log 2)/\beta \le \textstyle\sum_m v_m(\hat{f}_m) \le \bar{W}. \tag{9.17}$$

Proof. Notice that $H(\mathbf{u}) = -\sum_i u_i \log(u_i)$, the entropy of the distribution $\mathbf{u}$, is bounded.

Indeed, since there are $N \leq 2^K$ possible states, one has $0 \leq H(\mathbf{u}) \leq \log N \leq \log 2^K = K \log 2$.

Since in the optimal solution of problem (9.13), $\mathbf{f} = \hat{\mathbf{f}}$ and $\mathbf{u} = \hat{\mathbf{u}}$, we have $H(\hat{\mathbf{u}}) + \beta \sum_m v_m(\hat{f}_m) \geq H(\bar{\mathbf{u}}) + \beta \bar{W}$. So $\beta[\sum_m v_m(\hat{f}_m) - \bar{W}] \geq H(\bar{\mathbf{u}}) - H(\hat{\mathbf{u}}) \geq -H(\hat{\mathbf{u}}) \geq -K \cdot \log 2$. Also, clearly $\bar{W} \geq \sum_{m=1}^{M} v_m(\hat{f}_m)$, so (9.17) follows. $\square$

9.4 Reducing the queueing delay

Consider a strictly feasible arrival rate vector λ in the scheduling problem in section 9.1. With Algorithm 1, the long-term average service rates are in general not strictly higher than the arrival rates, so traffic suffers from queueing delay when traversing the links. To reduce the delay, consider a modified version of problem (9.10):

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{w}} & -\sum_i u_i \log(u_i) + c \sum_k \log(w_k) \\ \text{s.t.} & \sum_i (u_i \cdot x_k^i) \geq \lambda_k + w_k, \forall k \\ & u_i \geq 0, \sum_i u_i = 1 \\ & 0 \leq w_k \leq \bar{w}, \forall k \end{array} \tag{9.18}$$

where $0 < c < 1$ is a small constant. Note that we have added the new variables $w_k \in [0, \bar{w}]$ (where $\bar{w}$ is a constant upper bound), and require $\sum_i u_i \cdot x_k^i \geq \lambda_k + w_k$. In the objective function, the term $c \cdot \log(w_k)$ is a penalty function to avoid w_k being too close to 0.

Since λ is in the interior of the capacity region, there is a vector λ' also in the interior and satisfying $\lambda' > \lambda$ component-wise. So there exist $\mathbf{w}' > 0$ and $\mathbf{u}'$ (such that $\sum_i u'_i x_k^i = \lambda'_k := \lambda_k + w'_k, \forall k$) satisfying the constraints. Therefore, in the optimal solution, we have $w_k^* > 0, \forall k$ (otherwise the objective function is $-\infty$, smaller than the objective value that can be achieved by $\mathbf{u}'$ and $\mathbf{w}'$). Thus $\sum_i u_i^* \cdot x_k^i \geq \lambda_k + w_k^* > \lambda_k$. This means that the service rate is strictly larger than the arrival rate, bringing the extra benefit that the queue lengths tend to decrease to 0.

Similar to section 9.2, we form a partial Lagrangian (with $\mathbf{y} \geq 0$ as dual variables)

$$\begin{aligned}\mathcal{L}(\mathbf{u}, \mathbf{w}; \mathbf{y}) &= -\sum_i u_i \log(u_i) + c\sum_k \log(w_k) + \\ &\quad \sum_k [y_k(\sum_i u_i \cdot x_k^i - \lambda_k - w_k)] \\ &= [-\sum_i u_i \log(u_i) + \sum_k (y_k \sum_i u_i \cdot x_k^i)] + \\ &\quad \sum_k [c \cdot \log(w_k) - y_k w_k] - \sum_k (y_k \lambda_k).\end{aligned} \tag{9.19}$$

Note that the only difference from (9.11) is the extra term $\sum_k [c \cdot \log(w_k) - y_k w_k]$. Given $\mathbf{y}$, the optimal $\mathbf{w}$ is $w_k = \min\{c/y_k, \bar{w}\}, \forall k$, and the optimal $\mathbf{u}$ is the stationary distribution of the CSMA Markov Chain with $\mathbf{r} = \mathbf{y}$. Therefore the (sub)gradient algorithm to update $\mathbf{y}$ is $y_k \leftarrow y_k + \alpha(\lambda_k + w_k - s_k(\mathbf{y}))$.

Since $\mathbf{r} = \mathbf{y}$, we have the following localized algorithm at link k to update r_k. Notice its similarity to Algorithm 1.

Algorithm 3: Enhanced Algorithm 1 to reduce queueing delays

At time t_{j+1} where $j = 0, 1, 2, \ldots$, let

$$r_k(j+1) = [r_k(j) + \alpha(j) \cdot (\lambda'_k(j) + \min\{c/r_k(j), \bar{w}\} - s'_k(j))]_D \tag{9.20}$$

for all k, where $\alpha(j)$ is the step size, and $D = [0, r_{max}]$ where r_{max} can be $+\infty$. As in Algorithm 1, even when link k' has no backlog (i.e., zero queue length), we let it send dummy packets with its current aggressiveness $r_{k'}$. This ensures that the (ideal) average service rate of link k is $s_k(\mathbf{r}(j))$ for all k.

Since Algorithm 3 "pretends" to serve some arrival rates higher than the actual arrival rates (due to the positive term $\min\{c/r_k(j), \bar{w}\}$), Q_k is not only stable, but also tends to be small. The convergence and stability properties of Algorithm 3 when $r_{max} = \infty$ are discussed in (i) of Appendix C. If $r_{max} < \infty$, the properties are similar to those in (ii) of Appendix C.

For joint CSMA scheduling and congestion control, a simple way to reduce the delay, similar to [69], is as follows. In item 2 ("congestion control") of Algorithm 2, let the actual flow rate be $\rho \cdot f_m(j)$ where ρ is slightly smaller than 1, and keep other parts of the algorithm unchanged. Then, each link provides a service rate higher than the actual arrival rate. So the delay is reduced with a small cost in the flow rates.

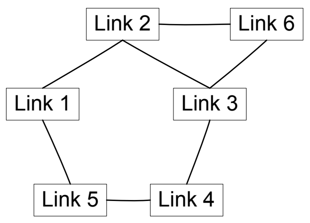
9.5 Simulations

9.5.1 CSMA scheduling: i.i.d. input traffic with fixed average rates

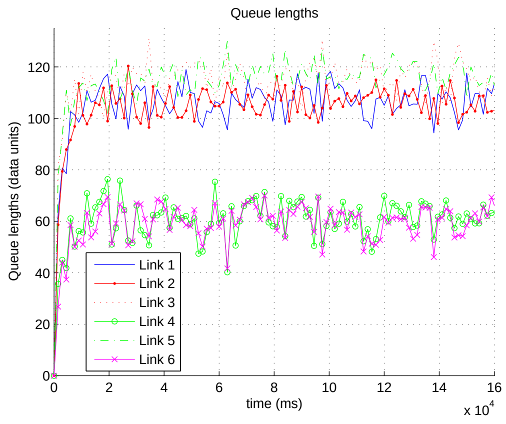
In our C++ simulations, the transmission time of all links is exponentially distributed with mean 1ms, and the backoff time of link k is exponentially distributed with mean $1/\exp(r_k)$ ms. The capacity of each link is 1(data unit)/ms. There are 6 links in "Network 1", whose CG is shown in Fig. 9.3 (a). Define $0 \leq \rho < 1$ as the "load factor", and let $\rho = 0.98$ in this simulation. The arrival rate vector is set to λ=ρ*[0.2*(1,0,1,0,0,0) + 0.3*(1,0,0,1,0,1) + 0.2*(0,1,0,0,1,0) + 0.3*(0,0,1,0,1,0)] = ρ*(0.5,0.2,0.5,0.3,0.5,0.3) (data units/ms). We have multiplied by $\rho < 1$ a convex combination of some maximal ISs to ensure that $\lambda \in \mathcal{C}$.

Initially, all queues are empty, and the initial value of r_k is 0 for all k. r_k is then adjusted using Algorithm 1 once every $T = 5ms$ (i.e., $T(j) = T, \forall j$), with a constant step size $\alpha(j) = \alpha = 0.23, \forall j$. Fig. 9.3 (b) shows the evolution of the queue lengths with $r_{max} = 8$. They are stable despite some oscillations. The vector $\mathbf{r}$ is not shown since in this simulation, it is roughly α/T times the queue lengths. Fig. 9.3 (c) shows the evolution of queue lengths using Algorithm 3 with $c = 0.01$, $\bar{w} = 0.02$ and all other parameters unchanged. The algorithm drives the queue lengths to around zero, thus significantly reducing the queueing delays.

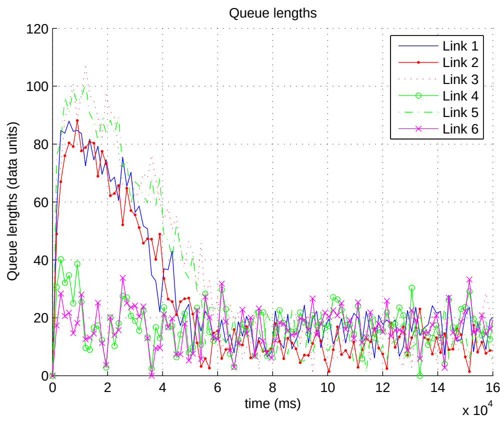
Fig 9.4 shows the results of Algorithm 3 with $\alpha(j) = 0.46/[(2+j/1000)\log(2+j/1000)]$ and $T(j) = (2 + j/1000)$ ms, which satisfy the conditions for convergence in section 9.6. The constants $c = 0.01$, $\bar{w} = 0.02$, and $r_{max} = \infty$. To show the negative drift of queues, assume that initially, all queue lengths are 300 data units in Fig 9.4. We see that the TA vector $\mathbf{r}$ converges (Fig 9.4 (a)), and the queues tend to decrease and are stable (Fig 9.4



(a) Conflict Graph

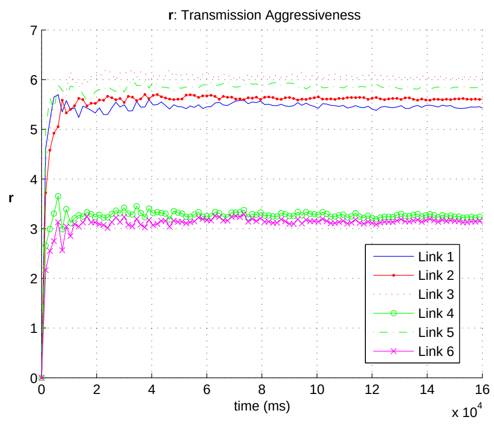


(b) Queue lengths, with constant step size. The vector **r** is not shown since it is proportional to the queue lengths.

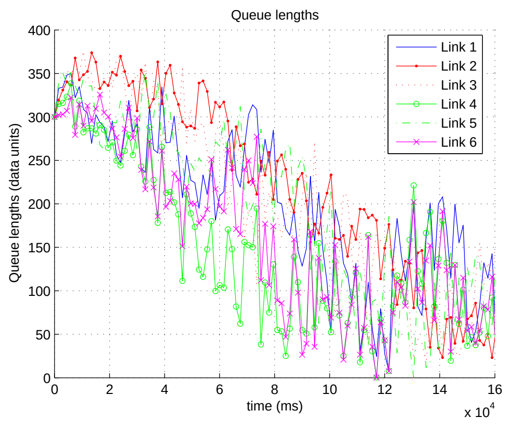


(c) Queue lengths (with Algorithm 3)

Figure 9.3: Adaptive CSMA Scheduling (Network 1)



(a) The vector $\mathbf{r}$



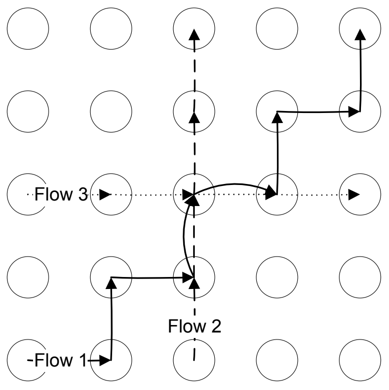
(b) Queue lengths

Figure 9.4: Decreasing step sizes

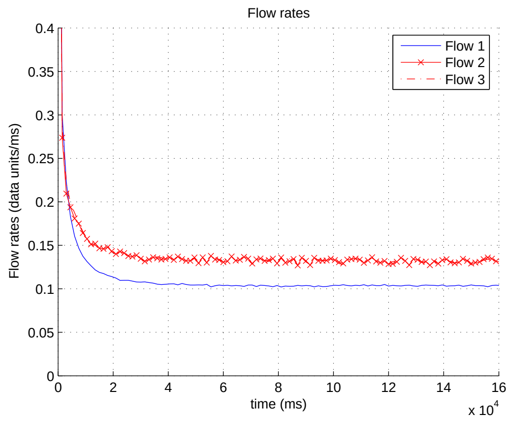
(b)). However, there are more oscillations in the queue lengths than the case with constant step size. This is because when $\alpha(j)$ becomes smaller when j is large, $\mathbf{r}(j)$ becomes less responsive to the variations of queue lengths.

9.5.2 Joint scheduling and congestion control

In Fig 9.5, we simulate a more complex network ("Network 2"). We also go one step further than Network 1 by giving the actual locations of the nodes, not only the CG. Fig 9.5 (a) shows the network topology, where each circle represents a node. The nodes are arranged in a grid for convenience, and the distance between two adjacent nodes (horizontally or vertically) is 1. Assume that the transmission range is 1, so that a link can only be formed by two adjacent nodes. Assume that two links cannot transmit simultaneously if there are two nodes, one in each link, being within a distance of 1.1 (In IEEE 802.11, for example, DATA and ACK packets are transmitted in opposite directions. This model considers the interference among the two links in both directions). The paths of 3 multi-hop flows are plotted. The utility function of each flow is $v_m(f_m) = \log(f_m + 0.01)$. The weighting factor



(a) Network 2 and flow directions



(b) Flow rates

Figure 9.5: Flow rates in Network 2 (Grid Topology) with Joint scheduling and congestion control

is $\beta = 3$. (Note that the input rates are adjusted by the congestion control algorithm instead of being specified as in the last subsection.)

Fig 9.5 (b) shows the evolution of the flow rates, using Algorithm 2 with $T = 5ms$ and $\alpha = 0.23$. We see that they become relatively constant after an initial convergence. By directly solving (9.16) centrally, we find that the theoretical optimal flow rates for the three flows are 0.11, 0.134 and 0.134 (data unit/ms), very close to the simulation results. The queue lengths are also stable (in fact, uniformly bounded as proved in section 9.7.3).

9.6 Throughput-optimality of the CSMA scheduling algorithm

9.6.1 Overview

Now we explain the throughput-optimality of Algorithm 1. First, it is useful to extend the algorithm a little bit as follows.

$$r_k(j+1) = [r_k(j) + \alpha(j) \cdot (\lambda'_k(j) + h(r_k(j)) - s'_k(j))]_D \tag{9.21}$$

where $D := [0, r_{max}]$ and the function $h(\cdot) \geq 0$. If $h(\cdot) = 0$, then algorithm (9.21) reduces to Algorithm 1. If $h(\cdot) > 0$, then algorithm (9.21) "pretends" to serve some arrival rates higher than the actual ones. The benefit of this is that the queue lengths tend to decrease since the algorithm serves a bit more packets than needed.

The intuition for the throughput-optimality is that one can make $\mathbf{r}$ change slowly (i.e., "quasi-static") to allow the CSMA Markov chain to approach its stationary distribution (and thus obtaining good estimation of $s_k(\mathbf{r})$). This allows the separation of time scales of the dynamics of $\mathbf{r}(j)$ and the CSMA Markov chain.

Before proceeding, we need to give more precise definitions of queue stability and throughput optimality.

For simplicity, assume the following i.i.d. Bernoulli arrivals (although it can be readily generalized [67]): Let $a_k(t) \in \{0, 1\}$ be the arrival process at link k. For $t \in [j, j+1], j = 0, 1, 2, \ldots$ (i.e., in a given "slot" with length 1), $a(t) = 1$ with probability λ_k and $a(t) = 0$ otherwise. Then, $A_k(t) := \int_0^t a_k(\tau) d\tau$, the cumulative amount of arrived traffic at link k by time t, satisfies that $E(A_k(t))/t = \lambda_k$.

Let $x_k(t) \in \{0, 1\}$ be the instantaneous transmission state of link k at (continuous) time t. For link k, define the cumulative "service" by time t as $S_k(t) = \int_{\tau=0}^t x_k(\tau) d\tau$, and the cumulative departure by time t as $D_k(t) = \int_{\tau=0}^t x_k(\tau) I(Q_k(\tau) > 0) d\tau$, where $I(\cdot)$ is the indicator function and $Q_k(\tau) := Q_k(0) + A_k(\tau) - D_k(\tau)$ is the queue length of link k at time τ. Note that there is no departure if the queue is empty but we allow $x_k(\tau) = 1$ even if $Q_k(\tau) = 0$ (in which case dummy packets are sent).

Definition 3. *The queues are "rate stable" if* $\lim_{t \to \infty} [A_k(t) - D_k(t)]/t = 0, \forall k$ *almost surely.*

Another notion of stability is the positive (Harris) recurrence of the network Markov chain.

Definition 4. *An algorithm is said to be "throughput-optimal" if for any* $\lambda \in \mathcal{C}$, *it makes* *the system rate stable or positive (Harris) recurrent. In this case, we also say that the algorithm achieves the "maximal throughput".*

Now, we give a summary of the properties of algorithm (9.21).

(i) With properly-chosen constant step sizes $\alpha(j) = \alpha, \forall j$ and update intervals $T(j) = T, \forall j$, one can arbitrarily approximate the maximal throughput.

(ii) With properly-chosen decreasing step sizes and increasing update intervals (e.g., $\alpha(j) = 1/[(j+2)\log(j+2)]$, $T(j) = j+2$) and function $h(\cdot)$, and with $r_{max} = +\infty$, for any $\lambda \in \mathcal{C}$, the vector $\mathbf{r}(j)$ converges and the queues are rate-stable. Therefore the algorithm is throughput-optimum.

(iii) In a variant of the algorithm where $\mathbf{r}(j)$ is guaranteed to be bounded, with suitable decreasing step sizes and *constant* update intervals, one can arbitrarily approximate the maximal throughput.

9.6.2 More formal results

We now state the main results about algorithm (9.21) and its variants, and give proof sketches in the next section.

For constant step sizes and update intervals, we have the following.

Theorem 3. *Let $r_{max} < +\infty$ and $h(r_k(j)) = \epsilon > 0$ in algorithm (9.21). Define the capacity region*

$$\begin{aligned}\mathcal{C}'(r_{max}, \epsilon) : \;&=\; \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C}, \text{ and} \\ &\qquad \arg\max_{\mathbf{r} \geq \mathbf{0}} F(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1}) \in [0, r_{max}]^K\}\end{aligned}$$

If $\lambda \in \mathcal{C}'(r_{max}, \epsilon)$, then there exist constant step size $\alpha(j) = \alpha$ and update interval $T(j) = T$ such that all queues are stable.

Remark*: Note that $\mathcal{C}'(r_{max}, \epsilon) \to \mathcal{C}$ as $r_{max} \to +\infty$ and $\epsilon \to 0$. So the maximal throughput can be arbitrarily approximated by setting r_{max} and ϵ.*

For time-varying step sizes and update intervals, we have the following.

Theorem 4. *Assume that λ is strictly feasible (i.e., $\lambda \in \mathcal{C}$). Also assume that there is a maximal instantaneous arrival rate $\bar{\lambda}$ for any link (i.e., $\lambda'_k(j) \leq \bar{\lambda}, \forall k, j$). In algorithm (9.21), let $h(\cdot) = 0$ and $r_{max} = +\infty$. Denote $\Delta(m) := \sum_{j=0}^{m-1} \alpha(j)$. Choose $\alpha(j)$ and non-decreasing $T(j)$ such that*

$$\alpha(j) > 0, \sum_j \alpha(j) = \infty, \sum_j \alpha^2(j) < \infty \tag{9.22}$$

$$\sum_{m=1}^{\infty} [\alpha(m)\Delta(m)]^2 < \infty \tag{9.23}$$

$$\sum_{m=1}^{\infty} [\alpha(m) \cdot \Delta(m) \cdot f(m)/T(m+1)] < \infty \tag{9.24}$$

where

$$f(m) := \exp\{(\tfrac{5}{2}K + 1) \cdot [\lambda_{max} \cdot \Delta(m) + \log(2)]\} \tag{9.25}$$

where K is the number of links, and $\lambda_{max} := \bar{\lambda}$.

Then with algorithm (9.21), $\mathbf{r}(j)$ converges to some $\mathbf{r}^*$ with probability 1. The vector $\mathbf{r}^*$ satisfies that $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$. Also, the queues are rate stable.

Remark: Besides "rate stability", another notion of stability is the positive (Harris) recurrence of the underlying network Markov process, in particular the queue lengths. Note that with the time-varying step sizes and update intervals in the setup of Theorem 4, the Markov process is not time-homogeneous, in which case positive (Harris) recurrence is not well defined. This is the reason why we choose to prove the "rate stability". One concern for rate stability is that the queue lengths may go to infinity, since "rate stability" only ensures that $\lim_{t\to\infty}[A_k(t) - D_k(t)]/t = \lim_{t\to\infty} Q_k(t)/t = 0, \forall k$. However, this issue can be avoided in an enhanced algorithm in Theorem 5.

What $\alpha(j), T(j)$ satisfy the above conditions?

Proposition 24. *The setting $\alpha(j) = 1/[(j+2)\log(j+2)]$ and $T(j) = j+2$ satisfies conditions (9.22), (9.23) and (9.24). Note that this setting* does not depend on, or require the knowledge of K and λ_{max}, and thus can generally apply to any network.

Similarly, the same is true for the following settings. (i) $\alpha(j) = 1/[(j+2)\log(j+2)]$ *and* $T(j) = (j+2)^{\gamma}$ *for any* $\gamma > 0$*; (ii)* $\alpha(j) = c_0/[(a \cdot j + b + 2)\log(a \cdot j + b + 2)]$ *and* $T(j) = a \cdot j + b + 2$ *(with constants* $a > 0, b > 0, c_0 > 0$*).*

Prop. 24 is not difficult to check [67].

An enhancement which leads to smaller queues are given in the following theorem.

Theorem 5. *We make the same assumptions as in Theorem 4, with the only differences that* $h(r_k(j)) = \min\{c/r_k(j), \bar{w}\}$ *where* $c, \bar{w} > 0$ *(see section 9.4 for an explanation of the function), and* $\lambda_{max} := \bar{\lambda} + \bar{w}$*.*

Then, $\mathbf{r}(j)$ *converges to some* $\mathbf{r}^*$ *with probability 1. The vector* $\mathbf{r}^*$ *satisfies that* $s_k(\mathbf{r}^*) > \lambda_k, \forall k$*. Also, the queues are rate stable, and return to 0 infinitely often w. p. 1.*

As explained in section 9.4, setting $h(r_k(j))$ this way ensures that the convergent point $\mathbf{r}^*$ makes the average service rates strictly larger than the arrival rates. As a result, in addition to the results of Theorem 4, the queues also return to 0 infinitely often and tend to be small.

Also, all settings in Prop. 24 still satisfy conditions (9.22), (9.23) and (9.24) with the new definition of λ_{max}.

In a variant of the algorithm, one can use decreasing $\alpha(j)$ and constant update intervals $T(j)$ (instead of increasing $T(j)$). However, this variant requires that $\mathbf{r}(j)$ be bounded. Therefore, it can only *approximate*, but not achieve, the maximal throughput. The variant is

$$r_k(j+1) = r_k(j) + \alpha(j) \cdot [\lambda'_k(j) + \epsilon - s'_k(j) + \bar{h}(r_k(j))]. \tag{9.26}$$

Note that there is no projection in (9.26). Instead, $\bar{h}(r_k(j))$ is used to bound $\mathbf{r}(j)$ in a "softer" way:

$$\bar{h}(y) = \begin{cases} r_{min} - y & \text{if } y < r_{min} \\ 0 & \text{if } y \in [r_{min}, r_{max}] \\ r_{max} - y & \text{if } y > r_{max} \end{cases} \tag{9.27}$$

Then, the following can be shown.

Theorem 6. *Assume that*

$$\begin{aligned} \lambda &\in \mathcal{C}(r_{min}, r_{max}, \epsilon) \\ &:= \{\lambda | \arg\max_{\mathbf{r}} F(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1}) \in (r_{min}, r_{max})^K\}. \end{aligned}$$

Then, if $\alpha(j) > 0$ is non-increasing and satisfies $\sum_j \alpha(j) = \infty$, $\sum_j \alpha(j)^2 < \infty$ and $\alpha(0) \leq 1$, then $\mathbf{r}(j)$ converges to $\mathbf{r}^*$ as $i \to \infty$ with probability 1, where $\mathbf{r}^*$ satisfies $s_k(\mathbf{r}^*) = \lambda_k + \epsilon > \lambda_k, \forall k$. Also, the queues are rate stable and return to 0 infinitely often.

Remark: Clearly, as $r_{min} \to -\infty$, $r_{max} \to \infty$ and $\epsilon \to 0$, $\mathcal{C}(r_{min}, r_{max}, \epsilon) \to \mathcal{C}$. So the maximal throughput can be arbitrarily approximated by setting r_{max}, r_{min} and ϵ.

9.6.3 Proof sketches

We now give proof sketches of the above theorems. The full proofs are in [67].

Proof sketch of Theorem 3

The basic idea is that when $T(j) = T$ is large enough, $s'_k(j)$ and $\lambda'_k(j)$ are very close to $s_k(\mathbf{r}(j))$ and λ_k. If $\alpha(j) = \alpha$ is small enough, then the algorithm approximately solves $\max_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1})$. So the average service rate at any link k is close to $\lambda_k + \epsilon$ which is larger than the arrival rate λ_k. Therefore, all queues are stable.

Proof sketch of Theorem 4

Some notation

Before proving Theorem 4, we need some notation. Let $x^0(m)$ be the state of the CSMA Markov chain at time t_m. Define the random vector $U(m) := (\mathbf{s}'(m-1), \lambda'(m-1), \mathbf{r}(m), x^0(m))$ for $m \geq 1$ and $U(0) = (\mathbf{r}(0) = \mathbf{0}, x^0(0))$. Let $\mathcal{F}_m$ be the σ-field generated by $U(0), U(1), \ldots, U(m)$. $\mathcal{F}_m$ represents what happened up to time t_m.

