THE COMPUTATION OF EQUILIBRIA IN CONGESTION NETWORKS

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy by

Pattarawit Polpinit

Supervisor: Paul W. Goldberg

January 2009

Contents

Illustrations

Figures

Tables

Notations

The following notations and abbreviations are found throughout this thesis:

$\mathbb{Z}$	The set of integers.
$\mathbb{R}^+$	The set of positive real numbers.
N	The set of nodes ($N = \{1, \ldots, n\}$).
E	The set of links ($E = \{1, \ldots, e\}$).
G	The directed network ($G = (N, E)$).
M	The set of players ($M = \{1, \ldots, m\}$).
f_j	A flow routed on link j.
$\ell_j(\cdot)$	A latency function of link j.
f^i	A flow quantity controlled by player i.
f^i_j	A flow quantity of player i on link j ($f^i = \sum_{j \in E} f^i_j$).
$\mathbf{f}^i$	A routing strategy of player i that represents a partition of f^i amongst all n links ($\mathbf{f}^i = (f^i_1, \ldots, f^i_n)$).
$\mathbf{f}$	A flow of which is a combination of player strategies, one for each player ($\mathbf{f} = (\mathbf{f}^1, \ldots, \mathbf{f}^m)$).
f	The total network flow quantity ($f = \sum_{i \in M} f^i$ and $f = \sum_{j \in E} f_j$).
$\mathbf{w}$	A weight vector ($\mathbf{w} = (f^1, \ldots, f^m)$).
$C^i(\mathbf{f})$	A cost of player i in $\mathbf{f}$ ($C^i(\mathbf{f}) = \sum_{j \in E} f^i_j \ell_j(f_j)$ where f^i_j is in $\mathbf{f}^i$ for every $j \in E$).
$C^i_j(\mathbf{f})$	A cost of player i from link j in $\mathbf{f}$ ($C^i_j(\mathbf{f}) = f^i_j \ell_j(f_j)$ where f^i_j is in $\mathbf{f}$).
$C(\mathbf{f})$	A social cost of flow $\mathbf{f}$ ($C(\mathbf{f}) = \sum_{i \in M} C^i(\mathbf{f})$ and $C(\mathbf{f}) = \sum_{j \in E} f_j \ell_j(f_j)$). Often we omit $\mathbf{f}$ if the reference to it is clearly understood from a context or unnecessary.
NE	Nash equilibrium.
SE	System equilibrium..
SSL	Selfish Stackelberg leadership.
UE	User equilibrium.

$\mathbf{f}^{\mathrm{SE}}$	Social optimal flow.
$\mathbf{f}^{\mathrm{NE}}$	Nash flow.
$\mathbf{f}^{\mathrm{SSL}}$	Selfish Stackelberg leadership flow.
$\mathbf{f}^{\mathrm{UE}}$	User equilibrium flow.
C^{SE}	Social cost of $\mathbf{f}^{\mathrm{SE}}$.
C^{NE}	Social cost of $\mathbf{f}^{\mathrm{NE}}$.
C^{SSL}	Social cost of $\mathbf{f}^{\mathrm{SSL}}$.
$\ell_{\min}$	A minimal latency of links used.
$\ell_{\max}$	A maximal latency of links used.
RLS	Randomised Local Search.
$\lambda_h(\mathbf{f})$	A flow after the h-th move from an initial flow $\mathbf{f}$ ($\lambda_0(\mathbf{f}) = \mathbf{f}$ and $\lambda_h(\mathbf{f})$ is obtained by a single move of one players flow from $\lambda_{h-1}(\mathbf{f})$).
c_j	A capacity of link j.
$c_{\max}$	The ratio of largest to smallest non-empty link capacities.
$f_{\max}$	The ratio of largest to smallest player flow.
$P(\lambda_h)$	A potential of flow λ_h ($P(\lambda_h) = \sum_{j\in E}(f_j(\lambda_h))^2/c_j + \sum_{i\in M}(f^i)^2/c_{\mathbf{f}^i(\lambda_h)}$).

Preface

This thesis is mainly my own work and the sources from which material is drawn are identified within. Chapter 1 and 2 contain introductory materials, literature review, notations, definitions, properties and examples of which are drawn from various authors from several related works, in particular from Roughgarden and Tardos [RT02] more than others. Chapter 4 is based on the paper [GP07] that has been co-authored by Paul Goldberg, and is intended to submit for publication. Chapter 5, is an extension of Goldberg's work [Gol04]—namely in [Gol04] identical links are considered while we consider variable-capacity links. Some definitions and notations in that chapter are borrowed from Even-dar et al.'s work [EDKM07], for instance the potential function. Finally Section 5.3 is drawn from [Gol04] to demonstrate the distributed version of the Randomised Local Search.

Abstract

We study a class of games in which a finite number of "selfish players" each controls a quantity of traffic to be routed through a congestion network in which n directed links are connected from a common source to a common destination. In particular, we investigate its equilibria in two settings.