Given a vector of TA $\mathbf{r}(m)$ at time t_m of algorithm (9.21), the vector $\mathbf{g}(m)$ whose k-th element $g_k(m) := \lambda_k - s_k(\mathbf{r}(m))$ is the gradient of $F(\mathbf{r}(m); \lambda)$. To find the desired $\mathbf{r}^* = \arg\max_{\mathbf{r}\geq 0} F(\mathbf{r}; \lambda)$, the ideal algorithm (9.7) would follow the direction of $\mathbf{g}(m)$. However, we only have an estimation of $g_k(m)$, denoted by

$$g'_k(m) = \lambda'_k(m) - s'_k(m). \tag{9.28}$$

Denote $E(\cdot|\mathcal{F}_m)$ by $E_m(\cdot)$. The "error bias" of $g'_k(m)$ is defined as

$$\begin{aligned} B_k(m) : &= E_m(g'_k(m)) - g_k(m) \\ &= [E_m(\lambda'_k(m)) - \lambda_k] - \\ &\quad [E_m(s'_k(m)) - s_k(\mathbf{r}(m))]. \end{aligned} \tag{9.29}$$

Define also the zero-mean "noise"

$$\begin{aligned} \eta_k(m) : &= [\lambda'_k(m) - E_m(\lambda'_k(m))] \\ &\quad -[s'_k(m) - E_m(s'_k(m))]. \end{aligned}$$

Since both $s'_k(m)$ and $\lambda'_k(m)$ are bounded, the noise is also bounded: $|\eta_k(m)| \leq c_2$ for some $c_2 > 0$. Then, we have

$$g'_k(m) = g_k(m) + B_k(m) + \eta_k(m). \tag{9.30}$$

We now sketch the proof in three steps, with the complete proof in [67]. These steps are relatively independent–if one step can be modified or strengthened later, other steps can still apply.

Step 1: Bounding the error bias $B_k(m), m = 1, 2, \ldots$

This step shows that the error bias $B_k(m)$ (9.29) decreases "fast enough" with time. This is done by bounding the two parts of $B_k(m)$.

(i) First, we show that $\forall k$,

$$\begin{aligned} &|E_m[s'_k(m)] - s_k(\mathbf{r}(m))| \leq \\ &2K \exp\{(\frac{5}{2}K + 1)[||\mathbf{r}(m)||_\infty + \log(2)]\}/T_{m+1} \end{aligned} \tag{9.31}$$

where T_{m+1} means $T(m+1)$. This is obtained by analyzing the mixing time of the CSMA Markov chain with TA $\mathbf{r}(m)$ (via the conductance [70] of the chain).

By (9.21), we have $r_k(j+1) \le r_k(j) + \alpha(j)\lambda_{max}, \forall k, j$, so $r_k(m) \le \lambda_{max} \sum_{j=0}^{m-1} \alpha(j), \forall k$. Using this in (9.31) yields

$$\begin{aligned} &|E_m[s'_k(m)] - s_k(\mathbf{r}(m))| \\ &\quad \le 2K \cdot f(m)/T_{m+1}, \forall k. \end{aligned} \tag{9.32}$$

(ii) Next, with the Bernoulli arrival process $a_k(t)$ we assumed, it can be shown that

$$|E_m[\lambda'_k(m)] - \lambda_k| \le 1/T_{m+1}. \tag{9.33}$$

Therefore, $|B_k(m)| \le 2K \cdot f(m)/T_{m+1} + 1/T_{m+1} \le 3K \cdot f(m)/T_{m+1}$. Denote by $\mathbf{B}(m)$ the vector of $B_k(m)$'s. Since $|r_k(m) - r_k^*| \le |r_k^*| + |r_k(m)| \le \bar{r} + \lambda_{max} \sum_{j=0}^{m-1} \alpha(j)$ where $\bar{r} = \max_k |r_k^*|$, we have

$$\begin{aligned} & \sum_{m=1}^{\infty} \alpha(m) |(\mathbf{r}(m) - \mathbf{r}^*)^T \cdot \mathbf{B}(m)| \\ \le \quad & 3K^2 \sum_{m=1}^{\infty} \{\alpha(m)[\bar{r} + \lambda_{max}\Delta(m)] \cdot f(m)/T_{m+1}]\} \\ < \quad & \infty \end{aligned} \tag{9.34}$$

where the last step has used condition (9.24).

Step 2: Convergence of $\mathbf{r}(j), j = 1, 2, \ldots$ to $\mathbf{r}^*$

Lemma 10. *If (9.34) and (9.23) hold, then with Algorithm (9.21) whose parameters satisfy the conditions in Theorem 4,* $\mathbf{r}(j)$ *converges to* $\mathbf{r}^*$ *(the optimal solution of problem (9.4)) with probability 1.*

The proof is related to the theory of *stochastic approximation* [57; 58]. We use $d(j) := ||\mathbf{r}(j) - \mathbf{r}^*||_2^2$ as a Lyapunov function and show that $d(j) \to 0$ with probability 1. Ideally, if we have the accurate gradient $\mathbf{g}(j)$ of $F(\mathbf{r}(j); \lambda)$, it is well known that $d(j) \to 0$ with suitable step sizes $\alpha(j)$. The key purpose of (9.34) and (9.23) is to control the effect of the estimation error. Essentially, the result of step 1, (9.34), ensures that the effect of the bias

$\mathbf{B}(j)$ diminishes as $j \to \infty$, and (9.23) ensures that the effect of the martingale noise $\eta(j)$ diminishes (by the martingale convergence theorem.) Combining these and the fact that $\sum_j \alpha(j) = \infty$, the convergence to $\mathbf{r}^*$ can be established.

Since $\mathbf{r}^* = \arg\max_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$, we have $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$.

Step 3: Rate stability

Lemma 11. *With probability 1,* $\lim_{t\to\infty} S_k(t)/t = s_k(\mathbf{r}^*), \forall k$.

This is Lemma 4 in [67]. Since we know from step 2 that $\mathbf{r}(j) \to \mathbf{r}^*$, it seems quite intuitive that in the long term, the average service rate of link k converges to $s_k(\mathbf{r}^*)$. However, the actual proof is not straightforward.

Finally, the following result [67] concludes the proof.

Lemma 12. *If* $\lim_{t\to\infty} S_k(t)/t = s_k(\mathbf{r}^*), \forall k$ *with probability 1, and if* $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$*, then the system is rate stable.*

Proof sketch of Theorem 5

The proof Theorem 5 follows the same line as that of Theorem 4. As the only difference, $\mathbf{r}(j)$ converges to a different $\mathbf{r}^*$ which is the optimal vector of dual variables of problem (9.18)). $\mathbf{r}^*$ satisfies the strict inequality $s_k(\mathbf{r}^*) > \lambda_k, \forall k$, which further guarantees that the queue lengths tend to be small and return to 0 infinitely often.

Proof sketch of Theorem 6

The intuition is that when the step sizes becomes small, $\mathbf{r}(j)$ changes slowly and is quasi-static. Also, $\mathbf{r}(j)$ is bounded, so that the mixing time of the CSMA Markov chain is always bounded. Therefore, in a large time scale, the algorithm follows the gradient direction which leads to the convergence.

Technically, this is proved by using the differential-equation approach in [74] (also used in [46]). That is, the trajectory of $\mathbf{r}(j)$ converges to the solution of a differential equation which converges to $\mathbf{r}^*$. The proof is similar to that of Theorem 9 in the next chapter, and is omitted here.

9.7 Appendices

9.7.1 Proof of the fact that $\mathcal{C}$ is the interior of $\bar{\mathcal{C}}$

Proposition 25. *λ is strictly feasible if and only if $\lambda \in$ int $\bar{\mathcal{C}}$. (In other words, $\mathcal{C} =$ int $\bar{\mathcal{C}}$.)*

Proof. (i) If λ is strictly feasible, then it can be written as $\lambda = \sum_i \bar{p}_i x^i$ where $\bar{p}_i > 0, \forall i$ and $\sum_i \bar{p}_i = 1$. Let $\bar{p}_0$ be the probability corresponding to the all-0 IS, and $\bar{p}_k$ be the probability of the IS $\mathbf{e}_k$, $k = 1, 2, \dots, K$. Let $d_0 = \min\{\bar{p}_0/K, \min_k \bar{p}_k\} > 0$. We claim that for any λ' that satisfies

$$|\lambda'_k - \lambda_k| \le d_0, \forall k, \tag{9.35}$$

we have $\lambda' \in \bar{\mathcal{C}}$. Indeed, if λ' satisfies (9.35), we can find another probability distribution $\bar{\mathbf{p}}'$ such that $\sum_i \bar{p}'_i x^i_k = \lambda'_k, \forall k$. $\bar{\mathbf{p}}'$ can be constructed as follows: let $\bar{p}'_0 = \bar{p}_0 - \sum_k(\lambda'_k - \lambda_k)$, $\bar{p}'_k = \bar{p}_k + (\lambda'_k - \lambda_k)$, and let the probabilites of all other IS's be the same as those in $\bar{\mathbf{p}}$. By condition (9.35), we have $\bar{\mathbf{p}}' \ge \mathbf{0}$. Also, $\sum_i \bar{p}'_i x^i_k = \lambda'_k, \forall k$.

Therefore, $\mathcal{B}(\lambda, d_0) \subseteq \bar{\mathcal{C}}$ where $d_0 > 0$. So $\lambda \in$ int $\bar{\mathcal{C}}$.

(ii) Assume that $\lambda \in$ int $\bar{\mathcal{C}}$. We now construct a $\mathbf{p} > \mathbf{0}$ such that $\lambda = \sum_i p_i x^i$. First, choose an arbitrary $\mathbf{p}_I > \mathbf{0}$ (such that $\sum_i p_{I,i} = 1$) and let $\lambda_I := \sum_i p_{I,i} x^i$. If it happens to be that $\lambda_I = \lambda$, then λ is strictly feasible. In the following we assume that $\lambda_I \ne \lambda$. Since $\lambda \in$ int $\bar{\mathcal{C}}$, there exists a small-enough $d > 0$ such that $\lambda_{II} := \lambda + d \cdot (\lambda - \lambda_I) \in \bar{\mathcal{C}}$. So λ_{II} can be written as $\lambda_{II} = \sum_i p_{II,i} x^i$ where $\mathbf{p}_{II} \ge \mathbf{0}$ and $\sum_i p_{II,i} = 1$.

Notice that $\lambda = \alpha \cdot \lambda_I + (1 - \alpha) \cdot \lambda_{II}$ where $\alpha := d/(1 + d) \in (0, 1)$. So $\lambda = \sum_i p_i x^i$ where $p_i := \alpha \cdot p_{I,i} + (1 - \alpha) \cdot p_{II,i}, \forall i$. Since $\alpha > 0, 1 - \alpha > 0$ and $p_{I,i} > 0, p_{II,i} \ge 0, \forall i$, we have $p_i > 0, \forall i$. Therefore λ is strictly feasible. □

9.7.2 Proof the Proposition 21

Consider the convex optimization problem (9.10), where λ is strictly feasible (i.e., $\lambda = \sum_i \bar{p}_i \cdot x^i$ for some $\bar{p}_i > 0, \forall x^i$ and $\sum_i \bar{p}_i = 1$). Problem (9.10) is clearly feasible and the feasible region is closed and convex. The objective function (the entropy) is bounded in the feasible region. So, the optimal value is bounded.

We now check whether the Slater condition [56] (pages 226-227) is satisfied. Since all the constraints in (9.10) are linear, we only need to check whether there exists a *feasible* $\mathbf{u}$ which is in the relative interior [56] of the domain $\mathcal{D}$ of the objective function $-\sum_i u_i \log(u_i)$, which is $\mathcal{D} = \{\mathbf{u} | u_i \geq 0, \sum_i u_i = 1\}$. Since $\lambda = \sum_i \bar{p}_i \cdot x^i$ where $\bar{p}_i > 0, \forall i$ and $\sum_i \bar{p}_i = 1$, letting $\mathbf{u} = \bar{\mathbf{p}}$ satisfies the requirement. Therefore the Slater condition is satisfied. As a result, there exist finite dual variables $y_k^* \geq 0, w_i^* \geq 0, z^*$ such that the Lagrangian

$$\begin{aligned} & \mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*) \\ = \; & -\textstyle\sum_i u_i \log(u_i) + \sum_k y_k^* (\sum_i u_i \cdot x_k^i - \lambda_k) \\ & + z^* (\textstyle\sum_i u_i - 1) + \sum_i w_i^* u_i \end{aligned} \tag{9.36}$$

is maximized by the optimal solution $\mathbf{u}^*$, and the maximum is attained.

We first claim that the optimal solution satisfies $u_i^* > 0, \forall i$. Suppose $u_i^* = 0$ for all i's in a non-empty set $\mathcal{I}$. For convenience, denote $\bar{\mathbf{p}}$ as the vector of $\bar{p}_i$'s. Since both $\mathbf{u}^*$ and $\bar{\mathbf{p}}$ are feasible for problem (9.10), any point on the line segment between them is also feasible. Then, if we slightly move $\mathbf{u}$ from $\mathbf{u}^*$ along the direction of $\bar{\mathbf{p}} - \mathbf{u}^*$, the change of the objective function $H(\mathbf{u}) := -\sum_i u_i \log(u_i)$ (at $\mathbf{u}^*$) is proportional to

$$\begin{aligned} & (\bar{\mathbf{p}} - \mathbf{u}^*)^T \nabla H(\mathbf{u}^*) \\ = \; & \sum_i (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1] \\ = \; & \sum_{i \notin \mathcal{I}} (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1] + \sum_{i \in \mathcal{I}} \bar{p}_i [-\log(u_i^*) - 1]. \end{aligned}$$

For $i \notin \mathcal{I}$, $u_i^* > 0$, so $\sum_{i \notin \mathcal{I}} (\bar{p}_i - u_i^*)[-\log(u_i^*) - 1]$ is bounded. But for $i \in \mathcal{I}$, $u_i^* = 0$, so that $-\log(u_i^*) - 1 = +\infty$. Also, since $\bar{p}_i > 0$, we have $(\bar{\mathbf{p}} - \mathbf{u}^*)^T \nabla H(\mathbf{u}^*) = +\infty$. This means that $H(\mathbf{u})$ increases when we slightly move $\mathbf{u}$ away from $\mathbf{u}^*$ towards $\bar{\mathbf{p}}$. Thus, $\mathbf{u}^*$ is not the optimal solution.

Therefore $u_i^* > 0, \forall i$. By complementary slackness, $w_i^* = 0$. So the term $\sum_i w_i^* u_i$ in (9.36) is 0. Since $\mathbf{u}^*$ maximizes $\mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*)$, it follows that

$$\frac{\partial \mathcal{L}(\mathbf{u}^*; \mathbf{y}^*, \mathbf{w}^*, z^*)}{\partial u_i} = -\log(u_i^*) - 1 + \sum_k y_k^* x_k^i + z = 0, \forall i.$$

Combining these identities and $\sum_i u_i^* = 1$, we have

$$u_i^* = \frac{\exp(\sum_k y_k^* x_k^i)}{\sum_j \exp(\sum_k y_k^* x_k^j)}, \forall i. \tag{9.37}$$

Plugging (9.37) back into (9.36), we have $\max_{\mathbf{u}} \mathcal{L}(\mathbf{u}; \mathbf{y}^*, \mathbf{w}^*, z^*) = -F(\mathbf{y}^*; \lambda)$. Since $\mathbf{u}^*$ and the dual variables $\mathbf{y}^*$ solves (9.10), $\mathbf{y}^*$ is the solution of $\min_{\mathbf{y} \geq \mathbf{0}}\{-F(\mathbf{y}; \lambda)\}$ (and the optimum is attained). So, $\sup_{\mathbf{r} \geq 0} F(\mathbf{r}; \lambda)$ is attained by $\mathbf{r} = \mathbf{y}^*$. The above proof also shows that (9.4) is the dual problem of (9.10).

9.7.3 Analysis of Algorithm 2

Lemma 13. *Assume that the utility function $v_m(f_m)$ (strictly concave) satisfies $v_m'(0) \leq V < \infty, \forall m$. Denote by L as the largest number of hops of a flow in the network. Then in Algorithm 2, $b_{km}(j) \leq \beta \cdot V + \alpha + 2\alpha \cdot (L-1), \forall k, m$ at all time step j.*

Proof. According to Algorithm 2, the source of flow m solves $f_m(j) = \arg\max_{f_m' \in [0,1]}\{\beta \cdot v_m(f_m') - q_{\delta(m),m}(j) \cdot f_m'\}$. It is easy to see that if $q_{\delta(m),m}(j) \geq \beta \cdot V$, then $f_m(j) = 0$, i.e., the source stops sending data. Thus $q_{\delta(m),m}(j+1) \leq q_{\delta(m),m}(j)$. If $q_{\delta(m),m}(j) < \beta \cdot V$, then $q_{\delta(m),m}(j+1) \leq q_{\delta(m),m}(j) + \alpha < \beta \cdot V + \alpha$. Since initially $q_{km}(0) = 0, \forall k, m$, by induction, we have

$$q_{\delta(m),m}(j) \leq \beta \cdot V + \alpha, \forall j, m. \tag{9.38}$$

Denote $b_{km}(j) := q_{km}(j) - q_{down(k,m),m}(j)$. In Algorithm 2, no matter whether flow m has the maximal back-pressure at link k, the actual average service rate $s_{km}'(j) = 0$ if $b_{km}(j) \leq 0$. That is, $s_{km}'(j) > 0$ only if $b_{km}(j) > 0$. Since $s_{km}'(j) \leq 1$, by item 3 of Algorithm 2, $q_{down(k,m),m}(j+1) \leq q_{down(k,m),m}(j) + \alpha$ and $q_{km}(j+1) \geq q_{km}(j) - \alpha$. Then, if $b_{km}(j) > 0$, we have $b_{km}(j+1) \geq b_{km}(j) - 2\alpha > -2\alpha$. If $b_{km}(j) \leq 0$, then

$b_{km}(j+1) \geq b_{km}(j)$. Since $b_{km}(0)=0$, by induction, we have

$$b_{km}(j) \geq -2\alpha, \forall j, k, m. \tag{9.39}$$

Since $\sum_{k:a_{mk}=1} b_{km}(j) = q_{\delta(m),m}(j)$, combined with (9.38) and (9.39), we have $b_{km}(j) \leq \beta \cdot V + \alpha + 2\alpha \cdot (L-1)$. □

Total utility

Regard each period (with length T) as a "time slot" in [44]. By Lemma 13, $b_{km}(j) \leq \beta \cdot V + \alpha + 2\alpha \cdot (L-1), \forall k, m, j$. Since $r_k(j) = [\max_m b_{km}(j)]_+$, we have $0 \leq r_k(j) \leq C := \beta \cdot V + \alpha + 2\alpha \cdot (L-1)$. Thus, the mixing time of the CSMA Markov chain in any period is bounded [67]. So

$$|E_j[s'_k(j)] - s_k(\mathbf{r}(j))| \leq \frac{C_1}{T} \tag{9.40}$$

where the constant C_1 depends on C and K ([67]), and $E_j(\cdot)$ means the expectation conditioned on the values of all random variables up to time t_j.

Since $u_i^* := p_i(\mathbf{r}(j)), \forall i$ maximizes $H(\mathbf{u}) + \sum_k [r_k(j) \sum_i (x_k^i \cdot u_i)]$ (see Proposition 22), similar to the proof of Proposition 23, we have

$$\begin{aligned} & \sum_k [r_k(j) \sum_i (x_k^i \cdot u_i^*)] \\ = \; & \sum_k [r_k(j) \cdot s_k(\mathbf{r}(j))] \\ \geq \; & \max_{\mu \in \bar{\mathcal{C}}} \sum_k [r_k(j) \cdot \mu_k] - K \cdot \log(2) \end{aligned}$$

where $\bar{\mathcal{C}}$ is the set of feasible service rates (including $\mathcal{C}$ and its boundary).

By this inequality and (9.40),

$$\begin{aligned} \sum_k \{r_k(j) \cdot E_j[s'_k(j)]\} \;\geq\; & \max_{\mu \in \bar{\mathcal{C}}} \sum_k [r_k(j) \cdot \mu_k] \\ & -K \cdot \log(2) - K \cdot C \cdot C_1/T. \end{aligned}$$

Define $\tilde{r}_k(j) := r_k(j)/\alpha$ (then $\tilde{r}_k(j)$ corresponds to the maximal differential backlog

$W_k^*(j)$ in [44], since the change of $r_k(j)$ has been scaled by the step size α), we have

$$\begin{aligned}\sum_k \{\tilde{r}_k(j) \cdot E_j[s'_k(j)]\} \quad &\geq \quad \max_{\mu \in \bar{\mathcal{C}}} \sum_k [\tilde{r}_k(j) \cdot \mu_k] \\ &\quad -[K \cdot \log(2) + K \cdot C \cdot C_1/T]/\alpha.\end{aligned}$$

Now, using Corollary 1 in [44], it follows that

$$\begin{aligned}&\liminf_{J\to\infty} \sum_m v_m(\bar{f}_m(J)) \\ \geq \quad & \bar{W} - \frac{2[K \cdot \log(2) + K \cdot C \cdot C_1/T]/\alpha + 5K}{2\beta/\alpha} \\ = \quad & \bar{W} - \frac{[K \cdot \log(2) + K \cdot C \cdot C_1/T] + 5\alpha \cdot K/2}{\beta} \qquad (9.41)\end{aligned}$$

where $\bar{f}_m(J) := \sum_{j=0}^{J-1} E[f_m(j)]/J$ is the expected average rate of flow m up to the J's period. We have used the fact that $R_k^{max} = 1$, $\mu_{max,k}^{in} = \mu_{max,k}^{out} = 1$, where R_k^{max} is the maximal flow input rate at link k, $\mu_{max,k}^{in}$ and $\mu_{max,k}^{out}$ are the maximal rate the link k can receive or transmit.

As expected, when $T \to \infty$ and $\alpha \to 0$, this bound matches the bound in Proposition 23. Also, as $\beta \to \infty$, $\alpha \to 0$, and $T \to \infty$ in a proper way (since C and C_1 depend on β), $\liminf_{J\to\infty} \sum_m v_m(\bar{f}_m(J)) \to \bar{W}$.

Queue lengths

By (9.38) and (9.39), we have

$$q_{km}(j) \leq \beta \cdot V + \alpha + 2(L-1)\alpha, \forall k, m, j.$$

Also, in view of the dynamics of $q_{km}(j)$ in Algorithm 2, the actual queue lengths $Q_{km}(j) \leq (T/\alpha) \cdot q_{km}(j), \forall k, m, j$. Therefore,

$$Q_{km}(j) \leq \frac{T}{\alpha}[\beta \cdot V + (2L-1)\alpha] \qquad (9.42)$$

So all queue lengths are *uniformly bounded.* The bound increases with T, β and decreases with α.

The above bounds (9.41) and (9.42), however, are not very tight. Our simulation shows near-optimal total utility without a very large β, T or a very small α. This leads to moderate queue lengths.

9.7.4 Extensions: adaptive CSMA scheduling as a modular MAC-layer protocol

Using derivations similar to section 9.3.1, our CSMA algorithm can serve as a modular "MAC-layer scheduling component" in cross-layer optimization, combined with other components in the transport layer and network layer (see Fig. 8.1 for the different layers), usually with queue lengths as the shared information. For example, in addition to its combination with congestion control (at the transport layer), we demonstrate in this section its combination with optimal multipath routing, anycast and multicast (at the network layer). Therefore this is a joint optimization of the transport layer, network layer and the MAC layer.

Anycast

To make the formulation more general, let's consider anycast with multipath routing. (This includes unicast with multipath routing as a special case.) Assume that there are M flows. Each flow m has a source $\delta(m)$ (with some abuse of notation) which generates data and a set of destinations $\mathcal{D}(m)$ which receive the data. "Anycast" means that it is sufficient for the data to reach any node in the set $\mathcal{D}(m)$. However, there is no specific "path" for each flow. The data generated by the source is allowed to split and traverse any link before reaching the destinations (i.e., multipath routing). This allows better utilization of the network resource by routing the data through less congested parts of the network. (For simplicity, we don't consider the possibility of physical-layer multicast here, i.e., the effect that a node's transmission can be received by multiple nodes simultaneously.)

In this case, it is more convenient to use a "node-based" formulation [24; 43]. Denote the number of nodes by J. For each node j, let $\mathcal{I}(j) := \{k|(k,j) \in \mathcal{L}\}$, where $\mathcal{L}$ is the set of links (it is also the set $\mathcal{V}$ in the conflict graph), and let $\mathcal{O}(j) := \{k|(j,k) \in \mathcal{L}\}$. Denote the rate of flow m on link (j,l) by s_{jl}^m. Then the (approximate) utility maximization problem, similar to (9.13), is

$$\begin{array}{ll} \max_{\mathbf{u},\mathbf{s},\mathbf{f}} & -\sum_i u_i \log(u_i) + \beta \cdot \sum_{m=1}^M v_m(f_m) \\ \text{s.t.} & s_{jl}^m \geq 0, \forall (j,l) \in \mathcal{L}, \forall m \\ & f_m + \sum_{l \in \mathcal{I}(j)} s_{lj}^m \leq \sum_{l \in \mathcal{O}(j)} s_{jl}^m, \forall m, j = \delta(m) \\ & \sum_{l \in \mathcal{I}(j)} s_{lj}^m \leq \sum_{l \in \mathcal{O}(j)} s_{jl}^m, \forall m, j \neq \delta(m), j \notin \mathcal{D}(m) \\ & \sum_i u_i \cdot x_{(j,l)}^i \geq \sum_m s_{jl}^m, \forall (j,l) \in \mathcal{L} \\ & u_i \geq 0, \sum_i u_i = 1. \end{array}$$

Associate a dual variable $q_j^m \geq 0$ to the 2nd and 3rd lines of constraints (for each m and $j \notin \mathcal{D}(m)$), and define $q_j^m = 0$ if $j \in \mathcal{D}(m)$. (Note that there is no flow-conservation constraint for flow m at each node in $\mathcal{D}(m)$.) Then similar to section 9.3.1, a partial Lagrangian is

$$\begin{array}{rl} & \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\ = & -\sum_i u_i \log(u_i) \\ & +\beta \cdot \sum_m v_m(f_m) - \sum_m q_{\delta(m)}^m f_m \\ & +\sum_{(j.l) \in \mathcal{L}, m} [s_{jl}^m \cdot (q_j^m - q_l^m)]. \end{array} \tag{9.43}$$

First fix $\mathbf{u}$ and $\mathbf{q}$, consider maximizing $\mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q})$ over $\mathbf{s}$, subject to $s_{jl}^m \geq 0$ and $\sum_i u_i \cdot x_{(j,l)}^i \geq \sum_m s_{jl}^m$. For each link (j,l), let the maximal back-pressure $z_{(j,l)} := \max_m (q_j^m - q_l^m)$. Then clearly, if $z_{(j,l)} > 0$, a flow m' with $q_j^{m'} - q_l^{m'} = z_{(j,l)}$ should be served (with the whole rate $\sum_i u_i \cdot x_{(j,l)}^i$). If $z_{(j,l)} \leq 0$, then no flow is served. After we plug this solution of $\mathbf{s}$ back to (9.43), the rest derivation is the same as in section 9.3.1. Therefore the distributed algorithm is as follows. We again assume $v_m'(0) \leq V < +\infty, \forall m$.

Initially, assume that all queues are empty, and set $q_j^m = 0, \forall j, m$. Then iterate as follows. (Similar to Algorithm 2, the step size is α, and the update interval is T. For simplicity, we omit the time index here.)

- CSMA scheduling and routing: If $z_{(j,l)} > 0$, link (j,l) lets $r_{(j,l)} = z_{(j,l)}$ in the CSMA operation. Choose a flow m' with $q_j^{m'} - q_l^{m'} = z_{(j,l)}$. When it gets the opportunity to transmit, serve flow m'. If $z_{(j,l)} \leq 0$, then link (j,l) keeps silent. (Note that there is no replication of packets.)

- Congestion control: For each flow m, if node j is its source, then it sets $f_m = \arg\max_{f'_m \in [0,1]}\{\beta \cdot v_m(f'_m) - q_j^m f'_m\}$.
- The dual variables q_j^m are updated as follows: $q_j^m \leftarrow [q_j^m - \alpha \sum_{l\in\mathcal{O}(j)} s_{jl}^m)]_+ + \alpha \sum_{l\in\mathcal{I}(j)} s_{lj}^m$ if $j \neq \delta(m)$ and $j \notin \mathcal{D}(m)$; and $q_j^m \leftarrow [q_j^m - \alpha \sum_{l\in\mathcal{O}(j)} s_{jl}^m)]_+ + \alpha(f_m + \sum_{l\in\mathcal{I}(j)} s_{lj}^m)$ if $j = \delta(m)$. (By doing this, roughly $q_j^m \propto Q_j^m$ where Q_j^m is the corresponding queue length.) Always let $q_j^m = 0$ if $j \in \mathcal{D}(m)$.

Furthermore, the above algorithm can be readily extended to incorporate channel selection in *multi-channel wireless networks*, with each "link" defined by a triplet $(j,l;c)$, which refers to the logical link from node j to l on channel c. In this scenario, the conflict graph is defined on the set of links $(j,l;c)$.

Multicast with network coding

Assume that there are M multicast sessions. Each session m has a source $\delta(m)$ which generates data and a set of destinations $\mathcal{D}(m)$ which receive the data. Different from "anycast", here the data must reach all nodes in the set $\mathcal{D}(m)$. There are two possible designs for multicast. (1) Fixed multicast tree, where the routes of each multicast session are fixed. (2) Multicast combined with multipath routing and network coding. Case (1) is straightforward, but the routing may not be optimal. In case (2), [63] demonstrates an algorithm which achieves the optimal utility, which however, requires *centralized* Maximal-Weight scheduling at the MAC layer. In this section, we show that CSMA scheduling can be combined with it, leading to a *fully distributed* algorithm. To facilitate network coding, we let all the packets have the same size (Note that our results are *insensitive* to the distribution of the transmission time, i.e., packet size, if the transmission time and waiting time are not both constant [52]).

According to the theory of network coding [64], a certain flow rate for a multicast session can be supported if and only if it can be supported separately for each destination node.

Let s_{jl}^{mp} be the "information flow rate" on link (j,l) in multicast session m destined for node $p \in \mathcal{D}(m)$, and s_{jl}^{m} be the "capacity" for session m on link (j,l). The above condition is that $s_{jl}^{mp} \le s_{jl}^{m}, \forall p \in \mathcal{D}(m)$. Then, the approximate utility maximization problem is

$$\begin{array}{ll}
\max_{\mathbf{u},\mathbf{s},\mathbf{f}} & H(\mathbf{u}) + \beta \cdot \sum_{m=1}^{M} v_m(f_m) \\
\text{s.t.} & s_{jl}^{mp} \ge 0, \forall (j,l) \in \mathcal{L}, \forall m, \forall p \in \mathcal{D}(m) \\
& f_m + \sum_{l \in \mathcal{I}(j)} s_{lj}^{mp} \le \sum_{l \in \mathcal{O}(j)} s_{jl}^{mp},\ \forall m, j = \delta(m), p \in \mathcal{D}(m) \\
& \sum_{l \in \mathcal{I}(j)} s_{lj}^{mp} \le \sum_{l \in \mathcal{O}(j)} s_{jl}^{mp},\ \forall m, p \in \mathcal{D}(m), j \ne \delta(m), j \ne p \\
& s_{jl}^{mp} \le s_{jl}^{m}, \forall p \in \mathcal{D}(m), \forall (j,l) \in \mathcal{L} \\
& \sum_i u_i \cdot x_{(j,l)}^{i} \ge \sum_m s_{jl}^{m}, \forall (j,l) \in \mathcal{L} \\
& u_i \ge 0, \sum_i u_i = 1.
\end{array}$$

Associate a dual variable $q_j^{mp} \ge 0$ to the 2nd and 3rd lines of constraints (for each $m, p \in \mathcal{D}(m)$ and $j \ne p$), and define $q_j^{mp} = 0$ if $j = p$. Then a partial Lagrangian is

$$\begin{array}{rl}
& \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\
= & H(\mathbf{u}) \\
& +\beta \cdot \sum_m v_m(f_m) - \sum_m (\sum_{p \in \mathcal{D}(m)} q_{\delta(m)}^{mp}) f_m \\
& +\sum_{(j.l) \in \mathcal{L}, m, p \in \mathcal{D}(m)} s_{jl}^{mp} [(q_j^{mp} - q_l^{mp})].
\end{array} \tag{9.44}$$

We first optimize $\mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q})$ over $\{s_{jl}^{mp}\}$, subject to $0 \le s_{jl}^{mp} \le s_{jl}^{m}$. A solution is as follows: $s_{jl}^{mp} = 0, \forall p$ satisfying $q_j^{mp} - q_l^{mp} \le 0$, and $s_{jl}^{mp} = s_{jl}^{m}, \forall p$ satisfying $q_j^{mp} - q_l^{mp} > 0$. Define the "back-pressure" of session m on link (j,l) as $W_{jl}^{m} := \sum_{p \in \mathcal{D}(m)} (q_j^{mp} - q_l^{mp})_+$. By plugging the above solution to (9.44), we have

$$\begin{array}{rl}
& \mathcal{L}(\mathbf{u},\mathbf{s},\mathbf{f};\mathbf{q}) \\
= & H(\mathbf{u}) \\
& +\beta \cdot \sum_m v_m(f_m) - \sum_m (\sum_{p \in \mathcal{D}(m)} q_{\delta(m)}^{mp}) f_m \\
& +\sum_{(j.l) \in \mathcal{L}, m} s_{jl}^{m} W_{jl}^{m}.
\end{array} . \tag{9.45}$$

Now we optimize it over $\{s_{jl}^{m}\}$, subject to $\sum_i u_i \cdot x_{(j,l)}^{i} \ge \sum_m s_{jl}^{m}$. One can find that the rest is similar to previous derivations. To avoid repetition, we directly write down the algorithm. Assume $v_m'(0) \le V < +\infty, \forall m$.

Initially, assume that all queues are empty, and set $q_j^{mp} = 0, \forall j, m, p$. Then iterate:

- CSMA scheduling, routing, and network coding: Link (j,l) computes the maximal back-pressure $z_{(j,l)} := \max_m W_{jl}^m$. If $z_{(j,l)} > 0$, then let $r_{(j,l)} = z_{(j,l)}$ in the CSMA operation. Choose a session m' with $W_{jl}^{m'} = z_{(j,l)}$. When it gets the opportunity to transmit, serve session m'. To do so, node j performs a random linear combination[5] of the head-of-line packets from the queues of session m' with destination $p \in \mathcal{D}(m')$ which satisfies $q_j^{m'p} - q_l^{m'p} > 0$, and transmits the coded packet (similar to [63]). The coded packet, after received by node l, is replicated and put into corresponding queues of session m' at node l (with destination $p \in \mathcal{D}(m')$ such that $q_j^{m'p} - q_l^{m'p} > 0$). The destinations can eventually decode the source packets [63]. If $z_{(j,l)} = 0$, then link (j,l) keeps silent.

- Congestion control: For each flow m, if node j is its source, then it sets $f_m = \arg\max_{f'_m \in [0,1]} \{\beta \cdot v_m(f'_m) - (\sum_{p \in \mathcal{D}(m)} q_{\delta(m)}^{mp}) f'_m\}$.

- The dual variables q_j^m are updated as follows: $q_j^{mp} \leftarrow [q_j^{mp} - \alpha \sum_{l \in \mathcal{O}(j)} s_{jl}^{mp})]_+ + \alpha \sum_{l \in \mathcal{I}(j)} s_{lj}^{mp}$ if $j \neq \delta(m)$ and $j \neq p$ where $p \in \mathcal{D}(m)$; and $q_j^{mp} \leftarrow [q_j^{mp} - \alpha \sum_{l \in \mathcal{O}(j)} s_{jl}^{mp})]_+ + \alpha(f_m + \sum_{l \in \mathcal{I}(j)} s_{lj}^{mp})$ if $j = \delta(m)$. (Note that each packet generated by the source $j = \delta(m)$ is replicated and enters the queues at the source for all destinations of session m.) By doing this, roughly $q_j^{mp} \propto Q_j^{mp}$ where Q_j^{mp} is the corresponding queue length. Always let $q_j^{mp} = 0$ if $j = p$ where $p \in \mathcal{D}(m)$.

Note that both algorithms in section 9.7.4 can be analyzed using the approach in section 9.7.3 for Algorithm 2.

[5]We briefly explain how to perform a "random linear combination" of these packets. For more details, please refer to [63]. (Note that *our main focus* here is to show how to combine CSMA scheduling with other network protocols, instead of network coding itself.) Initially, each packet generated by the source in each session is associated with an ID. Assume that each packet is composed of many "blocks", where each block has γ bits. So, each block can be viewed as a number in a finite field F_{2^γ} which has 2^γ elements. For each packet P to be combined here, randomly choose a coefficient $a_P \in F_{2^\gamma}$. Denote the i'th block of packet P as $P(i)$. Then the corresponding block in the code packet Z is computed as $Z(i) = \sum_P a_P P(i)$, where the multiplication and summation is on the field F_{2^γ}, and the summation is over all the packets to be combined.

Clearly, each packet in the network is a linear combination of some source packets. The ID's of these source packets and the corresponding coefficients are included in the packet header, and are updated after each linear combination along the path (such that the destinations can decode the source packets).

Chapter 10

CSMA Scheduling with Collisions

10.1 Motivation and overview

We have shown that an adaptive CSMA (Carrier Sense Multiple Access) distributed algorithm can achieve the maximal throughput in a general class of networks. In wireless networks, however, the algorithm needs an idealized assumption that the sensing time is negligible, so that there is no collision. In this chapter, we study more practical CSMA-based scheduling algorithms with collisions. First, we provide a discrete-time model of this CSMA protocol and give an explicit throughput formula, which has a simple product-form due to the quasi-reversibility structure of the model. Second, we show that the algorithm in the last chapter can be extended to approach throughput optimality in this case. Finally, sufficient conditions are given to ensure the convergence and stability of the proposed algorithm.

To combine the scheduling algorithm (with collisions) with congestion control, we follow a similar approach used in the last chapter. The details of the combination is given in [71].

To achieve throughput-optimality even with collisions, we need to limit the impact of collisions. Our basic idea, as outlined in section 8.3.1, is to use a protocol similar to the RTS/CTS mode of IEEE 802.11, where we let each link fix its transmission probability but adjust its transmission time (or length) to meet the demand. In the absence of hidden nodes, collisions only occur among the small RTS packets but not the data packets. Also, the collision probability is limited since we fix the transmission probabilities. These two key factors combined ensure a limited impact of collisions. When the transmission lengths are large enough, the protocol intuitively approximates the idealized-CSMA.

However, to precisely model and compute the service rates in the CSMA protocol with collisions, and to prove the throughput-optimality of our algorithms are not straightforward. First, the Markov chain used to model the CSMA protocol is no longer time-reversible. Also, the resulting stationary distribution, although in a product-form, is no longer a Markov Random Field.

Finally, it is worth noting that an interesting byproduct of our general CSMA model developed in this chapter is the unification of several known models for slotted-ALOHA, wireless LAN (as in Bianchi [65]) and the idealized-CSMA model. Indeed, we believe that the general CSMA model captures some essence of random access algorithms.

We note a recent work [61] by Ni and Srikant who developed an alternative algorithm to deal with collisions. Their algorithm uses alternate control phases and data phases. Collisions only occur in the control phase, but not in the data phase. The same product-form distribution as in the last chapter can be obtained for the data phase, which is then used to achieve the maximal throughput.

10.2 CSMA/CA-based scheduling with collisions

10.2.1 Basic protocol

We describe the basic CSMA/CA protocol with fixed transmission probabilities (which suffices for our later development.) Let σ be the duration of each idle slot (or "minislot"). (In 802.11a, for example, $\sigma = 9\mu s$.) In the following we will simply use "slot" to refer to the minislot.

Assume that all links are saturated (i.e., always have packets to transmit). In each slot, if (the transmitter of) link i is not already transmitting and if the medium is idle, the transmitter of link i starts transmission with probability p_i (also denote $q_i := 1-p_i$). If at a certain slot, link i did not choose to transmit but a conflicting link starts transmitting, then link i keeps silent until that transmission ends. If they start transmitting at the same slot, then a collision happens. (In this chapter, we focus on networks without hidden nodes.)

Each link transmits a short probe packet with length γ (all "lengths" here are measured in number of slots) before the data is transmitted (similar to the RTS/CTS mode in 802.11). This increases the overhead of successful transmissions, but can avoid collisions of long data packets. When a collision happens, only the probe packets collide, so each collision lasts a length of γ. Assume that a successful transmission of link i lasts τ_i (which includes a constant overhead τ' and the data payload τ_i^p which is a random variable). Clearly $\tau_i \geq \tau'$. Let the p.m.f. (probability mass function) of τ_i be

$$Pr\{\tau_i = b_i\} = P_i(b_i), \forall b_i \in \mathcal{Z}_{++} \tag{10.1}$$

and assume that the p.m.f. has a *finite support*, i.e., $P_i(b_i) = 0, \forall b_i > b_{max} > 0$. Then the mean of τ_i is $T_i := \sum_{b \in \mathcal{Z}_{++}} b \cdot P_i(b)$.

Fig. 10.1 illustrates the timeline of a 3-link network where link 1 and 2 conflict, and link 2 and 3 conflict.

The above model possesses a quasi-reversibility property that will lead to a simple throughput formula. A process is "time-reversible" if the process and its time-reversed process are statistically indistinguishable [49]. Our model, in Fig. 10.1, reversed in time, follows the same protocol as described above, except for the order of the overhead and the payload, which are reversed. A key reason for this property is that the collisions start and finish at the same time.

10.2.2 Notation

Let the "on-off state" be $x \in \{0,1\}^K$, and x_k be the k-th element of x. Define $x_k = 1$ if link k is active (transmitting) in state x, and $x_k = 0$ otherwise. Then x is a vector indicating which links are active in a given slot. Let $G(x)$ be the subgraph of G after removing all vertices (each representing a link) with state 0 (i.e., any link j with $x_j = 0$) and their associated edges. In general, $G(x)$ is composed of a number of connected components

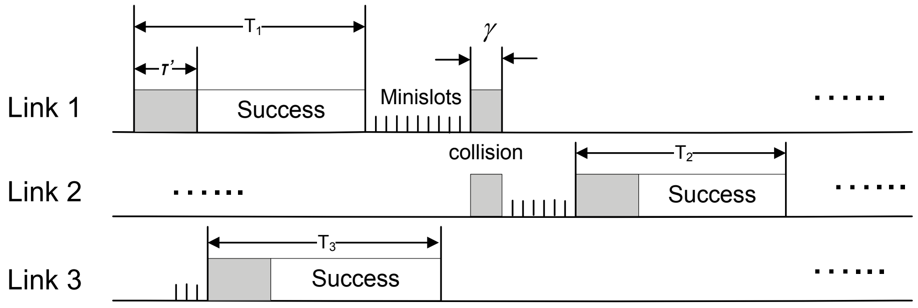


Figure 10.1: Timeline in the basic model (In this figure, $\tau_i = T_i, i = 1, 2, 3$ are constants.)

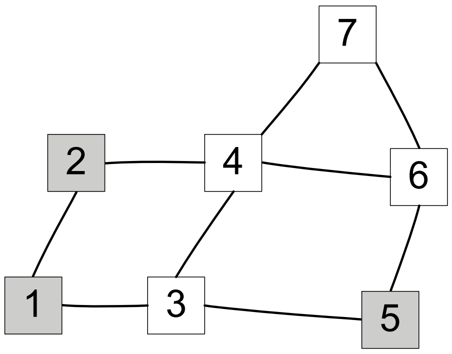


Figure 10.2: An example conflict graph (each square represents a link). In this on-off state x, links 1, 2, 5 are active. So $S(x) = \{5\}$, $\phi(x) = \{1, 2\}$, $h(x) = 1$.

(simply called "components") $C_m(x), m = 1, 2, \dots, M(x)$ (where each component is a set of links, and $M(x)$ is the total number of components in $G(x)$). If a component $C_m(x)$ has only one active link (i.e., $|C_m(x)| = 1$), then this link is having a successful transmission; if $|C_m(x)| > 1$, then all the links in the component are experiencing a collision. Let the set of "success" links in state x be $S(x) := \{k | k \in C_m(x) \text{ with } |C_m(x)| = 1\}$, and the set of links which are experiencing collisions be $\phi(x)$. Also, define the "collision number" $h(x)$ as the number of components in $G(x)$ with size larger than 1. Fig. 10.2 shows an example. Note that the transmissions in a collision component $C_m(x)$ are "synchronized", i.e., the links in $C_m(x)$ must have started transmitting in the same slot, and will end transmitting in the same slot after γ slots (the length of the probe packets).

10.2.3 Throughput computation

By properly defining the "state", the above CSMA/CA protocol defines a discrete time Markov chain. This Markov chain is quasi-reversible, and its stationary distribution has a simple product-form, as shown in section 10.5.1 during the proof of Theorem 7. Then, the probability of any on-off state x can be computed:

Theorem 7. *With the stationary distribution, the probability of $x \in \{0,1\}^K$ is*

$$\begin{aligned} p(x) &= \frac{1}{E}(\gamma^{h(x)} \prod_{k \in S(x)} T_k) \prod_{i:x_i=0} (1-p_i) \prod_{j:x_j=1} p_j \\ &= \frac{1}{E}(\gamma^{h(x)} \prod_{k \in S(x)} T_k) \prod_{i=1}^{K} p_i^{x_i} q_i^{1-x_i} \qquad (10.2) \end{aligned}$$

where T_i is the mean transmission length of link k, and E is a normalizing term such that $\sum_x p(x) = 1$.[1]

The proof is given in section 10.5.1.

Remark: This simple product-form formula turns out to be quite powerful. It not only allows us to extend our algorithms previously designed for idealized CSMA to the practical CSMA with collisions, but also unifies several well-known models — Slotted Aloha, Bianchi's model of Wireless LAN [65] and the idealized CSMA turn out to be special cases of Theorem 7. We will elaborate on this later in section 10.6.

Now we re-parametrize T_k by a variable r_k. Let $T_k := \tau' + T_0 \cdot \exp(r_k)$, where τ' is the overhead of a successful transmission (e.g., RTS, CTS, ACK packets), and $T_k^p := T_0 \cdot \exp(r_k)$ is the mean length of the payload. $T_0 > 0$ is a constant "reference payload length". Let $\mathbf{r}$ be the vector of r_k's. By Theorem 7, the probability of x (with a given $\mathbf{r}$) is

$$p(x;\mathbf{r}) = \frac{1}{E(\mathbf{r})} g(x) \cdot \prod_{k \in S(x)} (\tau' + T_0 \cdot \exp(r_k)) \qquad (10.3)$$

where $g(x) = \gamma^{h(x)} \prod_{i=1}^{K} p_i^{x_i} q_i^{1-x_i}$ is not related to $\mathbf{r}$, and the normalizing term is

$$E(\mathbf{r}) = \sum_{x'} [g(x') \cdot \prod_{k \in S(x')} (\tau' + T_0 \cdot \exp(r_k))]. \qquad (10.4)$$

[1] *In this chapter, several kinds of "states" are defined. With a little abuse of notation, we always use $p(\cdot)$ to denote the probability of the "state" under the stationary distribution of the CSMA Markov chain. This does not cause confusion since the meaning of $p(\cdot)$ is clear from its argument.*

Then, the probability that link k is transmitting payload in a given slot is

$$s_k(\mathbf{r}) = \frac{T_0 \cdot \exp(r_k)}{\tau' + T_0 \cdot \exp(r_k)} \sum_{x:k \in S(x)} p(x; \mathbf{r}) \tag{10.5}$$

Recall that the capacity of each link is 1. So $s_k \in [0, 1]$ is also the *service rate* of link k.

10.3 A Distributed algorithm to approach throughput-optimality

The following theorem states that any $\lambda \in \mathcal{C}$ can be supported by properly choosing the mean payload lengths $T_k^p := T_0 \exp(r_k), \forall k$.

Theorem 8. *Assume that $\gamma, \tau' > 0$, and transmission probabilities $p_k \in (0, 1), \forall k$ are fixed. Given any $\lambda \in \mathcal{C}$, there exists a unique $\mathbf{r}^* \in \mathcal{R}^K$ such that the service rate of link k is equal to the arrival rate for all k:*

$$s_k(\mathbf{r}^*) = \lambda_k, \forall k. \tag{10.6}$$

And $\mathbf{r}^*$ is the solution of the convex optimization problem

$$\max_{\mathbf{r}} L(\mathbf{r}; \lambda) \tag{10.7}$$

where $L(\mathbf{r}; \lambda) = \sum_k (\lambda_k r_k) - \log(E(\mathbf{r}))$. This is because $\partial L(\mathbf{r}; \lambda)/\partial r_k = \lambda_k - s_k(\mathbf{r}), \forall k$.

The proof is given in section 10.5.2.

Theorem 8 motivates us to design a gradient algorithm to solve problem (10.7). However, similar to the idealized-CSMA case in the last chapter, due to the randomness of the system, λ_k and $s_k(\mathbf{r})$ cannot be obtained directly and need to be estimated. In the following algorithm, each link k dynamically adjusts its mean payload length T_k^p based on local information.

Algorithm 4: Transmission length control algorithm

The vectors $\mathbf{r}$ is updated at time t_i, $i = 1, 2, \ldots$. Let $t_0 = 0$ and $t_i - t_{i-1} = M$ (milliseconds), $i = 1, 2, \ldots$. Let "period i" be the time between t_{i-1} and t_i, and $\mathbf{r}(i)$

be the value of $\mathbf{r}$ at the end of period i, i.e., at time t_i. For simplicity, assume that the update interval (M milliseconds) is a multiple of minislots, and $\mathbf{r}(i), \forall i$ is always set at the beginning of a minislot. Initially, link k sets $r_k(0) \in [r_{min}, r_{max}]$ where r_{min}, r_{max} are two parameters (to be further discussed). Then at time t_i, $i = 1, 2, \ldots$, update

$$r_k(i) = r_k(i-1) + \alpha(i)[\lambda'_k(i) - s'_k(i) + h(r_k(i-1))] \tag{10.8}$$

where $\alpha(i) > 0$ is the step size in period i, $\lambda'_k(i), s'_k(i)$ are the empirical average arrival rate and service rate[2] in period i (i.e., the actual amount of arrived traffic and served traffic in period i divided by M. Note that $\lambda'_k(i), s'_k(i)$ are random variables which are generally not equal to λ_k and $s_k(\mathbf{r}(i-1))$. Assume that the maximal instantaneous arrival rate is $\bar{\lambda}$, so $\lambda'_k(i) \le \bar{\lambda}, \forall k, i$. And $h(\cdot)$ is a "penalty function", defined below, to keep $\mathbf{r}(i)$ in a bounded region. (This is a "softer" approach than directly projecting $r_k(i)$ to the set $[r_{min}, r_{max}]$. The purpose is only to simplify the proof of Theorem 9 later.)

$$h(y) = \begin{cases} r_{min} - y & \text{if } y < r_{min} \\ 0 & \text{if } y \in [r_{min}, r_{max}] \\ r_{max} - y & \text{if } y > r_{max} \end{cases} \tag{10.9}$$

In period $i+1$, given $\mathbf{r}(i)$, we need to choose $\tau_k^p(i)$, the payload lengths of each link k, so that $E(\tau_k^p(i)) = T_k^p(i) = T_0 \exp(r_k(i))$. If $T_k^p(i)$ is an integer, then we let $\tau_k^p(i) = T_k^p(i)$; otherwise, we randomize $\tau_k^p(i)$ as follows:

$$\tau_k^p(i) = \begin{cases} \lceil T_k^p(i) \rceil & \text{with probability } T_k^p(i) - \lfloor T_k^p(i) \rfloor \\ \lfloor T_k^p(i) \rfloor & \text{with probability } \lceil T_k^p(i) \rceil - T_k^p(i) \end{cases} \tag{10.10}$$

Clearly, there are other ways to randomize $\tau_k^p(i)$ if $T_k^p(i)$ is not integer.

Intuitively speaking, Algorithm 4 says that when $r_k \in [r_{min}, r_{max}]$, if the empirical arrival rate of link k is larger than the service rate, then link k should transmit more aggressively by using a larger Tx length, and vice versa.

[2]Like in the last chapter, we let link k send dummy packets when the queue is empty (so each link is saturated). This ensures that the CSMA Markov chain has the desired stationary distribution in (10.2). Note that the transmitted dummy packets are also included when $s'_k(i)$ is computed.

Algorithm 4 is parametrized by r_{min}, r_{max} which are fixed during the execution of the algorithm. Note that the choice of r_{max} affects the maximal possible payload length. Also, as discussed below, the choices of r_{max} and r_{min} also determine the "capacity region" of Algorithm 4.

We define the region of arrival rates

$$\mathcal{C}(r_{min}, r_{max}) := \{\lambda | \mathbf{r}^* := \arg\max_{\mathbf{r}} L(\mathbf{r}; \lambda) \in (r_{min}, r_{max})^K\}$$

(recall that $\mathbf{r}^*$ is unique for a given strictly feasible λ). Later we will show that the algorithm can "support" any $\lambda \in \mathcal{C}(r_{min}, r_{max})$ in some sense under certain conditions on the step sizes.

Clearly, $\mathcal{C}([r_{min}, r_{max}]) \to \mathcal{C}$ as $r_{min} \to -\infty$ and $r_{max} \to \infty$, where $\mathcal{C}$ is the set of all strictly feasible λ (by Theorem 8). Therefore, although given r_{min}, r_{max}, the region $\mathcal{C}(r_{min}, r_{max})$ is generally smaller than $\mathcal{C}$, one can choose r_{min}, r_{max} to arbitrarily approach the maximal capacity region $\mathcal{C}$. Also, there is a tradeoff between the capacity region and the maximal packet length.

Theorem 9. *Assume that the vector of arrival rates $\lambda \in \mathcal{C}(r_{min}, r_{max})$. With Algorithm 4,*

(i) If $\alpha(i) > 0$ is non-increasing and satisfies $\sum_i \alpha(i) = \infty$, $\sum_i \alpha(i)^2 < \infty$ and $\alpha(1) \leq 1$, then $\mathbf{r}(i) \to \mathbf{r}^*$ as $i \to \infty$ with probability 1, where $\mathbf{r}^*$ satisfies $s_k(\mathbf{r}^*) = \lambda_k, \forall k$.

(ii) If $\alpha(i) = \alpha$ (i.e., constant step size), then for any $\delta > 0$, there exists $\alpha > 0$ such that $\liminf_{N\to\infty} \sum_{i=1}^{N} s'_k(i)/N] \geq \lambda_k - \delta, \forall k$ with probability 1. In other words, one can achieve average service rates arbitrarily close to the arrival rates by choosing small enough α.

The complete proof is given in section 10.5.3. But the result can be intuitively understood as follows. If the step size is small, r_k is "quasi-static" such that roughly, the service rate is averaged (over multiple periods) to $s_k(\mathbf{r})$, and the arrival rate is averaged to λ_k. Thus the algorithm solves the optimization problem (10.7) by a stochastic approximation [74] argument, such that $\mathbf{r}(i)$ converges to $\mathbf{r}^*$ in part (i), and $r(i)$ is near $\mathbf{r}^*$ with high probability in part (ii).

Corollary 2. *Consider a variant of Algorithm 4:*

$$r_k(i) = r_k(i-1) + \alpha(i)[\lambda'_k(i) + \epsilon - s'_k(i) + h(r_k(i-1))] \tag{10.11}$$

where $\epsilon > 0$. That is, the algorithm "pretends" to serve the arrival rate $\lambda + \epsilon \cdot \mathbf{1}$ (where $\epsilon > 0$) which is slightly larger than the actual λ. Assume that

$$\begin{aligned} \lambda &\in \mathcal{C}'(r_{min}, r_{max}, \epsilon) \\ &:= \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C}(r_{min}, r_{max})\}. \end{aligned}$$

(i) If $\alpha(i) > 0$ is non-increasing and satisfies $\sum_i \alpha(i) = \infty$, $\sum_i \alpha(i)^2 < \infty$ and $\alpha(1) \leq 1$, then $\mathbf{r}(i) \to \mathbf{r}^*$ as $i \to \infty$ with probability 1, where $\mathbf{r}^*$ satisfies $s_k(\mathbf{r}^*) = \lambda_k + \epsilon > \lambda_k, \forall k$

(ii) If $\alpha(i) = \alpha$ (i.e., constant step size) where α is small enough, the all queues are positive recurrent.

Algorithm (10.11) is parametrized by r_{min}, r_{max} and ϵ. Clearly, as $r_{min} \to -\infty$, $r_{max} \to \infty$ and $\epsilon \to 0$, $\mathcal{C}'(r_{min}, r_{max}, \epsilon) \to \mathcal{C}$, the maximal capacity region.

The proof is similar to that of Theorem 9 and is given in [73]. A sketch is as follows: Part (i) is similar to (i) in Theorem 9. The extra fact that $s_k(\mathbf{r}^*) > \lambda_k, \forall k$ reduces the queue size compared to Algorithm 4. Part (ii) holds because if we choose $\delta = \epsilon/2$, then by Theorem 9, $\liminf_{N\to\infty} \sum_{i=1}^{N} s'_k(i)/N] \geq \lambda_k + \epsilon - \delta > \lambda_k, \forall k$ almost surely if α is small enough. Then the result follows by showing that the queues have negative drift.

10.4 Numerical examples

Consider the conflict graph in Fig. 10.3. Let the vector of arrival rates be $\lambda = \rho \cdot \hat{\lambda}$, where $\rho \in (0, 1)$ is the "load", and $\hat{\lambda}$ is a convex combination of several maximal IS: $\hat{\lambda} = 0.2*[1, 0, 1, 0, 1, 0, 0] + 0.2*[0, 1, 0, 0, 1, 0, 1] + 0.2*[0, 0, 0, 1, 0, 1, 0] + 0.2*[0, 1, 0, 0, 0, 1, 0] + 0.2*[1, 0, 1, 0, 0, 1, 0] = [0.4, 0.4, 0.4, 0.2, 0.4, 0.6, 0.2]$. Since $\rho \in (0, 1)$, λ is strictly feasible. Fix the Tx probabilities at $p_k = 1/16, \forall k$. The "reference payload length" $T_0 = 15$. The collision length (e.g., RTS length) is $\gamma = \eta \cdot 10$, and the overhead of successful transmission

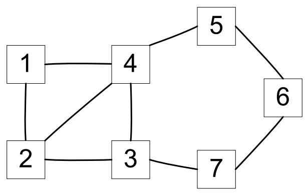
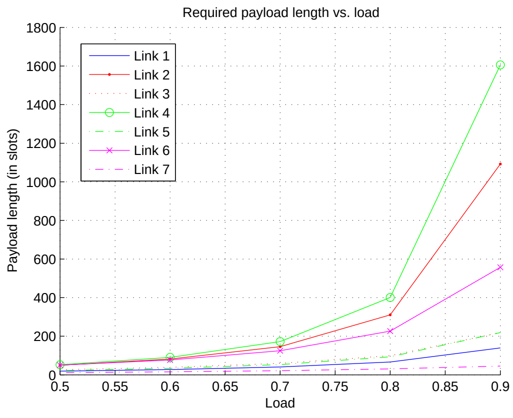
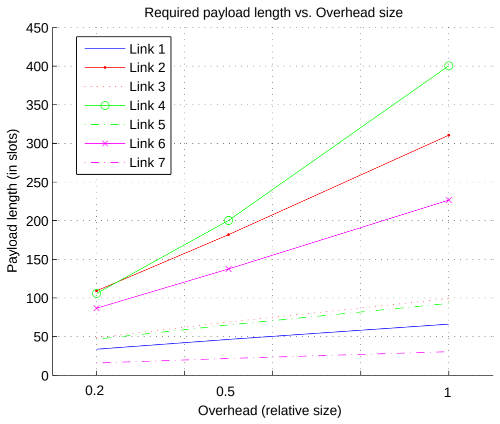


Figure 10.3: The conflict graph in simulations



(a) Relation with the load (given $\eta = 1$)

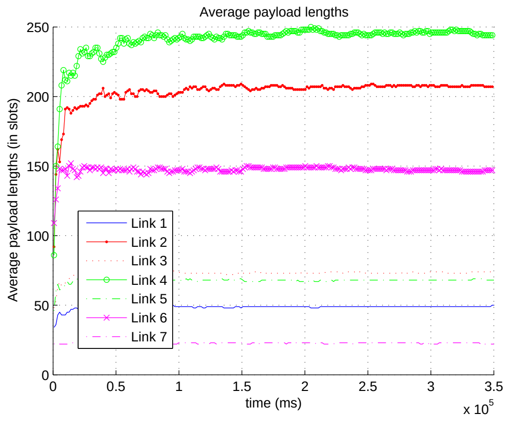


(b) Relation with the overhead (given $\rho = 0.8$)

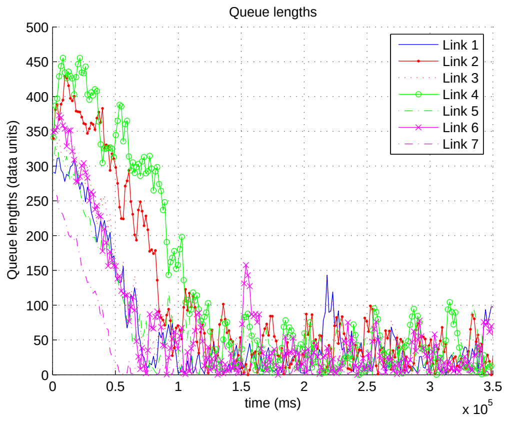
Figure 10.4: Required mean payload lengths

is $\tau' = \eta \cdot 20$, where η is a "relative size" of the overhead for simulation purpose. Later we will let $\eta \in \{1, 0.5, 0.2\}$ to illustrate the effects of overhead size.

Now we vary ρ and η. And in each case we solve problem (10.7) to obtain the required mean payload length $T_k^p := T_0 \cdot \exp(r_k^*), k = 1, 2, \ldots, 7$. Fig. 10.4 (a) shows how T_k^p's change as the load ρ changes, where $\eta = 1$. Clearly, as ρ increases, T_k^p's tend to increase. Also, the rate of increase becomes faster as ρ approaches 1. Therefore, there is a tradeoff between the throughput and transmission lengths (long transmission lengths introduce larger delays for conflicting links). Fig. 10.4 (b) shows how T_k^p's depends on the relative size η of overhead (with fixed $\rho = 0.8$ and $\eta \in \{1, 0.5, 0.2\}$). As expected, the smaller the overhead, the smaller T_k^p's are required.



(a) Convergence of the mean payload lengths



(b) Stability of the queues

Figure 10.5: Simulation of Algorithm (10.11) (with the conflict graph in Fig. 10.3)

Next, we evaluate algorithm (10.11) (a variant of Algorithm 4) in our C++ simulator. The update in (10.11) is performed every $M = 5ms$. Let the step size $\alpha(i) = 0.23/(2 + i/100)$. The upper bound $r_{max} = 5$, lower bound $r_{min} = 0$, and the "gap" $\epsilon = 0.005$. Assume the initial values of r_k's are 0.

Let the "load" of arrival rates be $\rho = 0.8$ (i.e., $\lambda = 0.8 \cdot \hat{\lambda}$), and the relative size of overhead $\eta = 0.5$ (i.e., $\gamma = 5, \tau' = 10$). To show the negative drift of the queue lengths, assume that initially all queue lengths are 300 (data units). As expected, Fig. 10.5 (a) shows the convergence of the mean payload lengths, and Fig. 10.5 (b) shows that all queues are stable.

10.5 Proofs of theorems

10.5.1 Proof of Theorem 7

Define the state

$$w := \{x, ((b_k, a_k), \forall k : x_k = 1)\} \tag{10.12}$$

where b_k is the total length of the current packet link k is transmitting, a_k is the remaining time (including the current slot) before the transmission of link k ends. Note that

(I) $a_k \le b_k, \forall k$.

(II) If $k \in \phi(x)$, then $b_k = \gamma$ and $a_k \in \{1, 2, \dots, \gamma\}$. An important observation here is that the transmissions in a collision component $C_m(x)$ are "synchronized", i.e., the links in $C_m(x)$ must have started transmitting at the same time, and will end transmitting at the same time, so all links in the component $C_m(x)$ have the same remaining time. That is, $a_k = a^{(m)}$ for any $k \in C_m(x)$ where $|C_m(x)| > 1$, and $a^{(m)}$ denotes the remaining time of the component $C_m(x)$. (To see this, any two links i and j in this component with an edge between them must have started transmitting at the same time. Otherwise, if i starts earlier, j would not transmit since it already hears i's transmission; and vice versa. By induction, all links in the component must have started transmitting at the same time.)

The transitions among the set of states defined in (10.12) form a discrete-time Markov chain which is ergodic. Since the transmission lengths are always bounded by b_{max} by assumption, we have $b_k \le b_{max}$, and therefore the Markov chain has a finite number of states. Its stationary distribution is expressed in the following lemma.

Lemma 14. *In the stationary distribution, the probability of a valid*[3] *state w as defined by (10.12) is*

$$p(w) = \frac{1}{E} \prod_{i:x_i=0} q_i \prod_{j:x_j=1} [p_j \cdot f(b_j, j, x)] \tag{10.13}$$

where

$$f(b_j, j, x) = \begin{cases} 1 & \text{if } j \in \phi(x) \\ P_j(b_j) & \text{if } j \in S(x) \end{cases}, \tag{10.14}$$

where $P_j(b_j)$ is the p.m.f. of link j's transmission length, as defined in (10.1). Also, E is a normalizing term such that $\sum_w p(w) = 1$, i.e., all probabilities sum up to 1. Note that $p(w)$ does not depend on the remaining time a_k's.

Proof. For a given state $w = \{x, ((b_k, a_k), \forall k : x_k = 1)\}$, define the set of active links whose

[3] *A state w is valid iff it satisfies (I) and (II) above.*

remaining time is larger than 1 as

$$A_1(w) = \{k | x_k = 1, a_k > 1\}.$$

Links in $A_1(w)$ will continue their transmissions (either with success or a collision) in the next slot.

Define the set of inactive links "blocked" by links in $A_1(w)$ as

$$\partial A_1(w) = \{j | e(j,k) = 1 \text{ for some } k \in A_1(w)\}$$

where $e(j,k) = 1$ means that there is an edge between j and k in the conflict graph. Links in $\partial A_1(w)$ will remain inactive in the next slot.

Write $\bar{A}_1(w) := A_1(w) \cup \partial A_1(w)$. Define the set of all other links as

$$A_2(w) = \mathcal{N} \backslash \bar{A}_1(w).$$

These links can change their on-off states x_k's in the next slot. On the other hand, links in $\bar{A}_1(w)$ will have the same on-off states x_k's in the next slot.

State w can transit in the next slot to another valid state $w' = \{x', ((b'_k, a'_k), \forall k : x'_k = 1)\}$, i.e., $Q(w, w') > 0$, if and only if w' satisfies that (i) $x'_k = x_k, \forall k \in \bar{A}_1(w)$; (ii) $b'_k = b_k, a'_k = a_k - 1, \forall k \in \bar{A}_1(w)$ such that $x_k = 1$; (iii) $a'_k = b'_k, \forall k \in A_2(w)$ such that $x'_k = 1$, and $b'_k = \gamma, \forall k \in A_2(w) \cap \phi(x')$. (If $A_2(w)$ is an empty set, then condition (iii) is trivially true.) The transition probability is

$$Q(w, w') = \prod_{i \in A_2(w)} [p_i \cdot f(b'_i, i, x')]^{x'_i} q_i^{1-x'_i}.$$

Define

$$\tilde{Q}(w', w) := \prod_{i \in A_2(w)} [p_i \cdot f(b_i, i, x)]^{x_i} q_i^{1-x_i}.$$

(If $A_2(w)$ is an empty set, then $Q(w, w') = 1$ and $\tilde{Q}(w', w) := 1$.) If w and w' does not satisfy conditions (i), (ii), (iii), then $Q(w, w') = 0$, and also define $\tilde{Q}(w', w) = 0$. ($\tilde{Q}(w', w)$ can be viewed as the transition probability of the time-reversed process: notice the similarity between $Q(w, w')$ and $\tilde{Q}(w', w)$.)

Then, if $Q(w,w') > 0$ (and $\tilde{Q}(w',w) > 0$), $p(w)/\tilde{Q}(w',w) = \frac{1}{E}\prod_{i \notin A_2(w)}[p_i \cdot f(b_i,i,x)]^{x_i} q_i^{1-x_i}$. And $p(w')/Q(w,w') = \frac{1}{E}\prod_{i \notin A_2(w)}[p_i \cdot f(b'_i,i,x')]^{x'_i} q_i^{1-x'_i}$. But for any $i \notin A_2(w)$, i.e., $i \in \bar{A}_1(w)$, we have $x'_i = x_i, b'_i = b_i$ by condition (i), (ii) above. Therefore, the two expressions are equal. Thus

$$p(w)Q(w,w') = p(w')\tilde{Q}(w',w), \forall w, w'.$$

Therefore,

$$\sum_w p(w) \cdot Q(w,w') = \sum_w p(w') \cdot \tilde{Q}(w',w) = p(w').$$

That is, the distribution (10.13) is invariant (or "stationary"). □

Using Lemma 14, the probability of any on-off state x, as in Theorem 7, can be computed by summing up the probabilities of all states w's with the same on-off state x, using (10.13).

Define the set of valid states $\mathcal{B}(x) := \{w|$ the on-off state is x in the state $w\}$. By Lemma 14, we have

$$\begin{aligned} p(x) &= \sum_{w \in \mathcal{B}(x)} p(w) \\ &= \frac{1}{E}\sum_{w \in \mathcal{B}(x)} \{\prod_{i:x_i=0} q_i \prod_{j:x_j=1} [p_j \cdot f(b_j,j,x)]\} \\ &= \frac{1}{E}(\prod_{i:x_i=0} q_i \prod_{j:x_j=1} p_j) \sum_{w \in \mathcal{B}(x)} \prod_{j:x_j=1} f(b_j,j,x) \\ &= \frac{1}{E}(\prod_{i:x_i=0} q_i \prod_{j:x_j=1} p_j) \cdot \sum_{w \in \mathcal{B}(x)} [\prod_{j \in S(x)} P_j(b_j)] \end{aligned} \tag{10.15}$$

Now we compute the term $\sum_{w \in \mathcal{B}(x)}[\prod_{j \in S(x)} P_j(b_j)]$. Consider a state $w = \{x, ((b_k,a_k), \forall k : x_k = 1)\} \in \mathcal{B}(x)$. For $k \in S(x)$, b_k can be different values in $\mathcal{Z}_{++}$. For each fixed b_k, a_k can be any integer from 1 to b_k. For a collision component $C_m(x)$ (i.e., $|C_m(x)| > 1$), the remaining time of each link in the component, $a^{(m)}$, can be any integer from 1 to γ. Then we have

$$\begin{aligned}
&\sum_{w\in\mathcal{B}(x)}[\prod_{j\in S(x)} P_j(b_j)] \\
=\ & \prod_{j\in S(x)}[\sum_{b_j}\sum_{1\le a_j\le b_j} P_j(b_j)] \prod_{m:|C_m(x)|>1}(\sum_{1\le a^{(m)}\le\gamma} 1) \\
=\ & \prod_{j\in S(x)}[\sum_{b_j} b_j P_j(b_j)]\cdot\gamma^{h(x)} \\
=\ & (\prod_{j\in S(x)} T_j)\gamma^{h(x)} \qquad (10.16)
\end{aligned}$$

Combining (10.15) and (10.16) completes the proof.

10.5.2 Proof of Theorem 8

If at an on-off state x, $k\in S(x)$ (i.e., k is transmitting successfully), it is possible that link k is transmitting the overhead or the payload. So we define a more detailed state (x,z), where $z\in\{0,1\}^K$. Let $z_k=1$ if $k\in S(x)$ and link k is transmitting its payload (instead of overhead). Let $z_k=0$ otherwise. Denote the set of all possible detailed state (x,z) by $\mathcal{S}$.

Then similar to the proof of Theorem 7, and using equation (10.3), we have the following product-form stationary distribution

$$p((x,z);\mathbf{r})=\frac{1}{E(\mathbf{r})}g(x,z)\cdot\exp(\sum_k z_k r_k) \qquad (10.17)$$

where

$$g(x,z)=g(x)\cdot(\tau')^{|S(x)|-\mathbf{1'z}}T_0^{\mathbf{1'z}}. \qquad (10.18)$$

where $\mathbf{1'z}$ is the number of links that are transmitting the payload in state (x,z).

Now we give alternative definitions of feasible and strictly feasible arrival rates. As will be shown soon, these definitions are equivalent to Definition 1 in section 8.1.

Definition 5. *(i) A vector of arrival rate $\lambda\in\mathcal{R}_+^K$ (where K is the number of links) is* feasible *if there exists a probability distribution $\bar{\mathbf{p}}$ over $\mathcal{S}$ (i.e., $\sum_{(x,z)\in\mathcal{S}}\bar{p}((x,z))=1$ and $\bar{p}((x,z))\ge 0$), such that*

$$\lambda_k=\sum_{(x,z)\in\mathcal{S}}\bar{p}((x,z))\cdot z_k. \qquad (10.19)$$

Let $\bar{\mathcal{C}}_{CO}$ be the set of feasible λ, where "CO" stands for "collision".

The rationale of the definition is that if λ can be scheduled by the network, the fraction of time that the network spent in the detailed states must be non-negative and sum up to 1. (Note that (10.19) is the probability that link k is sending its payload given the distribution of the detailed states.)

For example, in the Ad-hoc network in Fig. 8.2, $\lambda = (0.5, 0.5, 0.5)$ is feasible, because (10.19) holds if we let the probability of the detailed state $(x = (1,0,1), z = (1,0,1))$ be 0.5, the probability of the detailed state $(x = (0,1,0), z = (0,1,0))$ be 0.5, and all other detailed states have probability 0.

(ii) A vector of arrival rate $\lambda \in \mathcal{R}_+^K$ is strictly feasible *if it can be written as (10.19) where $\sum_{(x,z)\in\mathcal{S}} \bar{p}((x,z)) = 1$ and $\bar{p}((x,z)) > 0$. Let $\mathcal{C}_{CO}$ be the set of strictly feasible λ. We will show below that $\mathcal{C}_{CO}$ is the* interior *of $\bar{\mathcal{C}}_{CO}$.*

In the previous example, $\lambda = (0.5, 0.5, 0.5)$ is not strictly feasible since it cannot be written as (10.19) where all $\bar{p}((x,z)) > 0$. But $\lambda' = (0.49, 0.49, 0.49)$ is strictly feasible.

Proposition 26. ***Definition** 1 **and Definition 5 are equivalent.*** *That is,*

$$\bar{\mathcal{C}}_{CO} = \bar{\mathcal{C}} \tag{10.20}$$

$$\mathcal{C}_{CO} = \mathcal{C} \tag{10.21}$$

where $\bar{\mathcal{C}}$ and $\mathcal{C}$ are the feasible and strictly feasible sets of λ as defined in section 8.1 (without states with collisions).

Remark: This also implies that $\mathcal{C}_{CO}$ is the interior of $\bar{\mathcal{C}}_{CO}$, since $\mathcal{C}$ is the interior of $\bar{\mathcal{C}}$ by Prop. 19.

Proof. We first prove (10.20). By definition, any $\lambda \in \bar{\mathcal{C}}$ can be written as $\lambda = \sum_{\sigma\in\mathcal{X}} \bar{p}_\sigma \sigma$ where $\mathcal{X}$ is the set of independent sets, and $\bar{\mathbf{p}} = (\bar{p}_\sigma)_{\sigma\in\mathcal{X}}$ is a probability distribution, i.e., $\bar{p}_\sigma \geq 0, \sum_{\sigma\in\mathcal{X}} \bar{p}_\sigma = 1$. Now we construct a distribution $\mathbf{p}$ over the states $(x,z) \in \mathcal{S}$ as follows. Let $p((\sigma,\sigma)) = \bar{p}_\sigma, \forall \sigma \in \mathcal{X}$, and let $p((x,z)) = 0$ for all other states $(x,z) \in \mathcal{S}$.

Then, clearly $\sum_{(x,z)\in\mathcal{S}} p((x,z)) \cdot z = \sum_{\sigma\in\mathcal{X}} p((\sigma,\sigma)) \cdot \sigma = \sum_{\sigma\in\mathcal{X}} \bar{p}_\sigma \sigma = \lambda$, which implies that $\lambda \in \bar{\mathcal{C}}_{CO}$. So,

$$\bar{\mathcal{C}} \subseteq \bar{\mathcal{C}}_{CO}. \tag{10.22}$$

On the other hand, if $\lambda \in \bar{\mathcal{C}}_{CO}$, then $\lambda = \sum_{(x,z)\in\mathcal{S}} p((x,z)) \cdot z$ for some distribution $\mathbf{p}$ over $\mathcal{S}$. We define another distribution $\bar{\mathbf{p}}$ over $\mathcal{X}$ as follows. Let $\bar{p}_\sigma = \sum_{(x,z)\in\mathcal{S}:z=\sigma} p((x,z)), \forall \sigma \in \mathcal{X}$. Then, $\lambda = \sum_{(x,z)\in\mathcal{S}} p((x,z)) \cdot z = \sum_{\sigma\in\mathcal{X}} \sum_{(x,z)\in\mathcal{S}:z=\sigma} p((x,z))\sigma = \sum_{\sigma\in\mathcal{X}} \bar{p}_\sigma \sigma$, which implies that $\lambda \in \bar{\mathcal{C}}$. Therefore

$$\bar{\mathcal{C}}_{CO} \subseteq \bar{\mathcal{C}}. \tag{10.23}$$

Combining (10.22) and (10.23) yields (10.20).

To prove (10.21), we only need to show that $\mathcal{C}_{CO}$ is the interior of $\bar{\mathcal{C}}$. The proof is very similar to that in section 9.7.1, and is thus omitted. □

We now consider the following function (the "log likelihood function" [59] if we estimate the parameter $\mathbf{r}$ from $\bar{p}((x,z))$'s). We will show that $\mathbf{r}^*$ that maximizes $F(\mathbf{r};\lambda)$ over $\mathbf{r}$ satisfies (10.6).

$$\begin{aligned} & F(\mathbf{r};\lambda) \\ = & \textstyle\sum_{(x,z)\in\mathcal{S}} \bar{p}((x,z)) \log(p((x,z);\mathbf{r})) \\ = & \textstyle\sum_{(x,z)\in\mathcal{S}} \bar{p}((x,z))[\sum_k z_k r_k + \log(g(x,z)) - \log(E(\mathbf{r}))] \\ = & \textstyle\sum_k \lambda_k r_k + \sum_{(x,z)\in\mathcal{S}} [\bar{p}((x,z)) \log(g(x,z))] - \log(E(\mathbf{r})) \end{aligned}$$

where $\lambda_k = \sum_{(x,z)\in\mathcal{S}} \bar{p}((x,z)) \cdot z_k$ is the arrival rate at link k. Note that $F(\mathbf{r};\lambda)$ is concave in $\mathbf{r}$. This is because (a) the first term is linear in $\mathbf{r}$; (b) the second term does not involve $\mathbf{r}$ and (c) $E(r)$ as defined in (10.4) can be expanded to a sum of exponential terms of $\mathbf{r}$. So $\log(E(\mathbf{r}))$ is a log-sum-exp function which is convex [56]. Therefore $F(\mathbf{r};\lambda)$ is concave in $\mathbf{r}$.

Consider the following optimization problem

$$\sup_{\mathbf{r}} F(\mathbf{r};\lambda) \ \cdot \tag{10.24}$$

Note that the difference between $F(\mathbf{r};\lambda)$ and $L(\mathbf{r};\lambda)$ defined in Theorem 8 is a constant independent of $\mathbf{r}$. Therefore, problem (10.24) is equivalent to

$$\sup_{\mathbf{r}} L(\mathbf{r};\lambda). \tag{10.25}$$

Since $\log(p((x,z);\mathbf{r})) \le 0$, we have $F(\mathbf{r};\lambda) \le 0$. Therefore $\sup_{\mathbf{r}} F(\mathbf{r};\lambda)$ exists. Since λ is strictly feasible, $\sup_{\mathbf{r}} F(\mathbf{r};\lambda)$ can be attained (The proof of this subtle point is given a little later.) So the problem is the same as $\max_{\mathbf{r}} F(\mathbf{r};\lambda)$. Hence, the solution of (10.24), $\mathbf{r}^*$, satisfies

$$\begin{aligned}
&\frac{\partial F(\mathbf{r}^*;\lambda)}{\partial r_k} \\
= \;& \lambda_k - \frac{1}{E(\mathbf{r}^*)} \sum_{x:k\in S(x)} [g(x)\cdot T_0 \cdot \exp(r_k^*) \cdot \prod_{j\in S(x), j\ne k} (\tau' + T_0 \cdot \exp(r_j^*))] \\
= \;& \lambda_k - \frac{1}{E(\mathbf{r}^*)} \frac{T_0 \cdot \exp(r_k^*)}{\tau' + T_0\cdot \exp(r_k^*)} \sum_{x:k\in S(x)} [g(x)\cdot \prod_{j\in S(x)} (\tau' + T_0\cdot \exp(r_j^*))] \\
= \;& \lambda_k - \frac{T_0 \cdot \exp(r_k^*)}{\tau' + T_0\cdot \exp(r_k^*)} \sum_{x:k\in S(x)} p(x;\mathbf{r}^*) \\
= \;& \lambda_k - s_k(\mathbf{r}^*) = 0.
\end{aligned}$$

Proof of the attainability of $\sup_{\mathbf{r}} F(\mathbf{r};\lambda)$

Lemma 15. *Assume that λ is strictly feasible. Problem (10.24) is the dual problem of the following convex optimization problem, where the vector $\mathbf{u}$ can be viewed as a probability distribution over the detailed states (x,z):*

$$\begin{aligned}
\max_{\mathbf{u}} \quad & \{ \sum_{(x,z)\in\mathcal{S}} [-u_{(x,z)} \log(u_{(x,z)})] + \sum_{(x,z)\in\mathcal{S}} [u_{(x,z)} \cdot \log(g(x,z))] \} \\
s.t. \quad & \sum_{(x,z)\in\mathcal{S}: z_k=1} u_{(x,z)} \ge \lambda_k, \forall k \\
& u_{(x,z)} \ge 0, \sum_{(x,z)} u_{(x,z)} = 1.
\end{aligned} \tag{10.26}$$

Furthermore, $\sup_{\mathbf{r}} F(\mathbf{r};\lambda)$ *is attainable.*

Proof. Since λ is strictly feasible, problem (10.26) is strictly feasible and satisfies the Slater condition [56].

Let $y_k \geq 0$ be the dual variable associated with the constraint $\sum_{(x,z)\in\mathcal{S}:z_k=1} u_{(x,z)} \geq \lambda_k$, then a partial Lagrangian is

$$\begin{aligned}
& \mathcal{L}(\mathbf{u};\mathbf{y}) \\
= & \sum_{(x,z)\in\mathcal{S}} [-u_{(x,z)} \log(u_{(x,z)})] + \sum_{(x,z)\in\mathcal{S}} [u_{(x,z)} \cdot \log(g(x,z))] \\
& + \sum_k y_k [\sum_{(x,z)\in\mathcal{S}:z_k=1} u_{(x,z)} - \lambda_k] \\
= & \sum_{(x,z)\in\mathcal{S}} \{u_{(x,z)}[-\log(u_{(x,z)}) + \log(g(x,z)) + \sum_{k:z_k=1} y_k]\} - \sum_k (y_k\lambda_k).
\end{aligned}$$

So

$$\frac{\partial\mathcal{L}(\mathbf{u};\mathbf{y})}{\partial u_{(x,z)}} = -\log(u_{(x,z)}) - 1 + \log(g(x,z)) + \sum_{k:z_k=1} y_k.$$

If $u_{(x,z)} = p((x,z);\mathbf{y})$ (cf. equation (10.17)), then the partial derivative

$$\frac{\partial\mathcal{L}(\mathbf{u};\mathbf{y})}{\partial u_{(x,z)}} = \log(E(\mathbf{y})) - 1$$

which is the same for all state (x,z) (Given the dual variables $\mathbf{y}$, $\log(E(\mathbf{y}))$ is a constant). This means that $u_{(x,z)} = p((x,z);\mathbf{y}) > 0$ maximizes $\mathcal{L}(\mathbf{u};\mathbf{y})$ over $\mathbf{u}$ subject to $u_{(x,z)} \geq 0, \sum_{(x,z)} u_{(x,z)} = 1$ (since it is impossible to increase $\mathcal{L}(\mathbf{u};\mathbf{y})$ over $\mathbf{u}$ by slightly perturbing $\mathbf{u}$). Denote $l(\mathbf{y}) = \max_{\mathbf{u}} \mathcal{L}(\mathbf{u};\mathbf{y})$, and $u_{(x,z)}(y) = p((x,z);\mathbf{y})$ as the maximizer. Then the dual problem of (10.26) is $\inf_{\mathbf{y}\geq 0} l(\mathbf{y})$. Plugging the expression of $u_{(x,z)}(y)$ into $\mathcal{L}(\mathbf{u};\mathbf{y})$, it is not difficult to find that $\inf_{\mathbf{y}\geq 0} l(\mathbf{y})$ is equivalent to problem (10.24), with $\mathbf{r}$ and $\mathbf{y}$ interchangeable.

Since problem (10.26) satisfies the Slater condition, there exists a (finite) vector of optimal dual variable $\mathbf{r}^*$, which is also the solution of problem (10.24). Therefore, $\sup_{\mathbf{r}} F(\mathbf{r};\lambda)$ is attainable and is the same as $\max_{\mathbf{r}} F(\mathbf{r};\lambda)$. $\square$

Proof of the uniqueness of $\mathbf{r}^*$

Now we show the uniqueness of $\mathbf{r}^*$. Note that the objective function of (10.26) is strictly concave. Therefore $\mathbf{u}^*$, the optimal solution of (10.26) is unique. Consider two extended state $(\mathbf{e}_k, \mathbf{e}_k)$ and $(\mathbf{e}_k, \mathbf{0})$, where $\mathbf{e}_k$ is the K-dimensional vector whose k'th element is 1 and all other elements are 0's. We have $u^*_{(\mathbf{e}_k,\mathbf{e}_k)} = p((\mathbf{e}_k,\mathbf{e}_k);\mathbf{r}^*)$ and $u^*_{(\mathbf{e}_k,\mathbf{0})} = p((\mathbf{e}_k,\mathbf{0});\mathbf{r}^*)$. Then by (10.17),

$$u_{(\mathbf{e}_k,\mathbf{e}_k)}(\mathbf{r}^*)/u_{(\mathbf{e}_k,\mathbf{0})}(\mathbf{r}^*) = \exp(r^*_k) \cdot (T_0/\tau'). \tag{10.27}$$

Suppose that r^* is not unique, that is, there exist $\mathbf{r}^*_I \neq \mathbf{r}^*_{II}$ but both are optimal $\mathbf{r}$. Then, $r^*_{I,k} \neq r^*_{II,k}$ for some k. This contradict to (10.27) and the uniqueness of $\mathbf{u}^*$. Therefore $\mathbf{r}^*$ is unique.

10.5.3 Proof of Theorem 9

We will use results in [74] to prove Theorem 9. Similar techniques have been used in [46] to analyze the convergence of an algorithm in [26].

Part (i): Decreasing step size

Define the concave function

$$H(y) := \begin{cases} -(r_{min} - y)^2/2 & \text{if } y < r_{min} \\ 0 & \text{if } y \in [r_{min}, r_{max}] \\ -(r_{max} - y)^2/2 & \text{if } y > r_{max} \end{cases} \tag{10.28}$$

Note that $dH(y)/dy = h(y)$ where $h(y)$ is defined in (10.9). Let $G(\mathbf{r};\lambda) = F(\mathbf{r};\lambda) + \sum_k H(r_k)$. Since λ is strictly feasible, $\max_{\mathbf{r}} F(\mathbf{r};\lambda)$ has a unique solution $\mathbf{r}^*$. That is, $F(\mathbf{r}^*;\lambda) > F(\mathbf{r};\lambda), \forall \mathbf{r} \neq \mathbf{r}^*$. Since $\mathbf{r}^* \in (r_{min}, r_{max})^K$ by assumption, then $\forall \mathbf{r}$, $\sum_k H(r^*_k) = 0 \geq \sum_k H(r_k)$. Therefore, $G(\mathbf{r}^*;\lambda) > G(\mathbf{r};\lambda), \forall \mathbf{r} \neq \mathbf{r}^*$. So $\mathbf{r}^*$ is the unique solution of $\max_{\mathbf{r}} G(\mathbf{r};\lambda)$. Because $\partial G(\mathbf{r};\lambda)/\partial r_k = \lambda_k - s_k(\mathbf{r}) + h(r_k)$, Algorithm 4 tries to solve $\max_{\mathbf{r}} G(\mathbf{r};\lambda)$ with inaccurate gradients.

Let $\mathbf{v}^s(t)$ be the solution of the following differential equation (for $t \geq s$)

$$dv_k(t)/dt = \lambda_k - s_k(\mathbf{v}(t)) + h(v_k(t)), \forall k \tag{10.29}$$

with the initial condition that $\mathbf{v}^s(s) = \bar{\mathbf{r}}(s)$. So, $\mathbf{v}^s(t)$ can be viewed as the "ideal" trajectory of Algorithm 4 with the smoothed arrival rate and service rate. And (10.29) can be viewed as a continuous-time gradient algorithm to solve $\max_{\mathbf{r}} G(\mathbf{r}; \lambda)$. We have shown above that $\mathbf{r}^*$ is the unique solution of $\max_{\mathbf{r}} G(\mathbf{r}; \lambda)$. Therefore $\mathbf{v}^s(t)$ converges to the unique $\mathbf{r}^*$ for any initial condition.

Recall that in Algorithm 4, $\mathbf{r}(i)$ is always updated at the beginning of a minislot. Define $Y(i-1) := (s'_k(i), w_0(i))$ where $w_0(i)$ is the state w at time t_i. Then $\{Y(i)\}$ is a non-homogeneous Markov process whose transition kernel from time t_{i-1} to t_i depends on $\mathbf{r}(i-1)$. The update in Algorithm 4 can be written as

$$r_k(i) = r_k(i-1) + \alpha(i) \cdot [f(r_k(i-1), Y(i-1)) + M(i)]$$

where $f(r_k(i-1), Y(i-1)) := \lambda_k - s'_k(i) + h(r_k(i-1))$, and $M(i) = \lambda'_k(i) - \lambda_k$ is Martingale noise.

To use Corollary 8 in page 74 of [74] to show Algorithm 4's almost-sure convergence to $\mathbf{r}^*$, the following conditions are sufficient:

(i) $f(\cdot, \cdot)$ is Lipschitz in the first argument, and uniformly in the second argument. This holds by the construction of $h(\cdot)$;

(ii) The transition kernel of $Y(i)$ is continuous in $\mathbf{r}(i)$. This is true due to the way we randomize the transmission lengths in (10.10).

(iii) (10.29) has a unique convergent point $\mathbf{r}^*$, which has been shown above;

(iv) With Algorithm 4, $r_k(i)$ is bounded $\forall k, i$ almost surely. This is proved in Lemma 16 below.

(v) Tightness condition ((†) in [74], page 71): This is satisfied since $Y(i)$ has a bounded state-space (cf. conditions (6.4.1) and (6.4.2) in [74], page 76). The state space of $Y(i)$ is bounded because $s'_k(i) \in [0, 1]$ and $w_0(i)$ is in a finite set (which is shown in Lemma 17) below.

So, by [74], $\mathbf{r}(i)$ converges to $\mathbf{r}^*$ almost surely.

Lemma 16. *With Algorithm 4, $\mathbf{r}(i)$ is always bounded. Specifically, $r_k(i) \in [r_{min} - 2, r_{max} + 2\bar{\lambda}], \forall k, i$, where $\bar{\lambda}$, as defined before, is the maximal instantaneous arrival rate, so that $\lambda'_k(i) \leq \bar{\lambda}, \forall k, i$.*

Proof. We first prove the upper bound $r_{max}+2\bar{\lambda}$ by induction: (a) $r_k(0) \le r_{max} \le r_{max}+2\bar{\lambda}$; (b) For $i \ge 1$, if $r_k(i-1) \in [r_{max}+\bar{\lambda}, r_{max}+2\bar{\lambda}]$, then $h(r_k(i-1)) \le -\bar{\lambda}$. Since $\lambda'_k(i)-s'_k(i) \le \bar{\lambda}$, we have $r_k(i) \le r_k(i-1) \le r_{max}+2\bar{\lambda}$. If $r_k(i-1) \in (r_{min}, r_{max}+\bar{\lambda})$, then $h(r_k(i-1)) \le 0$. Also since $\lambda'_k(i)-s'_k(i) \le \bar{\lambda}$ and $\alpha(i) \le 1, \forall i$, we have $r_k(i) \le r_k(i-1)+\bar{\lambda}\cdot\alpha(i) \le r_{max}+2\bar{\lambda}$. If $r_k(i-1) \le r_{min}$, then

$$\begin{aligned}
r_k(i) &= r_k(i-1) + \alpha(i)[\lambda'_k(i) - s'_k(i) + h(r_k(i-1))] \\
&\le r_k(i-1) + \alpha(i)\{\bar{\lambda} + [r_{min} - r_k(i-1)]\} \\
&= [1-\alpha(i)] \cdot r_k(i-1) + \alpha(i)\{\bar{\lambda} + r_{min}\} \\
&\le [1-\alpha(i)] \cdot r_{min} + \alpha(i)\{\bar{\lambda} + r_{min}\} \\
&= r_{min} + \alpha(i) \cdot \bar{\lambda} \\
&\le \bar{\lambda} + r_{min} \le r_{max} + 2\bar{\lambda}.
\end{aligned}$$

The lower bound $r_{min} - 2$ can be proved similarly. □

Lemma 17. *In Algorithm 4, $w_0(i)$ is in a finite set.*

Proof. By Lemma 16, we know that $r_k(i) \le r_{max} + 2\bar{\lambda}, \forall k, i$, so $T_k^p(i) \le T_0 \exp(r_{max} + 2\bar{\lambda}), \forall k, i$. By (10.10), we have $\tau_k^p(i) \le T_0 \exp(r_{max} + 2\bar{\lambda}) + 1, \forall k, i$. Therefore, in state $w_0(i) = \{x, ((b_k, a_k), \forall k : x_k = 1)\}$, we have $b_k \le b_{max}$ for a constant b_{max} and $a_k \le b_k$ for any k such that $x_k = 1$. So, $w_0(i)$ is in a finite set. □

Part (ii): Constant step size

The intuition is the same as part (i). That is, if the constant step size is small enough, then the algorithm approximately solves problem (10.24). Please refer to [73] for the full proof.

10.6 Theorem 7 as a unification of several existing models

As mentioned before, Theorem 7 turns out to be a quite general result which can specialize to the following well-known cases.

10.6.1 Slotted Aloha

By setting $\gamma = 1$, $T_k = 1, \forall k$ in our model, both the collision length and transmission length are one slot (of course, the slot here can be longer than $\sigma = 9\mu s$ in 802.11a), which is slotted Aloha. By Theorem 7, we have

$$p(x) = \frac{1}{E} \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i}.$$

It is easy to see that the normalizing constant $E = 1$. Thus, $p(x) = \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i}$, as expected.

10.6.2 CSMA with the complete conflict graph (e.g., in a single-cell wireless LAN)

If the conflict graph is a complete graph, any pair of links conflict with each other, i.e., if more than one link transmit at the same time, there is a collision. In this scenario, the collision number $h(x) \le 1$. If $h(x) = 1$. then call x a "collision state". Also note that $|S(x)| = 0$, so by Theorem 7,

$$p(x) = \frac{1}{E}(\gamma^{h(x)} \prod_{k \in S(x)} T_k) \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i} = \frac{1}{E}\gamma \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i}. \tag{10.30}$$

If $h(x) = 0$ and $|S(x)| = 0$, then no link is active ($x_k = 0, \forall k$) in the state. In this case x is a "idle state", and

$$p(x) = \frac{1}{E} \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i} = \frac{1}{E} \prod_{i \in \mathcal{N}} q_i. \tag{10.31}$$

Finally, if $h(x) = 0$ and $|S(x)| = 1$, then only one link, denoted by $s(x)$, is active and its transmission is successful. Then x as a "success state", and

$$p(x) = \frac{1}{E} T_{s(x)} \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i} = \frac{1}{E} T_{s(x)} p_{s(x)} \prod_{i \neq s(x)} q_i. \tag{10.32}$$

Equation (10.30), (10.31) and (10.32) agree to the results in [65] for 802.11 networks when all links conflict with each other.

10.6.3 Idealized CSMA

In idealized CSMA [50], it is assumed that the carrier-sensing is instant (without delay). Therefore, a minislot is arbitrarily small such that the transmission probability p_i in each minislot can be made arbitrarily small to avoid collisions.

Consider a sequence of system indexed by n where the length of a minislot decreases like $1/n$. Since the packet transmission length and collision length are measured by the numbers of minislots, suppose that $\gamma^{(n)} = n \cdot \gamma$, $T_k^{(n)} = n \cdot T_k$. And suppose $p_k^{(n)} = p_k/n$. By Theorem 7, we have

$$\begin{aligned}
& p^{(n)}(x) \\
= \; & \frac{1}{E^{(n)}}((\gamma^{(n)})^{h(x)} \prod_{k \in S(x)} T_k^{(n)}) \prod_{i \in \mathcal{N}} (p_i^{(n)})^{x_i} (1 - p_i^{(n)})^{1-x_i} \\
= \; & \frac{1}{E^{(n)}} \prod_{k \in S(x)} [T_k^{(n)} p_k^{(n)}] \cdot \prod_{m:|C_m(x)|>1} [\gamma^{(n)}(\prod_{j \in C_m(x)} p_j^{(n)})] \cdot \prod_{i:x_i=0} (1 - p_i^{(n)}) \\
= \; & \frac{1}{E^{(n)}} \prod_{k \in S(x)} (T_k p_k) \cdot \prod_{m:|C_m(x)|>1} [\frac{\gamma}{n^{|C_m(x)|-1}} \prod_{j \in C_m(x)} p_j] \cdot \prod_{i:x_i=0} (1 - p_i^{(n)}).
\end{aligned}$$

As $n \to \infty$, $\frac{\gamma}{n^{|C_m(x)|-1}} \to 0$ for $m \in \mathcal{G}(x)$ and $\prod_{i:x_i=0}(1 - p_i^{(n)}) \to 1$. Therefore

$$\begin{aligned}
p^{(n)}(x) \; & \to \; \frac{1}{E_\infty} \prod_{k \in S(x)} (T_k p_k) \text{ if there is no collision component} \\
p^{(n)}(x) \; & \to \; 0 \text{ if there exists any collision component}
\end{aligned} \tag{10.33}$$

where E_∞ is a proper normalizing constant.

As expected, there is no collision in the network as $n \to \infty$. Also, the non-collision states have a product-form distribution which matches the results in [50; 51; 52].

Chapter 11

Conclusion

11.1 Summary

We have proposed a distributed CSMA scheduling algorithm, and showed that in both cases with or without collisions, it is throughput-optimal under a general interference model. We have utilized the product-form stationary distribution of CSMA networks in order to obtain the distributed algorithm and its throughput optimality. In the case with collisions, the key idea is to adjust the transmission time instead of the backoff time in order to limit the effect of collisions. Furthermore, we have combined that algorithm with end-to-end congestion control to approach the maximal utility, and showed the connection with the classical back-pressure algorithm. Our algorithm is easy to implement, and the simulation results are encouraging.

The adaptive CSMA algorithm is a modular MAC-layer protocol that can work with other protocols in the transport layer and network layer. For example, we have demonstrated its combination with optimal routing, anycast and multicast (with network coding).

11.2 Future work

Our current performance analysis of Algorithm 1 and 2 is based on a separation of time scales, i.e., the vector $\mathbf{r}$ is adapted slowly to allow the CSMA Markov chain to closely track the stationary distribution $\mathbf{p}(\mathbf{r})$. The simulations, however, indicate that such slow adaptations are not always necessary. In the future, we are interested to understand more about the case without time-scale separation. This case is more challenging since we need to analyze a "joint" Markov chain whose state is $(x, \mathbf{r})$, where $x \in \{0,1\}^K$ is the transmission state and $\mathbf{r} \in \mathcal{R}^K$ is the TA vector (instead of separating them due to their different time scales). New techniques are needed in order to establish the stability of the joint Markov chain.

Although it is not clear whether a complete time-scale separation is necessary, we do observe through simulations that, if the CSMA Markov chain mixes fast, then CSMA scheduling performs better in terms of the delay and delay oscillation. We think the reason is that with a faster mixing Markov chain, the transmission states with larger weights are easier to reach. This opens doors to another research direction related to Markov chain Monte Carlo, that is to design faster-mixing Markov chains (or network dynamics) with the desired stationary distribution. However, the new network dynamics should be amenable to distributed implementation in order to keep the advantages of CSMA scheduling.

Another interesting direction is the combination of CSMA scheduling with another scheduling algorithm in order to get the advantages of both. In [62], CSMA is combined with LQF (Longest-Queue-First) for the purpose of better delay performance. The performance analysis of the hybrid algorithm, however, still requires further research.

Part III

Scheduling in Stochastic Processing Networks

Chapter 12

Stable and Utility-Maximizing Scheduling in SPNs

12.1 Introduction

Stochastic Processing Networks (SPNs) are models of service, processing, communication, or manufacturing systems [75]. In such a network, service activities require parts and resources to produce new parts. Thus, parts flow through a network of buffers served by activities that consume parts and produce new ones. Typically, service activities compete for resources, which yields a scheduling problem. The goal of the scheduling is to maximize some measure of performance of the network, such as the net utility of parts being produced.

As SPNs are more general than queuing networks, one may expect the scheduling that minimizes an average cost such as total waiting time to be complex. Indeed, the optimal scheduling of queuing networks is known only for simple cases, such as serving the longest queue or the Klimov network [76]. For SPNs, one approach has been to consider these networks under the heavy-traffic regime [77]. In such a regime, a suitable scheduling may collapse the state space. For instance, when serving the longest queue, under heavy traffic the queue lengths become equal. It is then sometimes possible to analyze the SPN under heavy traffic as in [78]. Using this approach, in [79], the authors prove the asymptotic optimality under heavy traffic of maximum pressure policies for a class of SPNs. It may also happen that the control of the heavy traffic diffusion model is tractable while the original problem is not [81].

Another line of investigation explores a less ambitious formulation of the problem. Instead of considering the Markov decision problem of minimizing an average cost, this approach searches for controls that stabilize the network or that maximize the utility of its flows. This approach has been followed successfully for communication networks.

Tassiulas and Ephremides [47] proposed a maximum weight scheduling algorithm (MWS) that schedules the independent set (non-conflicting nodes) with the maximum sum of queue lengths. The authors prove that this algorithm achieves the maximum possible throughput. The central idea of considering the maximization of the sum of the user utilities is due to [82]. See also [83; 84]. Combining this objective with the scheduling appears in [85; 86]. In these works, it is shown that maximum backpressure algorithms combined with congestion control maximize the total utility of the flows in the network.

This chapter follows a similar approach. The objective is to achieve throughput optimality and maximize the total net utility of flows of parts that the network produces. However, the scheme proposed in the chapter differs from previous work. For instance, simple examples show that MWS is not stable for some SPNs and that a new approach is needed. The basic difficulty is that the maximum weight and related algorithms are too greedy and may lead some service activities to starve other service activities. Dai and Lin [80] show that MWS is stable in SPNs *if* the network structure satisfies a certain assumption (for example, in a limited class of SPNs where each service activity consumes parts from a single queue). We propose a *deficit maximum weight* (DMW) algorithm [87] that automatically makes certain service activities wait instead of always grabbing the parts they can use, therefore achieving throughput optimality without the assumption in [80].

The chapter is organized as follows. Section 12.2 illustrates through examples the basic difficulties of scheduling SPNs and the operations of the DMW scheduling algorithm. Section 12.3 defines the basic model. Section 12.4 describes the DMW algorithm formally and proves that it stabilizes the network. Section 12.5 explains that the algorithm, combined with the control of the input activities, maximizes the sum of the utilities of the network. Section 12.7 provides a number of simulation results to confirm the results of the chapter.

12.2 Examples

This section illustrates critical aspects of the scheduling of SPNs on simple examples. Figure 12.1 shows a SPN with one *input activity* (IA) represented by the shaded circle and four *service activities* (SAs) represented by white circles. SA2 needs one part from queue 2 and produces one part that leaves the network, similarly for SA4. SA3 needs one part from each of the queues 2, 3 and 4 and produces one part that leaves the network. SA1 needs one part from queue 1 and produces one part which is added to queue 4. Each SA takes one unit of time. There is a dashed line between two SAs if they cannot be performed simultaneously. These conflicts may be due to common resources that the SAs require. The parts arrive at the queues as follows: at even times, IA1 generates one part for each of the queues 1, 2 and 3; at odd times, no part arrives.

One simple scheduling algorithm for this network is as follows. At time 0, buffer the parts that arrive at queues 1, 2 and 3. At time 1, perform SA1 which removes one part from queue 1 and adds one part to queue 4. At time 2, use the three parts in queue 2, 3, 4 to perform SA3 and buffer the new arrivals. Repeat this schedule forever, i.e., perform SA1 and SA3 alternately. This schedule makes the system stable.

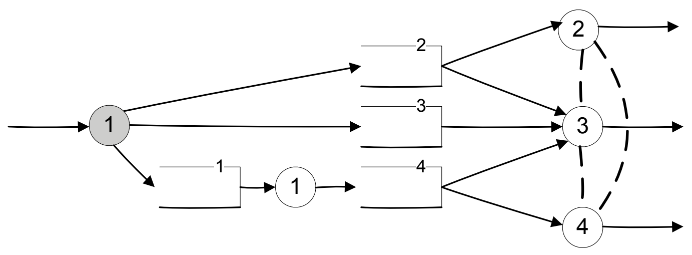


Figure 12.1: A network unstable under MWS

Interestingly, the maximum weight algorithm makes this system unstable (in a similar way to a counter example in [80]). By definition, at each time, this algorithm schedules the SAs that maximize the sum of the backpressures. Accordingly, at time 1, one part has arrived in queue 1, 2 and 3 (at time 0). Since queue 4 is empty, SA3 and SA4 cannot be scheduled, so this algorithm schedules SA1 and SA2, after which one part remains in queue 3 and queue 4. At time 2, the algorithm schedules SA4, and buffers new arrivals, after which two parts remain in queue 3, and one part in queue 1 and queue 2. Continuing in this way, the number of parts in queue 3 increases without bound since the algorithm never schedules SA3 and never serves queue 3. (In fact, any work-conserving algorithm leads to the same result in this example.) The deficit maximum weight algorithm that we propose in this chapter addresses this instability.

Fig. 12.2 provides another example of instability, this time due to randomness. There, SA1 processes each part in queue 1 and then produces one part for queue 2 or queue 3, each with probability 0.5. Each activation of SA2 assembles one part from queue 2 and one part from queue 3. Each SA takes one unit of time. If the parts arrive at queue 1 at rate $\lambda_1 < 1$, then one would expect the SPN to be able to process these parts. However, the difference between the number of parts that enter the queues 2 and 3 is null recurrent. Thus, no scheduling algorithm can keep the backlogs in the queues 2 and 3 bounded at the same time. In this chapter, we are only interested in those networks which are feasible to stabilize.

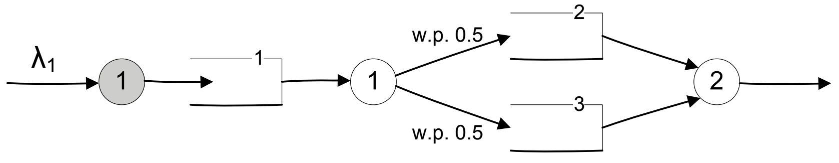


Figure 12.2: An infeasible example

Figure 12.3 shows another SPN. IA1 produces one part for queue 1. IA2 produces one part for queue 2 and one part for queue 3. The synchronized arrivals generated by IA2 correspond to the ordering of a pair of parts, as one knows that such a pair is needed for SA2. This mechanism eliminates the difficulty encountered in the example of Figure 12.2. In Figure 12.3, we say that each IA is "source" of a "flow" of parts (as a generalization of a “flow” in data networks). SA1 and SA2 in this network conflict, as indicated by the dashed line between the SAs. Similarly, SA2 and SA3 conflict. One may consider the problem of scheduling both the IAs (ordering parts) and the SAs to maximize some measure of performance. Our model assumes the appropriate ordering of sets of parts to match the requirements of the SAs.

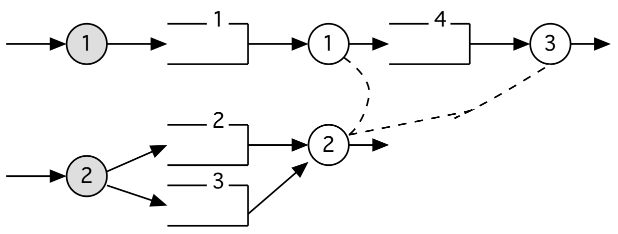


Figure 12.3: Arrivals and conflicting service activities

We explain the *deficit maximum weight* (DMW) scheduling algorithm on the example of Figure 12.1. In that example, we saw that MWS is unstable because it starves SA3. Specifically, MWS schedules SA2 and SA4 before the three queues can accumulate parts for SA3. The idea of DMW is to pretend that certain queues are empty even when they have parts, so that the parts can wait for the activation of SA3. The algorithm is similar to MWS but the weight of each SA is computed from the “virtual queue lengths” $q_k = Q_k - D_k, \forall k$. Here, Q_k is the actual length of queue k and $D_k \geq 0$ is called “deficit”.

DMW automatically finds the suitable values of the deficits D_k. To do this, DMW uses the maximum-weighted schedules without considering whether there are enough input parts available. When the algorithm activates a SA which does not have enough input parts in queue k, the SA produces fictitious parts, decreases q_k (which is allowed to be negative) and increases the deficit of queue k. This algorithm produces the results in Table 12.1, where each column gives the values of $\mathbf{q}$ and $\mathbf{D}$ after the activities in a slot. For deficits, only D_4 is shown since the deficits of all other queues are 0. In the table, SA0 means that no SA is scheduled because all the weights of the activities are non-positive. Note that when SA3 is activated for the first time, queue 4 is empty: $Q_4 = 0$. Therefore q_4 is decreased to -1, D_4 is increased to 1 and a fictitious part is produced. But since SA1 is activated simultaneously, q_4 becomes 0 after this slot. After that, the sequence

Activity→		SA0+IA1	SA3+SA1	SA0+IA1	SA3+SA1	...
q_1	0	1	0	1	0	...
q_2	0	1	0	1	0	...
q_3	0	1	0	1	0	...
q_4	0	0	0	0	0	...
D_4	0	0	1	1	1	

Table 12.1: Deficit Maximum Weight scheduling

(SA0+IA1, SA3+SA1) repeats forever and no more fictitious parts are produced. The key observation is that, although the virtual queue lengths are allowed to become negative, they remain bounded in this example. Consequently, with proper $\mathbf{D}$, the actual queue lengths $\mathbf{Q} = \mathbf{q} + \mathbf{D}$ are always non-negative, and thus avoid the starvation problem.

12.3 Basic model

For simplicity, assume a time-slotted system. In each slot, a set of input activities (IAs) and service activities (SAs) are activated. Assume that each activity lasts one slot for the ease of exposition. (In section 12.6, we will discuss the case where different activities have different durations.) There are M IAs, N SAs, and K queues in the network. Each IA, when activated, produces a deterministic number of parts for each of a fixed set of queues. Each SA, when activated, consumes parts from a set of queues and produces parts for another set of queues, and/or some "products" that leave the network.

The set of IAs, SAs and queues are defined to be consistent with the following. (i) Each IA is the "source" of a "flow" of parts, like in Figure 12.3. In other words, the parts generated by IA m can be exactly served by activating some SAs and eventually produce a number of products that leave the network. (This will be made more formal later.) Otherwise, it is impossible to stabilize the network. There are M IAs and M flows. (ii) Parts in different flows are buffered in separate queues. (iii) A SA n is associated with a set of input queues $\mathcal{I}_n$ and a set of output queues $\mathcal{O}_n$. Due to the way we define the queues in (ii), different flows are served by *disjoint* sets of SAs. (Even if two SAs in different flows essentially perform the same task, we still label them differently.) Also, *a SA is defined only if it is used by some flow.*

Each activation of IA m adds $a_{k,m}$ parts to queue k. Define the "input matrix" $A \in \mathcal{R}^{K*M}$ where $A_{k,m} = -a_{k,m}, \forall m, k$. Each activation of SA n consumes $b_{k,n}$ parts from each queue $k \in \mathcal{I}_n$ (the "input set" of SA n), and produces $b^{'}_{k,n}$ parts that are added to each queue $k \in \mathcal{O}_n$ (the "output set" of SA n), and (possible) a number of final products that leave the network. Assume that $\mathcal{I}_n \cap \mathcal{O}_n = \emptyset$. Accordingly, define the "service matrix" $B \in \mathcal{R}^{K*N}$, where $B_{k,n} = b_{k,n}$ if $k \in \mathcal{I}_n$, $B_{k,n} = -b^{'}_{k,n}$ if $k \in \mathcal{O}_n$, and $B_{k,n} = 0$ otherwise. Assume that all elements of A and B are integers. Also assume that the directed graph that represents the network has no cycle (see, for example, Fig. 12.1 and Fig. 12.3).

Let $\mathbf{a}(t) \in \{0,1\}^M, t = 0, 1, \dots$ be the "arrival vector" in slot t, where $a_m(t) = 1$ if IA m is activated and $a_m(t) = 0$ otherwise. Let $\lambda \in \mathcal{R}^M$ be the vector of (average) arrival rates. Let $\mathbf{x}(t) \in \{0,1\}^N$ be the "service vector" in slot t, where $x_n(t) = 1$ if SA n is activated and $x_n(t) = 0$ otherwise. Let $\mathbf{s} \in \mathcal{R}^N$ be a vector of (average) service rates.

Point (i) above means that, for any activation rate $\lambda_m > 0$ of flow m, there exists $\mathbf{s}_m \in \mathcal{R}^N$ such that

$$A_m \cdot \lambda_m + B \cdot \mathbf{s}_m = \mathbf{0} \tag{12.1}$$

where A_m is the m'th column of A. The vector $\mathbf{s}_m$ is the service rate vector for flow m that can serve λ_m. We also make the reasonable assumption that $\mathbf{s}_m$ is unique given λ_m, i.e., there is only one way to serve the arrivals. Summing up (12.1) over m gives

$$A \cdot \lambda + B \cdot \mathbf{s} = \mathbf{0}. \tag{12.2}$$

where $\mathbf{s} = \sum_m \mathbf{s}_m \succ \mathbf{0}$. (Note that since each flow is associated with a separate set of queues and SAs, equation (12.2) also implies (12.1) for all m.) By assumption, $\mathbf{s}$ is unique given λ, so we also write $\mathbf{s}$ in (12.2) as $\mathbf{s}(\lambda)$.

Due to resource sharing constraints among the SAs, not all SAs can be performed simultaneously at a given time. Assuming that all queues have enough parts such that any SA can be performed, let $\tilde{\mathbf{x}} \in \{0,1\}^N$ be a feasible service vector, and $\mathcal{X}$ be the set of such $\tilde{\mathbf{x}}$'s. (We also call $\tilde{\mathbf{x}}$ an independent set since the active SAs in $\tilde{\mathbf{x}}$ can be performed without conflicts.) Denote by Λ be the convex hull of $\mathcal{X}$, i.e.,

$$\Lambda := \{\mathbf{s} | \exists \mathbf{p} \succcurlyeq \mathbf{0} : \sum_{\tilde{\mathbf{x}} \in \mathcal{X}} p_{\tilde{\mathbf{x}}} = 1, \mathbf{s} = \sum_{\tilde{\mathbf{x}} \in \mathcal{X}} (p_{\tilde{\mathbf{x}}} \cdot \tilde{\mathbf{x}})\}$$

and let Λ^o be the interior of Λ. (That is, for any $\mathbf{s} \in \Lambda^o$, there is a ball $\tilde{\mathcal{B}}$ centered at $\mathbf{s}$ with radius $r > 0$ such that $\tilde{\mathcal{B}} \subset \Lambda$.)

We say that λ is *feasible* iff $\mathbf{s}(\lambda) \in \Lambda$;$\lambda$ is *strictly feasible* iff $\mathbf{s}(\lambda) \in \Lambda^o$.

In a more general setting, the output parts of a certain SA can split and go to more than one output sets. The split can be random or deterministic. For example, in a hospital, after a patient is diagnosed, he goes to a certain room based on the result. A probabilistic model for this is that the patients go to different rooms with certain probabilities after the SA (i.e., the diagnosis). The split can also be deterministic. For example, in manufacturing, the output parts of a SA may be put into two different queues alternately.

In both cases, we can define the element $B_{k,n}$ in the matrix B to be the average rate that SA n consumes (or adds) parts from (to) queue k. However, note that in the random case, it may not be feasible to stabilize all queues by any algorithm, even if there exist average rates satisfying (12.1). Fig. 12.2 described earlier is such an example. For simplicity, here we mainly consider networks without splitting.

12.4 DMW scheduling

In this section we consider the scheduling problem with strictly feasible arrival rates λ. We first describe the DMW algorithm and then show its throughput optimality.

Let the actual queue lengths at time t be $Q_k(t)$, $k = 1, 2, \cdots, K$. Define a "deficit" $D_k(t) \geq 0$. DMW uses the "virtual queue length" $q_k(t) = Q_k(t) - D_k(t)$ to compute the schedule in each slot.

DMW (Deficit Maximum Weight) Scheduling

Initially (at time 0), set $\mathbf{q}(0) = \mathbf{Q}(0) = \mathbf{D}(0) = \mathbf{0}$. Clearly, $\mathbf{Q}(0) = \mathbf{q}(0) + \mathbf{D}(0)$.

(i) Update of virtual queues $\mathbf{q}(t)$: In each time slot $t = 0, 1, 2 \ldots$, the set of SAs with the maximal backpressure is scheduled:

$$\mathbf{x}^*(t) \in \arg\max_{\mathbf{x} \in \mathcal{X}} d^T(t) \cdot \mathbf{x} \tag{12.3}$$

where $\mathbf{d}(t) \in \mathcal{R}^N$ is the vector of backpressure, defined as

$$\mathbf{d}(t) = B^T \mathbf{q}(t), \tag{12.4}$$

and $\mathcal{X}$ is the set of independent sets including non-maximal ones. Also, for any SA n, we require that $x_n^*(t) = 0$ if $d_n(t) \leq 0$.

Recall that an independent set is a set of SAs that can be performed simultaneously *assuming that all input queues have enough parts.* So, it is possible that SA n is scheduled (i.e., $x_n^*(t) = 1$) even if there are not enough parts in some input queues of SA n. In this case, SA n is activated as a "null activity" (to be further explained in (ii)). Then, update $\mathbf{q}$ as

$$\mathbf{q}(t+1) = \mathbf{q}(t) - A \cdot \mathbf{a}(t) - B \cdot \mathbf{x}^*(t) \tag{12.5}$$

where, as defined earlier, $\mathbf{a}(t)$ is the vector of actual arrivals in slot t (where the m'th element $a_m(t)$ corresponds to IA m). In this chapter, $\mathbf{x}^*(t)$ and $\mathbf{x}^*(\mathbf{q}(t))$ are interchangeable.

Expression (12.5) can also be written as

$$q_k(t+1) = q_k(t) - \mu_{out,k}(t) + \mu_{in,k}(t), \forall k$$

where $\mu_{out,k}(t)$ and $\mu_{in,k}(t)$ are the number of parts coming out of or into virtual queue k in slot t, expressed below. (We use v^+ and v^- to denote the positive and negative part of v. That is, $v^+ = \max\{0, v\}$ and $v^- = \max(0, -v\}$, so that $v = v^+ - v^-$.)

$$\begin{aligned} \mu_{out,k}(t) &= \textstyle\sum_{n=1}^N [B_{k,n}^+ x_n^*(t)] \\ \mu_{in,k}(t) &= \textstyle\sum_{m=1}^M [A_{k,m}^- a_m(t)] + \sum_{n=1}^N [B_{k,n}^- x_n^*(t)]. \end{aligned}$$

(ii) Update of actual queues $\mathbf{Q}(t)$ and deficits $\mathbf{D}(t)$: If SA n is scheduled in slot t but there are not enough parts in some of its input queues (or some input parts are fictitious, further explained below), SA n is activated as a null activity. Although the null activity n

does not actually consume or produce parts, parts are removed from the input queues and fictitious parts are added to the output queues as if SA n was activated normally. So the actual queue length

$$Q_k(t+1) = [Q_k(t) - \mu_{out,k}(t)]_+ + \mu_{in,k}(t). \tag{12.6}$$

Then the deficit is computed as

$$D_k(t+1) = Q_k(t+1) - q_k(t+1). \tag{12.7}$$

Remark: The key idea of DMW is to augment the "state" by including the virtual queues $\mathbf{q}(t)$, and then show the relationship between the virtual queues and actual queues.

The following is a useful property of $D_k(t)$.

Lemma 18. *$D_k(t)$ is non-decreasing with t, and satisfies*

$$D_k(t+1) = D_k(t) + [\mu_{out,k}(t) - Q_k(t)]_+.$$

Proof. By (12.7), (12.5) and (12.6), we have

$$\begin{aligned} D_k(t+1) &= Q_k(t+1) - q_k(t+1) \\ &= [Q_k(t) - \mu_{out,k}(t)]_+ - [q_k(t) - \mu_{out,k}(t)] \\ &= Q_k(t) - \mu_{out,k}(t) + [\mu_{out,k}(t) - Q_k(t)]_+ \\ &\quad -[q_k(t) - \mu_{out,k}(t)] \\ &= D_k(t) + [\mu_{out,k}(t) - Q_k(t)]_+, \end{aligned} \tag{12.8}$$

which also implies that $D_k(t)$ is non-decreasing with t. □

Proposition 27. *If $||\mathbf{q}(t)||^2 \leq G$ at all time t for some constant $G > 0$, then*

(i) $\mathbf{D}(t)$ is bounded. Also, only a finite number of null activities occur. So in the long term the null activities do not affect the average throughput.

(ii) $\mathbf{Q}(t)$ is bounded.

Proof. Since $||\mathbf{q}(t)||^2 \leq G$, we have $-G' \leq q_k(t) \leq G', \forall k, t$ where $G' := \lceil \sqrt{G} \rceil,$. We claim that $D_k(t) \leq G' + \mu_{out}, \forall k, t$ where μ_{out} is the maximum number of parts that could leave a queue in one slot. By the definition of the DMW algorithm, $D_k(t)$ is non-decreasing with t and initially $D_k(t) = 0$. Suppose to the contrary that $D_k(t)$ is above $G' + \mu_{out}$ for some k and t. Then there exists $t^{'}$ which is the first time that $D_k(t^{'})$ is above $G' + \mu_{out}$. In other words, $D_k(t^{'}) > G' + \mu_{out}$ and $D_k(t^{'} - 1) \leq G' + \mu_{out}$.

By (12.8) and (12.7), we have

$$\begin{aligned} D_k(t+1) &= D_k(t) + [\mu_{out,k}(t) - Q_k(t)]_+ \\ &= D_k(t) + \max\{0, \mu_{out,k}(t) - Q_k(t)\} \\ &= \max\{D_k(t), D_k(t) + \mu_{out,k}(t) - Q_k(t)\} \\ &= \max\{D_k(t), -q_k(t) + \mu_{out,k}(t)\} \end{aligned}$$

So $D_k(t^{'}) = \max\{D_k(t^{'} - 1), -q_k(t^{'} - 1) + \mu_{out,k}(t' - 1)\}$. Since $q_k(t^{'} - 1) \geq -G'$, $\mu_{out,k}(t) \leq \mu_{out}$, we have $D_k(t^{'}) \leq G' + \mu_{out}$. This leads to a contradiction. Therefore, $D_k(t) \leq G' + \mu_{out}, \forall t, k$.

Note that when a queue underflow (i.e., when $\mu_{out,k}(t) > Q_k(t)$ for some k, t) occurs, D_k is increased. Also, the increase of D_k is a positive integer. Since $\mathbf{D}(0) = \mathbf{0}$, $\mathbf{D}(t)$ is non-decreasing and remains bounded for all t, the number of queue underflows must be finite. Since we have assumed that the directed graph which represents the network has no cycle, it is clear that each underflow only "pollutes" a finite number of final outputs (i.e., the products). Therefore, in the long term the queue underflows (and the resulting null activities) do not affect the average throughput.

Part (ii)**:** $Q_k(t) = q_k(t) + D_k(t) \leq 2G' + \mu_{out}, \forall k, t$**.** □

In section 12.4.1 we will show that $\mathbf{q}(t)$ is bounded under certain conditions on the arrivals. By Proposition 27, $\mathbf{Q}(t)$ is bounded and the maximal throughput is achieved. But before that, we need to identify some useful properties of the system. Our analysis differs from existing analysis of MWS-like algorithms, e.g., in [47; 85], since we allow $q_k(t)$ to be negative, and an activity generally involves multiple input and output queues.

Lemma 19. *In DMW, if* $\mathbf{d}(t) = \mathbf{0}$ *at some time* t*, then* $\mathbf{q}(t) = \mathbf{0}$.

Proof. Let $z_m(t)$ and $w_n(t)$ be, respectively, the number of times that IA m and SA n have been performed until time t. Write $\mathbf{z}(t)$ and $\mathbf{w}(t)$ as the corresponding vectors. Using $\mathbf{q}(0) = \mathbf{0}$ and equation (12.5), we have

$$\mathbf{q}(t) = -A \cdot \mathbf{z}(t) - B \cdot \mathbf{w}(t). \tag{12.9}$$

For $\lambda_m = 1$, there exists $\mathbf{s}_m = \mathbf{s}'_m$ such that (12.1) holds. So $A_m = -B \cdot \mathbf{s}'_m$. Using this and (12.9),

$$\begin{aligned} \mathbf{q}(t) &= \textstyle\sum_m [B \cdot \mathbf{s}'_m z_m(t)] - B \cdot \mathbf{w}(t) \\ &= B \cdot \mathbf{v} \end{aligned}$$

where $\mathbf{v} := \sum_m [\mathbf{s}'_m z_m(t)] - \mathbf{w}(t)$. By assumption,

$$\mathbf{d}(t) = B^T \mathbf{q}(t) = B^T B \cdot \mathbf{v} = \mathbf{0}.$$

Thus, $\mathbf{v}^T B^T B \cdot \mathbf{v} = ||B \cdot \mathbf{v}||^2 = 0$. So, $B \cdot \mathbf{v} = \mathbf{q}(t) = \mathbf{0}$. □

Remark: We have used the fact that for any t, $\mathbf{q}(t)$ is always in the subspace $\mathcal{B} := \{\mathbf{u} | \mathbf{u} = B \cdot \mathbf{v} \text{ for some } \mathbf{v}\}$.

Lemma 20. *Assume that* λ *is a strictly feasible, i.e.,* $\exists \mathbf{y} \in \Lambda^o$ *such that*

$$A \cdot \lambda + B \cdot \mathbf{y} = \mathbf{0}. \tag{12.10}$$

Then there exists $\delta > 0$ *such that for any* $\mathbf{q}$ *satisfying* $\mathbf{q} \in \mathcal{B}$,

$$q^T B \cdot [\mathbf{x}^*(\mathbf{q}) - \mathbf{y}] \geq \delta ||\mathbf{q}|| \tag{12.11}$$

where

$$\mathbf{x}^*(\mathbf{q}) \in \arg\max_{\mathbf{x} \in \mathcal{X}} q^T B \cdot \mathbf{x}. \tag{12.12}$$

Proof. Since $\mathbf{y} \in \Lambda^o$, $\exists \sigma' > 0$ such that $\mathbf{y}' \in \Lambda$ for any $\mathbf{y}'$ satisfying $||\mathbf{y}' - \mathbf{y}|| \leq \sigma'$.

For any $\hat{\mathbf{q}}$ satisfying $||\hat{\mathbf{q}}|| = 1, \hat{\mathbf{q}} \in \mathcal{B}$, by Lemma 19, we have $\hat{\mathbf{d}} := B^T \hat{\mathbf{q}} \neq \mathbf{0}$. Also, $||B^T \hat{\mathbf{q}}|| \geq \hat{\sigma} := \min_{||\mathbf{q}'||=1, \mathbf{q}' \in \mathcal{B}} ||B^T \mathbf{q}'|| > 0$. Choose $\hat{\epsilon} > 0$ (which may depend on $\hat{\mathbf{q}}$) so that

$||\hat{\epsilon}\cdot B^T\hat{\mathbf{q}}|| = \sigma'$. Then, $\mathbf{y}+\hat{\epsilon}\cdot B^T\hat{\mathbf{q}} \in \Lambda$. Also, (12.12) implies that $\mathbf{x}^*(\mathbf{q}) \in \arg\max_{\mathbf{x}\in\Lambda} q^T B\cdot\mathbf{x}$. So, $\hat{\mathbf{q}}^T B \cdot \mathbf{x}^*(\hat{\mathbf{q}}) \geq \hat{\mathbf{q}}^T B \cdot [\mathbf{y} + \hat{\epsilon} \cdot B^T\hat{\mathbf{q}}] = \hat{\mathbf{q}}^T B \cdot \mathbf{y} + \hat{\epsilon} \cdot ||B^T\hat{\mathbf{q}}||^2 \geq \hat{\mathbf{q}}^T B \cdot \mathbf{y} + \hat{\sigma} \cdot \sigma'$. Let $\delta := \hat{\sigma}\cdot\sigma'$. Then

$$\min_{||\hat{\mathbf{q}}||=1,\hat{\mathbf{q}}\in\mathcal{B}} \hat{\mathbf{q}}^T B \cdot [\mathbf{x}^*(\hat{\mathbf{q}}) - \mathbf{y}] \geq \delta. \tag{12.13}$$

Consider any $\mathbf{q} \neq \mathbf{0}$. Let $\hat{\mathbf{q}} := \mathbf{q}/||\mathbf{q}||$, then $||\hat{\mathbf{q}}|| = 1$. Note that if $\mathbf{x}^*(\hat{\mathbf{q}}) \in \arg\max_{\mathbf{x}\in\mathcal{X}} \hat{\mathbf{q}}^T B\cdot \mathbf{x}$, then $\mathbf{x}^*(\hat{\mathbf{q}}) \in \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbf{q}^T B\cdot\mathbf{x}$ by linearity, so $q^T B\cdot\mathbf{x}^*(\hat{\mathbf{q}}) = q^T B \cdot \mathbf{x}^*(\mathbf{q})$. Therefore $q^T B \cdot [\mathbf{x}^*(\mathbf{q}) - \mathbf{y}] = q^T B \cdot [\mathbf{x}^*(\hat{\mathbf{q}}) - \mathbf{y}] = ||\mathbf{q}|| \cdot \hat{\mathbf{q}}^T B \cdot [\mathbf{x}^*(\hat{\mathbf{q}}) - \mathbf{y}] \geq \delta||\mathbf{q}||$, proving (12.11). If $\mathbf{q} = \mathbf{0}$, then (12.11) holds trivially. □

12.4.1 Arrivals that are smooth enough

Recall that λ is strictly feasible and (12.10) is satisfied. First consider a simple case when the arrival rates are "almost constant" at λ. Specifically, assume that $a_m(t) = \lfloor\lambda_m \cdot (t+1)\rfloor - \lfloor\lambda_m \cdot t\rfloor, \forall m, t$. Then $\sum_{\tau=0}^{t-1} a_m(\tau) = \lfloor\lambda_m \cdot t\rfloor \approx \lambda_m \cdot t, \forall t$, so that the arrival rates are almost constant. Later, we all show that $\mathbf{q}(t)$ is bounded under such arrivals. By Proposition 27, $\mathbf{Q}(t)$ is bounded and the maximal throughput is achieved.

However, since the "almost constant" assumption is quite strong in practice, it is useful to relax it and consider more general arrival processes. In particular, consider the following (mild) smoothness condition.

Condition 1: There exists $\sigma > 0$ and a positive integer T such that for all $l = 0, 1, 2, \ldots$, $\tilde{\mathbf{a}}_l + \sigma\cdot\mathbf{1}$ and $\tilde{\mathbf{a}}_l - \sigma\cdot\mathbf{1}$ are feasible vectors of arrival rates, where

$$\tilde{\mathbf{a}}_l \;:=\; \textstyle\sum_{\tau=l\cdot T}^{(l+1)\cdot T-1} \mathbf{a}(\tau)/T \tag{12.14}$$

is the vector of average arrival rates in the l'th time window of length T. In other words, there exists a large enough time window T such as the $\tilde{\mathbf{a}}_l$ is "uniformly" strictly feasible.

Remark: Note that $\tilde{\mathbf{a}}_l$ can be very different for different l's. That is, $\tilde{\mathbf{a}}_l, l = 0, 1, \ldots$ do not need to be all close to a certain strictly feasible λ.

Theorem 10. *Under Condition 1,* $\mathbf{q}(t)$ *is bounded for all* t*. Therefore (i) and (ii) in Prop. 27 hold.*

The proof is in section 12.9.1.

Corollary 3. *With the "almost constant" arrivals,* $\mathbf{q}(t)$ *is bounded for all* t.

Proof. Since $\sum_{\tau=0}^{t-1} a_m(\tau) = \lfloor \lambda_m \cdot t \rfloor$, we have $|\sum_{\tau=0}^{t-1} a_m(\tau) - \lambda_m \cdot t| \leq 1, \forall t$. So $|\sum_{\tau=l\cdot T}^{(l+1)\cdot T-1} a_m(\tau)/T - \lambda_m| = (1/T) \cdot |[\sum_{\tau=0}^{(l+1)\cdot T-1} a_m(\tau) - \lambda_m \cdot (l+1)T] - [\sum_{\tau=0}^{l\cdot T-1} a_m(\tau) - \lambda_m lT]| \leq 2/T$. Since λ is strictly feasible, there exists a large enough T to satisfy Condition 1. $\square$

12.4.2 More random arrivals

Assume that $a_m(t) \in \mathcal{Z}_+$ is a random variable with bounded support, and satisfies

$$E(a_m(t)) = \lambda_m, \forall t. \tag{12.15}$$

For simplicity, also assume that the random variables $\{a_m(t), m = 1, 2, \ldots, M, t = 0, 1, 2, \ldots\}$ are independent. (This assumption, however, can be easily relaxed.) Suppose that the vector λ is strictly feasible.

In general, this arrival process does not satisfy the smoothness condition (although when T is large, $\sum_{\tau=t}^{t+T-1} \mathbf{a}(\tau)/T$ is close to λ with high probability). With such arrivals, it is not difficult to show that $\mathbf{q}(t)$ is stable, but may not be bounded. As a result, the deficits $\mathbf{D}(t)$ may increase without bound. In this case, we show that the system is still "rate stable", in the sense that in the long term, the average output rates of the final products converge to the optimum output rates (with probability 1). The intuitive reason is that as $\mathbf{D}(t)$ becomes very large, the probability of generating fictitious parts approaches 0.

Theorem 11. *With the arrival process defined above, the system is "rate stable".*

The formal proof is given in section 12.9.2.

Although the system is throughput optimum, with $\mathbf{D}(t)$ unbounded, the actual queue lengths $\mathbf{Q}(t) = \mathbf{q}(t) + \mathbf{D}(t)$ may become large when $\mathbf{D}(t)$ is large. An alternative to avoid large $\mathbf{Q}(t)$ is to set an upper bound of $D_k(t)$, denoted by $\bar{D}$. In this alternative, we do not increase $D_k(t)$ once it hits $\bar{D}$. But $\mathbf{q}(t)$ still evolves according to part (i) of the DMW algorithm. Let the actual queue length be $Q_k(t) = [q_k(t) + D_k(t)]_+$. Fictitious parts are generated in slot t as long as $q_k(t) - \mu_{out,k}(t) < -D_k(t)$ (or, $Q_k(t) - \mu_{out,k}(t) < 0$). Given a $\bar{D}$, one expects that the output rates are lower than the optimum in general, since fictitious parts are generated with a certain probability after $D_k(t)$ first hits $\bar{D}$. But one can make the probability arbitrarily close to 0 by choose a large enough $\bar{D}$. The proof is similar to that of Theorem 11 and is not included here.

12.5 Utility maximization

Assume that for each IA m, there is a "reward function" $v_m(f_m)$ (where f_m is the activation rate), and a cost function $c_m f_m$, where c_m is the cost of the input materials of IA m per unit rate. Define the utility function as $u_m(f_m) := v_m(f_m) - c_m f_m$. Let $\mathbf{f} \in \mathcal{R}^M$ be the vector of input activation rates. Assume that $u_m(\cdot)$ is increasing and concave. The joint scheduling and congestion control algorithm (or "utility maximization algorithm") works as follows.

Utility Maximization Algorithm

Initially let $\mathbf{q}(0) = \mathbf{Q}(0) = \mathbf{0}$. In each time slot $t = 0, 1, 2, \ldots$, besides DMW Scheduling (12.3), i.e.,

$$\mathbf{x}^*(t) \in \arg\max_{\mathbf{x} \in \mathcal{X}} d^T(t) \cdot \mathbf{x},$$

IA m chooses the input rate

$$f_m(\mathbf{q}(t)) := \arg\max_{0 \le f \le 1} \{V \cdot u_m(f) + \mathbf{q}(t)^T A_m f\} \tag{12.16}$$

where $V > 0$ is a constant, and A_m is the m'th column of A. Then, update the virtual queues as

$$\mathbf{q}(t+1) = \mathbf{q}(t) - A \cdot \mathbf{f}(\mathbf{q}(t)) - B \cdot \mathbf{x}^*(t) \tag{12.17}$$

Since $f_m(\mathbf{q}(t))$ in general is not integer, we let $a_m(t) = \lfloor F_m(t+1) \rfloor - \lfloor F_m(t) \rfloor$, where $F_m(t) := \sum_{\tau=0}^{t-1} f_m(\mathbf{q}(\tau))$. And update the actual queues in the same way as (12.6).

Theorem 12. *With the above algorithm, $\mathbf{q}(t)$ and $\mathbf{Q}(t)$ are bounded. Also, there are at most a finite number of null activities which do not affect the long term throughput.*

The proof is in section 12.9.3.

The following is a performance bound of the utility maximization algorithm. The proof is similar to that in [85], and is given in section 12.9.4.

Theorem 13. *We have*

$$\sum_m u_m(\tilde{f}_m) \geq U^* - c/V \tag{12.18}$$

where $\tilde{f}_m := \liminf_{T\to\infty} \sum_{t=0}^{T-1} f_m(\mathbf{q}(t))$, U^* is the optimal total utility, and $c > 0$ is a constant defined in (12.20). That is, a larger V leads to better a lower bound of the achieved utility (at the cost of larger queue lengths).

12.6 Extensions

In the above, we have assumed that each activity lasts one slot for the ease of exposition. Our algorithms can be extended to the case where different activities have different durations under a particular assumption. The assumption is that each activity can be suspended in the middle and resumed later. If so, we can still use the above algorithm which re-computes the maximum weight schedule in each time slot. The only difference is that the activities performed in one time slot may not be completed at the end of the slot, but are suspended and to be continued in later slots. (The above assumption was also made in the "preempted" networks in [80]. There, whenever a new schedule is computed, the ongoing activities are suspended, or "preempted".)

In this case, the algorithms are adapted in the following way. The basic idea is the same as before. That is, we run the system according to the virtual queues $\mathbf{q}(t)$. Let the elements in matrices A and B be the average rates of consuming (or producing) parts per slot from (or to) different queues. Even if an activity is not completed in one slot, we still update the virtual queues $\mathbf{q}(t)$ according to the above average rates. That is, we view the parts in different queues as fluid and $\mathbf{q}(t)$ reflects the amount of fluid at each queue. However, only when an activity is completed, the actual parts are removed from or added to the output queues. Note that when an activity is suspended, all parts involved in the activity are frozen and are not available to other activities. When there are not enough parts in the queues to perform a scheduled activity, fictitious parts are used instead (and the corresponding deficits are increased).

On the other hand, if each activity cannot be suspended in the middle once it is started, then one possible scheme is to use long time slots in our algorithms. In slot t, each SA n with $x_n^*(t) = 1$ is activated as many times as possible. When each slot is very long, the wasted time during the slot becomes negligible, so the algorithm approximates the maximal throughput (with the cost of longer delay). Without using long slots, the non-preemptive version of the maximal-pressure algorithm proposed in [80] is not throughput-optimum in general, but it IS under a certain resource-sharing constraints [80].

12.7 Simulations

12.7.1 DMW scheduling

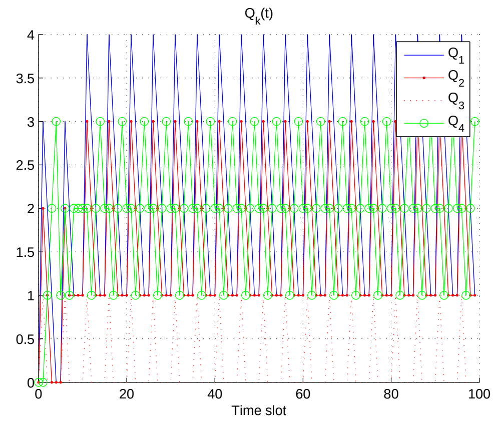
We simulate a network similar to Fig. 12.1, but with a different input matrix A and service matrix B below.

$$A = \begin{bmatrix} -3 \\ -2 \\ -1 \\ 0 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 2 & 1 \end{bmatrix}$$

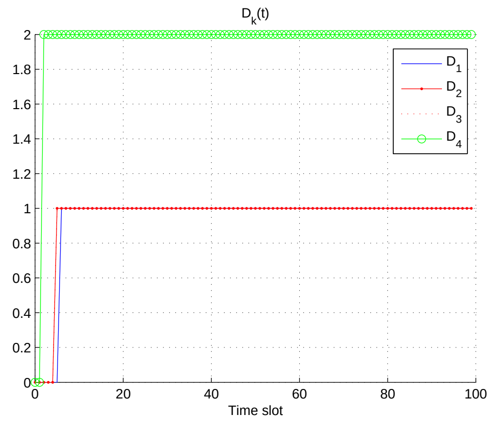
It is easy to check that if $\lambda_1 = 1/3$, we have $A \cdot \lambda_1 + B \cdot \mathbf{s} = \mathbf{0}$ where $\mathbf{s} := [1, 1/3, 1/3, 1/3]^T \in \Lambda$ (and $\mathbf{s}$ is unique). So, any $\lambda_1 \in (0, 1/3)$ is strictly feasible.

In the simulation, IA1 is activated in slot $5k, k = 0, 1, 2, \ldots,$. So the input rate $\lambda_1 = 1/5$ which is strictly feasible. Since SA 3 requires enough parts from several queues to perform, it is not difficult to see that normal MWS fails to stabilize queue 3. Fig. 12.4 shows that DMW stabilizes all queues and have bounded deficits.

Now we make a change to the arrival process. In time slot $4k, k = 0, 1, 2 \ldots$, IA 1 is independently activated with probability 0.8. As a result, the expected arrival rate is strictly feasible and also satisfies the smoothness condition (Condition 2) with $T = 4$. Fig. 12.5 shows that our algorithm stabilizes all queues. As expected, $D_k(t)$ stops increasing after some time since $\mathbf{q}(t)$ is bounded.

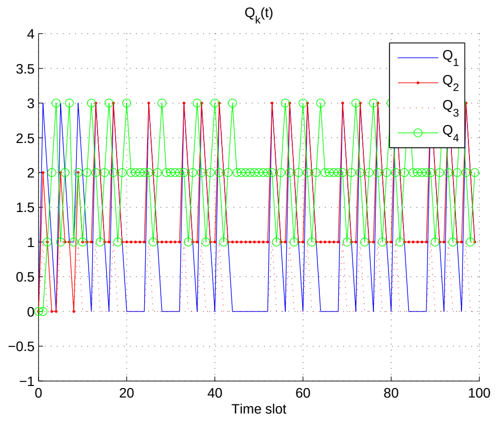


(a) Actual queue lengths $\mathbf{Q}(t)$

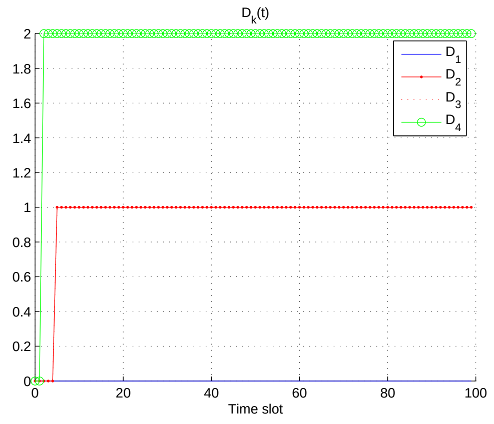


(b) Deficits $\mathbf{D}(t)$

Figure 12.4: DMW scheduling (with deterministic arrivals)



(a) Actual queue lengths $\mathbf{Q}(t)$

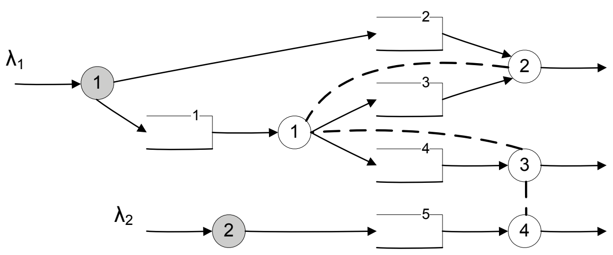


(b) Deficits $\mathbf{D}(t)$

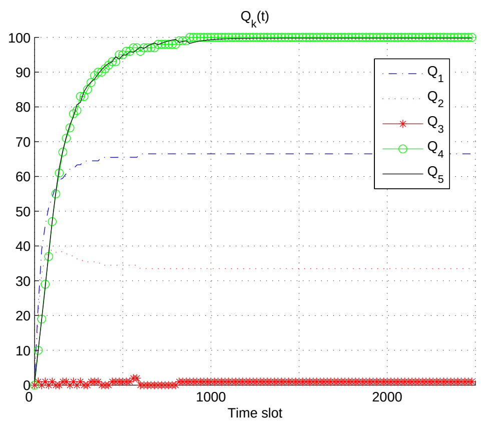
Figure 12.5: DMW Scheduling (with random arrivals)

12.7.2 Utility maximization

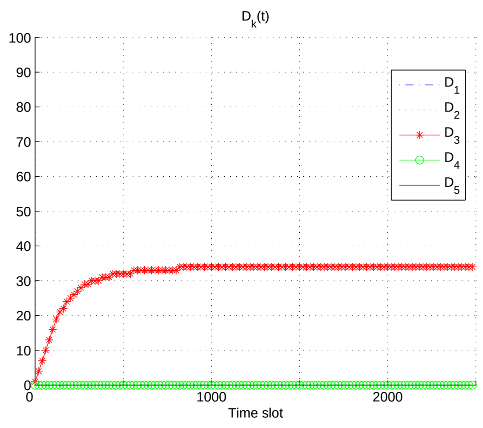
Consider the network in Fig. 12.6 (a). The utility functions of both flows are $u_m(\cdot) = \log(\cdot)$. We simulate the network with the utility maximization algorithm with $V = 50$. Fig. 12.6 (b) shows $Q_k(t)$ and $D_k(t)$. As expected, $D_k(t)$ stops increasing after some time due to the boundedness of $q_k(t)$. The average throughput of flow 1 and 2 is 0.4998 and 0.4998, which are very close to the theoretical optimal throughput computed by solving the utility maximization problem numerically. (To double-check the correctness of the algorithm, note that for example $q_1(t) + q_2(t) = [Q_1(t) - D_1(t)] + [Q_2(t) - D_2(t)]$ is about 100 after initial convergence. So by (12.16), $f_1(\mathbf{q}(t)) \approx V/[q_1(t) + q_2(t)] = 0.5$.)



(a) Network



(b) Actual queue lengths $\mathbf{Q}(t)$



(c) Deficits $\mathbf{D}(t)$

Figure 12.6: Utility Maximization

12.8 Summary and future work

We have proposed a "Deficit Maximum Weight" (DMW) scheduling algorithm for SPNs. It has been shown that the algorithm overcomes the instability issue of the Maximum Weight algorithm that arises in general SPNs. Under different assumptions of the arrival processes, DMW achieves the maximum throughput in different senses. We have also combined DMW scheduling with input rate control in order to approach the maximum utility of the SPN.

The performance of DMW could be further improved. For example, with null activities, a queue may receive fictitious parts in some slots and drop normal parts in other slots. So, one could store the dropped parts and replace the fictitious parts later when they arrive. The performance improvement with such schemes is interesting for future research.

The current algorithm achieves bounded queues when the arrival processes are smooth enough, and guarantees rate stability otherwise. Rate stability is a weaker form of stability than positive recurrence of the queues. We are also interested to enhance the current algorithm or design new algorithms to achieve positive recurrence under such arrival processes.

12.9 Skipped proofs

12.9.1 Proof of Theorem 10

To analyze the queue dynamics, consider the Lyapunov function $L(\mathbf{q}(t)) = ||\mathbf{q}(t)||^2$. We have

$$\begin{aligned} & \Delta(\mathbf{q}(t)) := L(\mathbf{q}(t+1)) - L(\mathbf{q}(t)) \\ = \; & ||\mathbf{q}(t) - A \cdot \mathbf{a}(t) - B \cdot \mathbf{x}^*(\mathbf{q}(t))||^2 - ||\mathbf{q}(t)||^2 \\ = \; & -\mathbf{q}(t)^T A \cdot \mathbf{a}(t) - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) \\ & + ||A \cdot \mathbf{a}(t) + B \cdot \mathbf{x}^*(\mathbf{q}(t))||^2 \\ \leq \; & -\mathbf{q}(t)^T A \cdot \mathbf{a}(t) - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c \qquad (12.19) \end{aligned}$$

where $c > 0$ is a constant, defined as

$$c := \sum_k (\mu_{k,in}^2 + \mu_{k,out}^2) \tag{12.20}$$

where $\mu_{k,in}$, $\mu_{k,out}$ are, respectively, the maximum amount of parts that can enter or leave queue k in one time slot.

Lemma 21. *Assume that* $\mathbf{q}(0) = \mathbf{0}$. *If for any* t,

$$L(\mathbf{q}(t+1)) - L(\mathbf{q}(t)) \leq -\delta ||\mathbf{q}(t)|| + c \tag{12.21}$$

where $\delta > 0$ *is a constant, then* $\mathbf{q}(t)$ *is always bounded. In particular,* $L(\mathbf{q}(t)) \leq c^2/\delta^2 + c$.

Proof. We prove this through the principle of mathematical induction. First, $L(\mathbf{q}(0)) = 0 \leq c^2/\delta^2 + c$.

Next, as the induction hypothesis, assume that $L(\mathbf{q}(t)) \leq c^2/\delta^2 + c$. Consider two cases. (i) If $L(\mathbf{q}(t)) \in [c^2/\delta^2, c^2/\delta^2 + c]$, then $||\mathbf{q}(t)|| \geq c/\delta$. By (12.21), we have $L(\mathbf{q}(t+1)) \leq L(\mathbf{q}(t)) \leq c^2/\delta^2 + c$. (ii) If $L(\mathbf{q}(t)) < c^2/\delta^2$, since $L(\mathbf{q}(t+1)) - L(\mathbf{q}(t)) \leq -\delta||\mathbf{q}(t)|| + c \leq c$, we also have $L(\mathbf{q}(t+1)) \leq c^2/\delta^2 + c$. This completes the proof. $\square$

Lemma 22. *Assume that condition 1 holds. Let* $\mathbf{y}(l \cdot T)$ *be the (unique) vector that satisfies*

$$A \cdot \tilde{\mathbf{a}}_l + B \cdot \mathbf{y}(l \cdot T) \quad = \quad \mathbf{0} \tag{12.22}$$

where $\tilde{\mathbf{a}}_l$ *is defined in (12.14). Then there exists* $\bar{\delta} > 0$ *such that*

$$q^T B \cdot [\mathbf{x}^*(\mathbf{q}) - \mathbf{y}(l \cdot T)] \geq \bar{\delta} ||\mathbf{q}||, \forall l, \forall \mathbf{q} \in \mathcal{B} \tag{12.23}$$

where $\mathbf{x}^*(\mathbf{q})$ *is defined in (12.12).*

Proof. By Condition 1, $\exists \sigma > 0$ such that for all l, $\tilde{\mathbf{a}}_l + \sigma \cdot \mathbf{1}$ and $\tilde{\mathbf{a}}_l - \sigma \cdot \mathbf{1}$ are feasible. Therefore, $\mathbf{y}(l \cdot T) + \mathbf{s}(\sigma \cdot \mathbf{1}_M) \in \Lambda$ and $\mathbf{y}(l \cdot T) - \mathbf{s}(\sigma \cdot \mathbf{1}_M) \in \Lambda$. Define $\sigma' > 0$ to be the minimum element of $\mathbf{s}(\sigma \cdot \mathbf{1}_M) \succ \mathbf{0}$, then $\mathbf{y}' \in \Lambda$ for any $\mathbf{y}'$ satisfying $||\mathbf{y}' - \mathbf{y}(l \cdot T)|| \leq \sigma'$. (This is because the set Λ is "comprehensive": if $\mathbf{s} \in \Lambda$, then $\mathbf{s}' \in \Lambda$ for any $\mathbf{0} \preceq \mathbf{s}' \preceq \mathbf{s}$.) Then, following the proof of Lemma 20, letting $\bar{\delta} := \hat{\sigma} \cdot \sigma'$ (which do not depend on l or $\mathbf{q}$) completes the proof. $\square$

Lemma 23. *Assume that the maximum change of any queue in one time slot is bounded by α. And the absolute value of every element of A and B is bounded by $\bar{b}$. Then*

$$\begin{aligned} & L(\mathbf{q}((l+1)T)) - L(\mathbf{q}(l \cdot T)) \\ \leq & -T \cdot \bar{\delta}||\mathbf{q}(l \cdot T)|| + c_2 \end{aligned}$$

where $c_2 > 0$ is a constant, defined as

$$c_2 := T \cdot c + KT^2\alpha \cdot (M+K)\bar{b}.$$

Proof. From (12.19), we have

$$\begin{aligned} & L(\mathbf{q}((l+1)T)) - L(\mathbf{q}(l \cdot T)) \\ \leq & -\textstyle\sum_{\tau=l\cdot T}^{(l+1)T-1} \mathbf{q}(\tau)^T A \cdot \mathbf{a}(\tau) \\ & -\textstyle\sum_{\tau=l\cdot T}^{(l+1)T-1} q(\tau)^T B \cdot \mathbf{x}^*(\mathbf{q}(\tau)) + T \cdot c. \end{aligned}$$

For any $\tau \in \{l \cdot T, \ldots, (l+1)T - 1\}$,

$$\begin{aligned} & \mathbf{q}(\tau)^T B \cdot \mathbf{x}^*(\mathbf{q}(\tau)) \\ \geq & \mathbf{q}(\tau)^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T)) \\ = & \mathbf{q}(l \cdot T)^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T)) + \\ & [\mathbf{q}(\tau) - \mathbf{q}(l \cdot T)]^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T)). \end{aligned}$$

Since $|q_k(\tau) - q_k(l \cdot T)| \leq T \cdot \alpha$, and each element of $\mathbf{x}^*(\mathbf{q}(l \cdot T))$ is bounded by 1, we have

$$|[\mathbf{q}(\tau) - \mathbf{q}(l \cdot T)]^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T))| \leq KN\bar{b}T\alpha.$$

Therefore,

$$\begin{aligned} & \mathbf{q}(\tau)^T B \cdot \mathbf{x}^*(\mathbf{q}(\tau)) \\ \geq & \mathbf{q}(l \cdot T)^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T)) - KN\bar{b}T\alpha. \end{aligned} \qquad (12.24)$$

Also, $\mathbf{q}(\tau)^T A \cdot \mathbf{a}(\tau) \geq \mathbf{q}(l \cdot T)^T A \cdot \mathbf{a}(\tau) - KM\bar{b}T\alpha$. Then

$$\begin{aligned}
& L(\mathbf{q}((l+1)T)) - L(\mathbf{q}(l \cdot T)) \\
\leq \;& T \cdot \{-\mathbf{q}(l \cdot T)^T A \cdot \tilde{\mathbf{a}}_l + KM\bar{b}T\alpha \\
& -\mathbf{q}(l \cdot T)^T B \cdot \mathbf{x}^*(\mathbf{q}(l \cdot T)) + KN\bar{b}T\alpha\} + T \cdot c \\
= \;& -T \cdot \mathbf{q}(l \cdot T)^T B \cdot [\mathbf{x}^*(\mathbf{q}(l \cdot T)) - \mathbf{y}(l \cdot T)] + c_2 \\
\leq \;& -T \cdot \bar{\delta}||\mathbf{q}(l \cdot T)|| + c_2
\end{aligned}$$

where the last two steps have used (12.22) and condition (12.23). □

Now Theorem 10 can be proved as follows.

Proof. Lemma 23 and Lemma 21 imply that $\mathbf{q}(l \cdot T)$ is bounded for all l. Because each queue has bounded increments per slot, $\mathbf{q}(t)$ is bounded for all t. □

12.9.2 Proof of Theorem 11

By (12.19), $L(\mathbf{q}(t+1)) - L(\mathbf{q}(t)) \leq -\mathbf{q}(t)^T A \cdot \mathbf{a}(t) - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c$. So

$$\begin{aligned}
& E[L(\mathbf{q}(t+1)) - L(\mathbf{q}(t))|\mathbf{q}(t)] \\
\leq \;& -\mathbf{q}(t)^T A \cdot E[\mathbf{a}(t)] - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c \\
= \;& -\mathbf{q}(t)^T A \cdot \lambda - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c \\
= \;& \mathbf{q}(t)^T B \cdot \mathbf{y} - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c \\
\leq \;& -\delta||\mathbf{q}(t)|| + c. \qquad (12.25)
\end{aligned}$$

Let $\mathcal{E}_0 := \{\mathbf{q}(t)| \, ||\mathbf{q}(t)|| \leq (c+1)/\delta$. Then if $\mathbf{q}(t) \notin \mathcal{E}_0$, $E[L(\mathbf{q}(t+1)) - L(\mathbf{q}(t))|\mathbf{q}(t)] \leq -1$; if $\mathbf{q}(t) \in \mathcal{E}_0$, $E[L(\mathbf{q}(t+1)) - L(\mathbf{q}(t))|\mathbf{q}(t)] < \infty$ due to the bounded change of queue lengths in each slot. Therefore, by Foster's criteria as used in [47], $\mathbf{q}(t)$ is stable.

Also, we claim that given a set $\mathcal{E}$, with probability 1, the time average $P(\mathcal{E}) := \lim_{T\to\infty} \sum_{t=0}^{T-1} I(\mathbf{q}(t) \in \mathcal{E})/T$ exists. To see this, partition the state space of $\mathbf{q}(t)$ into set $\mathcal{T}, \mathcal{R}_1, \mathcal{R}_2, \ldots$ where $\mathcal{R}_j, j = 1, 2, \ldots$ are closed sets of communicating states and $\mathcal{T}$ is the set of states not in $\cup_j \mathcal{R}_j$. If $\mathbf{q}(0) = \mathbf{0} \in \mathcal{R}_j$ for some j, then $\mathbf{q}(t)$ will not leave the set and all states in $\mathcal{R}_j$ are positive recurrent. Therefore there is a well defined stationary distribution in $\mathcal{R}_j$, so $P(\mathcal{E})$ exists w. p. 1. If $\mathbf{q}(0) = \mathbf{0} \in \mathcal{T}$, by Foster's criteria as used in [47] (Theorem 3.1), the negative drift implies that w. p. 1, $\mathbf{q}(t)$ enters some $\mathcal{R}_j$ in finite time. After that there is a well defined time average of $I(\mathbf{q}(t) \in \mathcal{E})$ w. p. 1. Therefore, the overall time average $P(\mathcal{E})$ exists. In both cases,

$$P(\mathcal{E}) = \pi_j(\mathcal{E}) \tag{12.26}$$

where $\pi_j(\cdot)$ is the stationary distribution on the $\mathcal{R}_j$, and $\mathcal{R}_j$ is the closed set of communicating states $\mathbf{q}(t)$ eventually enters.

To show the rate stability, consider two kinds of queues. WLOG, let $\mathcal{U}$ be the set of queues whose deficits go unbounded. According to Proposition 27, the queues outside the set only induce a finite number of null activities.

Consider queue $k \in \mathcal{U}$. For any $C > 0$, since $D_k(t) \to \infty$, there exists finite time t_k such that $D_k(t) \geq C, \forall t \geq t_k$. For $t \geq t_k$, queue k induces null activities at slot $t-1$ only when $q_k(t) < -D_k(t) \leq -C$. So the total number of null activities induced by queue k is not more than $N \cdot [t_k + \sum_{t=t_k}^{\infty} I(q_k(t) < -C)] \leq N \cdot [t_k + \sum_{t=0}^{\infty} I(q_k(t) < -C)]$, since queue k at most induces N null activities in one time slot. Therefore, the average rate the queue k induces null activities is

$$r_k \leq N \cdot \lim_{T \to \infty} \frac{1}{T}[t_k + \sum_{t=0}^{T-1} I(q_k(t) < -C)] = N \cdot Pr(q_k < -C). \tag{12.27}$$

where the marginal probability on the RHS is induced by the stationary distribution $\pi_j(\cdot)$ on the set R_j which $\mathbf{q}(t)$ eventually enters. So $\lim_{C \to +\infty} Pr(q_k < -C) = 0$. Since (12.27) holds for any $C > 0$, letting $C \to +\infty$ yields $r_k = 0$.

Therefore, the average rate of null activities is 0 in the long term w. p. 1. Also, if we imagine that the null activities produce real parts, then the output rates of the final products would be the maximum since the virtual queues $\mathbf{q}(t)$ are stable. Combining the two facts concludes the proof.

12.9.3 Proof of Theorem 12

Lemma 24. $\mathbf{q}(t)$ *is bounded.*

Proof. Choose any $\mathbf{f}' \in \mathcal{R}^m$ and $\mathbf{y}' > \mathbf{0}$ in Λ^o such that the flow conservation constraint is satisfied: $A \cdot \mathbf{f}' + B \cdot \mathbf{y}' = \mathbf{0}$, and $|\sum_m u_m(f'_m)| < \infty, \forall m$. The latter is feasible by letting $f'_m = \epsilon > 0, \forall m$ where ϵ is small enough.

By Lemma 20, we have for any $\mathbf{q} \in \mathcal{B}$,

$$q^T B \cdot [\mathbf{x}^*(\mathbf{q}) - \mathbf{y}'] \geq \delta' ||\mathbf{q}|| \tag{12.28}$$

for some constant $\delta' > 0$.

Also, since $f'_m \in [0,1]$, by (12.16),

$$\begin{aligned} & V \cdot u_m(f'_m) + \mathbf{q}(t)^T A_m \cdot f'_m \\ \leq \; & V \cdot u_m(f_m(\mathbf{q}(t))) + \mathbf{q}(t)^T A_m f_m(\mathbf{q}(t)), \forall m. \end{aligned}$$

Therefore

$$\begin{aligned} & V \cdot \textstyle\sum_{m=1}^M u_m(f'_m) + \mathbf{q}(t)^T A \cdot \mathbf{f}' \\ \leq \; & V \cdot \textstyle\sum_{m=1}^M u_m(f_m(\mathbf{q}(t))) + \mathbf{q}(t)^T A \cdot \mathbf{f}(\mathbf{q}(t)). \end{aligned}$$

Since $|\sum_m u_m(f'_m)| < \infty$, we have $\sum_{m=1}^M u_m(f_m(\mathbf{q}(t))) - \sum_{m=1}^M u_m(f'_m) \leq \sum_{m=1}^M u_m(1) - \sum_{m=1}^M u_m(f'_m) \leq C_1$ for some positive constant C_1. So

$$-\mathbf{q}(t)^T A \cdot \mathbf{f}(\mathbf{q}(t)) \leq -\mathbf{q}(t)^T A \cdot \mathbf{f}' + V \cdot C_1. \tag{12.29}$$

Similar to (12.19), the Lyapunov drift in the algorithm is

$$\Delta(\mathbf{q}(t)) \leq -\mathbf{q}(t)^T A \cdot \mathbf{f}(\mathbf{q}(t)) - q(t)^T B \cdot \mathbf{x}^*(\mathbf{q}(t)) + c. \tag{12.30}$$

Plugging (12.28) and (12.29) into (12.30) yields

$$\begin{aligned}
&\Delta(\mathbf{q}(t))\\
\leq\ & -\mathbf{q}(t)^T A\cdot\mathbf{f}' + V\cdot C_1 - q(t)^T B\cdot\mathbf{y}' - \delta'||\mathbf{q}(t)|| + c\\
=\ & -\mathbf{q}(t)^T[A\cdot\mathbf{f}' + B\cdot\mathbf{y}'] - \delta'||\mathbf{q}(t)|| + V\cdot C_1 + c\\
=\ & -\delta'||\mathbf{q}(t)|| + V\cdot C_1 + c.
\end{aligned}$$

Using Lemma 21, the above implies that for all t,

$$L(\mathbf{q}(t)) \leq [(V\cdot C_1 + c)/\delta']^2 + V\cdot C_1 + c.$$

So $\mathbf{q}(t)$ is bounded. $\square$

Define $\tilde{\mathbf{q}}(0) = \mathbf{0}$, and for $t = 0, 1, \ldots$, define

$$\tilde{\mathbf{q}}(t+1) = \tilde{\mathbf{q}}(t) - A\cdot\mathbf{a}(t) - B\cdot\mathbf{x}^*(t). \tag{12.31}$$

Lemma 25. *For all t, $||\tilde{\mathbf{q}}(t) - \mathbf{q}(t)|| \leq Z$ for some constant $Z > 0$.*

Proof. By (12.17) and $\mathbf{q}(0) = \mathbf{0}$, we have

$$\begin{aligned}
\mathbf{q}(t) &= \textstyle\sum_{\tau=0}^{t-1}[-A\cdot\mathbf{f}(\mathbf{q}(\tau)) - B\cdot\mathbf{x}^*(\tau)]\\
&= -A\textstyle\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau)) - B\cdot\sum_{\tau=0}^{t-1}\mathbf{x}^*(\tau).
\end{aligned}$$

By (12.31) and $\tilde{\mathbf{q}}(0) = \mathbf{0}$, we have

$$\begin{aligned}
\tilde{\mathbf{q}}(t) &= \textstyle\sum_{\tau=0}^{t-1}[-A\cdot\mathbf{a}(\tau) - B\cdot\mathbf{x}^*(\tau)]\\
&= -A\lfloor\textstyle\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau))\rfloor - B\cdot\sum_{\tau=0}^{t-1}\mathbf{x}^*(\tau).
\end{aligned}$$

So, $||\tilde{\mathbf{q}}(t) - \mathbf{q}(t)|| = ||A\cdot\{\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau)) - \lfloor\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau))\rfloor\}||$. Since each element of $\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau)) - \lfloor\sum_{\tau=0}^{t-1}\mathbf{f}(\mathbf{q}(\tau))\rfloor$ is between 0 and 1, and each element of A is bounded, we conclude that $||\tilde{\mathbf{q}}(t) - \mathbf{q}(t)|| \leq Z$ for some constant $Z > 0$. $\square$

Now we are ready to complete the proof.

Since $||\tilde{\mathbf{q}}(t)|| \leq ||\mathbf{q}(t)|| + ||\tilde{\mathbf{q}}(t) - \mathbf{q}(t)||$, combining the previous two lemmas, we know that $||\tilde{\mathbf{q}}(t)|| \leq G, \forall t$ for some $G > 0$. Define $\mathbf{D}(t) = \mathbf{Q}(t) - \tilde{\mathbf{q}}(t)$. Comparing the dynamics of $\mathbf{Q}(t)$ and $\tilde{\mathbf{q}}(t)$, it is clear that we can apply Proposition 27 to $\tilde{\mathbf{q}}(t), \mathbf{Q}(t)$ and $D(t)$ to complete the proof.

12.9.4 Proof of Theorem 13

Proof. Assume that $\mathbf{f}^* \in \mathcal{R}^m$ and $\mathbf{y}^* > \mathbf{0}$ achieves the optimal utility U^*. So $A \cdot \mathbf{f}^* + B \cdot \mathbf{y}^* = \mathbf{0}$ and $U^* = \sum_m u_m(f_m^*)$.

We also have $q^T B \cdot [\mathbf{x}^*(\mathbf{q}) - \mathbf{y}^*] \geq 0$. This is equivalent to (12.28) when $\delta' = 0$. Then, following the proof of Theorem 12 (but without using the upper bound C_1), we have

$$\begin{aligned} \Delta(\mathbf{q}(t)) &\leq -\mathbf{q}(t)^T [A \cdot \mathbf{f}^* + B \cdot \mathbf{y}^*] + \\ & \quad V \cdot [\sum_m u_m(f_m(\mathbf{q}(t))) - \sum_m u_m(f_m^*)] + c \\ &= V \cdot [\sum_m u_m(f_m(\mathbf{q}(t))) - U^*] + c. \end{aligned}$$

Summing over t from 0 to $T-1$ yields

$$L(\mathbf{q}(T)) - L(\mathbf{q}(0)) \leq V \cdot \sum_{t=0}^{T-1} \sum_m u_m(f_m(\mathbf{q}(t))) - VTU^* + T \cdot c.$$

Dividing both sides by $T \cdot V$, and using $L(\mathbf{q}(T)) - L(\mathbf{q}(0)) = L(\mathbf{q}(T)) \geq 0$, one gets

$$\sum_{t=0}^{T-1} \sum_m u_m(f_m(\mathbf{q}(t)))/T \geq U^* - c/V. \tag{12.32}$$

Since $u_m(\cdot)$ is concave, $u_m(\sum_{t=0}^{T-1} f_m(\mathbf{q}(t))/T) \geq \sum_{t=0}^{T-1} u_m(f_m(\mathbf{q}(t)))/T$. Using this, (12.32) and letting $T \to \infty$, we have (12.18). $\square$

Bibliography

[1] S. H. Low and P. P. Varaiya, “A New Approach to Service Provisioning in ATM Networks,” *IEEE Transactions on Networking*, vol. 1, no. 5, pp. 549-553, Oct. 1993.

[2] F. P. Kelly, “Charging and Rate Control for Elastic Traffic,” *European Transactions on Telecommunications*, vol. 8, pp. 33-37, 1997.

[3] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, “Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures,” in *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255-312, 2007.

[4] J. Nash, “Non-Cooperative Games,” *The Annals of Mathematics*, vol. 54, no. 2, pp. 286-295, 1951.

[5] ”Malware”, http://en.wikipedia.org/wiki/Malware

[6] L. A. Gordon, and M. P. Loeb, “The Economics of Information Security Investment,” *ACM Transactions on Information and System Security*, vol. 5, no. 4, pp. 438-457, Nov. 2002.

[7] H. R. Varian, “System Reliability and Free Riding”, in *Workshop on Economics and Information Security*, 2002.

[8] E. Koutsoupias and C. H. Papadimitriou, “Worst-Case Equilibria,” in *Annual Symposium on Theoretical Aspects of Computer Science*, 1999.

[9] T. Roughgarden and Tardos, ”How Bad is Selfish Routing”, *Journal of the ACM*, 2002.

[10] T. Roughgarden, "The Price of Anarchy is Independent of the Network Topology", in *the thiry-fourth annual ACM symposium on Theory of computing*, 2002, pp. 428 - 437.

[11] D. Acemoglu and A. Ozdaglar, "Competition and Efficiency in Congested Markets", *Mathematics of Operations Research*, vol. 32, no. 1, pp. 1-31, Feb. 2007.

[12] A. Ozdaglar, "Price Competition with Elastic Traffic", *LIDS report*, MIT, 2006.

[13] R. Johari and J.N. Tsitsiklis, "Efficiency Loss in a Network Resource Allocation Game", *Mathematics of Operations Research*, 29(3): 407-435, 2004.

[14] J. Aspnes, K. Chang, and A. Yampolskiy, "Inoculation Strategies for Victims of Viruses and the Sum-of-Squares Partition Problem", in *the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, 2005, pp. 43-52.

[15] M. Lelarge and J. Bolot, "A Local Mean Field Analysis of Security Investments in Networks," in *the 3rd International Workshop on Economics of Networked Systems (NetEcon)*, Aug. 2009.

[16] W. Liu, H. Tanaka, and K. Matsuura, "An Empirical Analysis of Security Investment in Countermeasures Based on an Enterprise Survey in Japan," in *the fifth Workshop on the Economics of Information Security* (*WEIS 2006*), June 2006.

[17] D. Fudenberg and J. Tirole, "*Game Theory*", MIT Press, Cambridge, 1991.

[18] R. J. Aumann, "Subjectivity and Correlation in Randomized Strategies," *Journal of Mathematical Economics*, vol. 1, pp. 67-96, 1974.

[19] R. B. Myerson, "Dual Reduction and Elementary Games," *Games and Economic Behavior*, vol. 21, no. 1-2, pp. 183-202, 1997.

[20] J. B. Rosen, "Existence and Uniqueness of Equilibrium Points for Concave N-Person Games," *Econometrica*, vol. 33, no. 3, pp. 520-534, Jul. 1965.

[21] UC Berkeley, "Minimum Standards for Security of Berkeley Campus Networked Devices", https://security.berkeley.edu/MinStds/AppA.min.htm

[22] L. Jiang, V. Anantharam, and J. Walrand, "Efficiency of Selfish Investment in Network Security", in *Workshop on the Economics of Networks, Systems, and Computation (NetEcon'08), ACM SIGCOMM 2008*, Seattle, WA, Aug. 2008.

[23] L. Jiang, V. Anantharam, and J. Walrand, "How Bad are Selfish Investments in Network Security?" submitted to *IEEE/ACM Transactions on Networking.*

[24] X. Lin, N. B. Shroff, and R. Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp.1452-1463, Aug. 2006.

[25] J. M. Harrison and R. J. Williams, "Workload Interpretation for Brownian Models of Stochastic Processing Networks," *Mathematics of Operations Research*, vol. 32, pp. 808-820, 2007.

[26] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," in *the 46th Annual Allerton Conference on Communication, Control, and Computing*, Sep. 23-26, 2008.

[27] L. Jiang and J. Walrand, "A Distributed Algorithm for Maximal Throughput and Optimal Fairness in Wireless Networks with a General Interference Model," EECS Technical Report, UC Berkeley, Apr. 2008. http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-38.html

[28] A. Eryilmaz, A. Ozdaglar and E. Modiano, "Polynomial Complexity Algorithms for Full Utilization of Multi-hop Wireless Networks," in *IEEE INFOCOM 2007*, Anchorage, Alaska, May 2007.

[29] E. Modiano, D. Shah, and G. Zussman, "Maximizing Throughput in Wireless Networks via Gossiping," *ACM SIGMETRICS Performance Evaluation Review*, vol. 34 , no. 1, Jun. 2006.

[30] S. Sanghavi, L. Bui, and R. Srikant, "Distributed Link Scheduling with Constant Overhead," in *ACM SIGMETRICS*, Jun. 2007.

[31] J. W. Lee, M. Chiang, and R. A. Calderbank, “Utility-Optimal Random-Access Control,” *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2741-2751, Jul. 2007.

[32] P. Gupta and A. L. Stolyar, “Optimal Throughput Allocation in General Random Access Networks,” in *Conference on Information Sciences and Systems*, Princeton, NJ, Mar. 2006.

[33] P. Chaporkar, K. Kar, and S. Sarkar, “Throughput guarantees in maximal scheduling in wireless networks,” in *the 43rd Annual Allerton Conference on Communication, Control and Computing*, Sep. 2005.

[34] X. Wu and R. Srikant, “Scheduling Efficiency of Distributed Greedy Scheduling Algorithms in Wireless Networks,” in *IEEE INFOCOM 2006*, Barcelona, Spain, Apr. 2006.

[35] A. Dimakis and J. Walrand, “Sufficient Conditions for Stability of Longest-Queue-First Scheduling: Second-Order Properties Using Fluid Limits,” *Advances in Applied Probability*, vol. 38, no. 2, pp. 505–521, 2006.

[36] C. Joo, X. Lin, and N. Shroff, “Understanding the Capacity Region of the Greedy Maximal Scheduling Algorithm in Multi-Hop Wireless Networks,” in *IEEE INFOCOM 2008*, Phoenix, Arizona, Apr. 2008.

[37] G. Zussman, A. Brzezinski, and E. Modiano, “Multihop Local Pooling for Distributed Throughput Maximization in Wireless Networks,” in *IEEE INFOCOM 2008*, Phoenix, Arizona, Apr. 2008.

[38] M. Leconte, J. Ni, and R. Srikant, “Improved Bounds on the Throughput Efficiency of Greedy Maximal Scheduling in Wireless Networks,” in *ACM MOBIHOC*, May 2009.

[39] X. Lin and N. Shroff, “The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Multihop Wireless Networks,” in *IEEE INFOCOM 2005*, Miami, Florida, Mar. 2005.

[40] P. Marbach, A. Eryilmaz, and A. Ozdaglar, "Achievable Rate Region of CSMA Schedulers in Wireless Networks with Primary Interference Constraints," in *IEEE Conference on Decision and Control*, 2007.

[41] A. Proutiere, Y. Yi, and M. Chiang, "Throughput of Random Access without Message Passing," in *Conference on Information Sciences and Systems*, Princeton, NJ, USA, Mar. 2008.

[42] S. Rajagopalan and D. Shah, "Distributed Algorithm and Reversible Network", in *Conference on Information Sciences and Systems*, Princeton, NJ, USA, Mar. 2008.

[43] Y. Xi and E. M. Yeh, "Throughput Optimal Distributed Control of Stochastic Wireless Networks," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2006.

[44] M. J. Neely, E. Modiano, and C. P. Li, "Fairness and Optimal Stochastic Control for Heterogeneous Networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396-409, Apr. 2008.

[45] B. Hajek, "Cooling Schedules for Optimal Annealing," *Mathematics of Operations Research*, vol. 13, no. 2, pp. 311–329, 1988.

[46] J. Liu, Y. Yi, A. Proutiere, M. Chiang, and H. V. Poor, "Convergence and Tradeoff of Utility-Optimal CSMA," http://arxiv.org/abs/0902.1996.

[47] L. Tassiulas and A. Ephremides, "Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.

[48] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% Throughput in an Input-Queued Switch," *IEEE Transactions on Communications*, vol. 47, no. 8, pp. 1260-1267, Aug. 1999.

[49] F. P. Kelly, *"Reversibility and Stochastic Networks*," Wiley, 1979.

[50] R. R. Boorstyn, A. Kershenbaum, B. Maglaris, and V. Sahin, "Throughput Analysis in Multihop CSMA Packet Radio Networks," *IEEE Transactions on Communications*, vol. 35, no. 3, pp. 267-274, Mar. 1987.

[51] X. Wang and K. Kar, "Throughput Modelling and Fairness Issues in CSMA/CA Based Ad-Hoc Networks," in *IEEE INFOCOM 2005*, Miami, Florida, Mar. 2005.

[52] S. C. Liew, C. Kai, J. Leung, and B. Wong, "Back-of-the-Envelope Computation of Throughput Distributions in CSMA Wireless Networks," in *IEEE ICC*, 2009.

[53] M. Durvy and P. Thiran, "Packing Approach to Compare Slotted and Non-Slotted Medium Access Control," in *IEEE INFOCOM 2006*, Barcelona, Spain, Apr. 2006.

[54] M. Durvy, O. Dousse, and P. Thiran, "Border Effects, Fairness, and Phase Transition in Large Wireless Networks", in *IEEE INFOCOM 2008*, Phoenix, Arizona, Apr. 2008.

[55] L. Jiang and S. C. Liew, "Improving Throughput and Fairness by Reducing Exposed and Hidden Nodes in 802.11 Networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 34-49, Jan. 2008.

[56] S. Boyd and L. Vandenberghe, "*Convex Optimization*", Cambridge University Press, 2004.

[57] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400-407, Sep. 1951.

[58] J. Zhang, D. Zheng, and M. Chiang, "The Impact of Stochastic Noisy Feedback on Distributed Network Utility Maximization," *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 645-665, Feb. 2008.

[59] M. J. Wainwright and M. I. Jordan, "Graphical Models, Exponential Families, and Variational Inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1-305, 2008.

[60] P. Whittle, "*Systems in Stochastic Equilibrium*," John Wiley & Sons, Inc., New York, NY, USA, 1986.

[61] J. Ni and R. Srikant, "Distributed CSMA/CA Algorithms for Achieving Maximum Throughput in Wireless Networks," in *Information Theory and Applications Workshop*, Feb. 2009.

[62] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-Length Based CSMA/CA Algorithms for Achieving Maximum Throughput and Low Delay in Wireless Networks," http://arxiv.org/pdf/0901.2333

[63] T. Ho and H. Viswanathan, "Dynamic Algorithms for Multicast with Intra-Session Network Coding," submitted to *IEEE Transactions on Information Theory*.

[64] R. Ahlswede, N. Cai, S. Li, and R.W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204-1216, Jul. 2000.

[65] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535-547, 2000.

[66] M. J. Neely and R. Urgaonkar, "Cross Layer Adaptive Control for Wireless Mesh Networks," *Ad Hoc Networks (Elsevier)*, vol. 5, no. 6, pp. 719-743, Aug. 2007.

[67] L. Jiang and J. Walrand, "Convergence and Stability of a Distributed CSMA Algorithm for Maximal Network Throughput," Technical Report, UC Berkeley, Mar. 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-43.html

[68] A. Warrier, S. Ha, P. Wason and I. Rhee, "DiffQ: Differential Backlog Congestion Control for Wireless Multi-hop Networks," Technical Report, Dept. Computer Science, North Carolina State University, 2008.

[69] Loc Bui, R. Srikant, and Alexander Stolyar, "Novel Architectures and Algorithms for Delay Reduction in Back-Pressure Scheduling and Routing," in *IEEE INFOCOM 2009 Mini-Conference.*

[70] P. Diaconis and D. Strook, "Geometric bounds for eigenvalues of Markov chains," *Annals of Applied Probability*, vol. 1, no. 1, pp. 36-61, Feb. 1991.

[71] L. Jiang and J. Walrand, "A Novel Approach to Model and Control the Throughput of CSMA/CA Wireless Networks", Technical Report, UC Berkeley, Jan 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-8.html

[72] L. Jiang and J. Walrand, "Approaching Throughput-Optimality in a Distributed CSMA Algorithm: Collisions and Stability," (invited), *ACM Mobihoc'09 S3 Workshop*, May 2009.

[73] L. Jiang and J. Walrand, "Approaching throughput-optimality in a Distributed CSMA Algorithm with Contention Resolution," Technical Report, UC berkeley, Mar. 2009. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-37.html

[74] V. Borkar, "*Stochastic Approximation: A Dynamical Systems Viewpoint*," 2008.

[75] R. J. Williams, "On Stochastic Processing Networks," Lecture Notes, 2006. `http://math.ucsd.edu/~williams/talks/belz/belznotes06.pdf`

[76] J. Walrand. *An Introduction to Queueing Networks*. Prentice Hall, 1988.

[77] J. M. Harrison, "Brownian Models of Open Processing Networks: Canonical Representation of Workload," *Annals of Applied Probability*, vol. 10, no. 1, pp. 75-103, 2000.

[78] J. M. Harrison and R. J. Williams, "Workload Reduction of a Generalized Brownian Network," *Annals of Applied Probability*, vol. 15, no. 4, pp. 2255-2295, 2005.

[79] J. G. Dai and W. Lin, "Asymptotic Optimality of Maximum Pressure Policies in Stochastic Processing Networks," *Annals of Applied Probability*, vol. 18, no. 6, pp. 2239-2299, 2008.

[80] J. G. Dai and W. Lin, "Maximum Pressure Policies in Stochastic Processing Networks," *Operations Research*, vol. 53, no. 2, pp. 197–218, Mar–Apr 2005.

[81] L. M. Wein, " Optimal Control of a Two-Station Brownian Network," *Mathematics of Operations Research,* vol. 15, no. 2, , pp. 215 - 242, May 1990.

[82] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research Society*, vol. 49, no. 3, pp. 237-252, 1998.

[83] S. H. Low and D. E. Lapsley, "Optimization Flow Control, I: Basic Algorithm and Convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861-874, Dec. 1999.

[84] J. Mo and J. Walrand, "Fair End-to-End Window-Based Congestion Control," *IEEE/ ACM Transactions on Networking*, vol. 8, no. 5, pp. 556-567, 2000.

[85] M. J. Neely, E. Modiano, and C-P. Li, "Fairness and Optimal Stochastic Control for Heterogeneous Networks," in *IEEE INFOCOM*, Mar. 2005.

[86] A. Eryilmaz and R. Srikant, "Fair Resource Allocation in Wireless Networks Using Queue-Length-Based Scheduling and Congestion Control," in *IEEE INFOCOM*, Mar. 2005.

[87] L. Jiang and J. Walrand, "Stable and Utility-Maximizing Scheduling for Stochastic Processing Networks," in *the 47th Annual Allerton Conference on Communication, Control, and Computing*, 2009.