Firstly, we consider a splittable flow model in which players are able to split their own flow between multiple paths through the network. Recent work on this model has contrasted the social cost of Nash equilibria with the best possible social cost. Here we show that additional costs are incurred in situations where a selfish "leader" player allocates its flow, and then commits to that choice so that other players are compelled to minimise their own cost based on the first player's choice. We find that even in simple networks, the leader can often improve its own cost at the expense of increased social cost. Focussing on a two-player case, we give upper and lower bounds on the worst-case additional cost incurred.

Secondly, we study the computation of pure Nash equilibrium for a load balancing game with variable-capacity links. In particular, we consider a simple generic algorithm for local optimisation; Randomised Local Search (RLS), which can simulate a network of selfish users that have no central control and only interact via the effect they have on the cost latency functions of links. It is known that an algorithm with series of self-improving moves will eventually reach the Nash equilibrium. Our contribution here is to show furthermore that Nash equilibria for this type of games are reached quickly by RLS.

Acknowledgements

The majority of thank has to go to my supervisor Paul Goldberg for introducing me to this research topic and continuous supports I have received from him throughout the course of my PhD.

I would like to thank the Algorithms and Computational Complexity Research Group at University of Warwick and the Complexity Theory and Algorithmics Group at University of Liverpool for all the help I have received during my time there.

I would like to thank some friends: Nick Palmer and Kasper Pedersen with whom I shared an office at Warwick, Kannika Thampanishvong and Ratchada Pattaranit with whom I shared a house at Warwick, Amornrat Aungwerojwit and Issavara Sirirungruang, Namthip Rujikaitkumjohn for all the encouraging phone calls, and especially Markus Jalsenius for being an excellent housemate at Liverpool and officemate both at Warwick and Liverpool.

I would also like to thank the Royal Thai government for their financial supports.

Finally, I would like to thank my parents for their support and encouragement throughout the entire time I have been in UK.

Chapter 1

Introduction

This thesis concerns the computation of *equilibria* in *congestion networks*. A congestion network is a directed graph in which a routing cost of each edge increases as a traffic on it increases. What happens when two or more players want to route their traffic through links in the network? Naturally, every player acts *selfishly*, choosing a routing strategy that will minimise its cost, without regard to other players' cost. Certainly, every player would expect the other players to behave in a similar egocentric fashion. Such a network is called *non-cooperative model* and a famous example of it is the Internet. A non-cooperative model has become increasing important in computer science and game theory community.

A non-cooperative model is extensively studied in game theory community. Game theory is the study of multi-person decision problems. A game is a mathematical representation of the phenomena when decision-makers interact. A game consists of a set of players, a set of strategies available to those players and a specification payoffs for each combination of strategies.

1.1 Background

To introduce and motivate the questions investigated in this thesis, let us informally describe two basic games. The first is "Cournot competition" (sometimes called "Cournot duopoly" in economics as the model represents a market with only two producers), which was introduced by Cournot in 1838. The second is "congestion game", which was introduced by Rosenthal in 1973 [Ros73a].

1.1.1 Cournot Competition

Imagine there are two firms in the market that produce the same good. Suppose that the cost experienced by each firm increases as the output it produces increases. All the

output is sold at a single price, determined by the demand for the good and the firms' total output. Suppose that the market price decreases as the total output increases (unless it is already zero). Each firm's revenue is its output times price. Thus each firm's profit is its revenue minus its cost.

Assuming both firms are selfish, namely each firm wants to achieve the highest possible profit, typical question to be asked would be what quantities q_1 and q_2, will firm 1 and firm 2 manufacture respectively? Assume that both firms make their decision simultaneously and independently; initially, each firm chooses to manufacture a quantity that will minimise its cost without information of the other firm's output. Both firms are allowed to change their output provided that, after the change, their profit will increase. After a long enough sequence of changes, we would expect the market to reach a stable state in a sense that no firm can increase its profit by changing its output unilaterally. This stable state is known in the game theory community as a *Nash equilibrium*.

Suppose that firm 1 has a *Stackelberg leadership*. That is firm 1 chooses what q_1 at the start of the game and has to fix to that q_1, then firm 2 chooses q_2, knowing the output chosen by firm 1. To find the Nash equilibrium in this case, firstly, for any output q_1, player 1 knows that player 2 will always manufacture the output that maximises its profit. Suppose that for each output q_1 of firm 1 there is one such output of firm 2; denote this with $b_2(q_1)$, thus the total output is $q_1 + b_2(q_1)$. Intuitively, firm 1 manufactures q_1 that together with its corresponding $b_2(q_1)$ maximises its profit.

It has been noticed that in the situation where firm 1 has the Stackelberg leadership, it produces more output and obtains more profit than the situation where both firms chooses a good to produce simultaneously. Subsequently, firm 2 produces less output and obtains less profit in the sequential-move scenario than it does in the simultaneous-choose scenario. More importantly, it turns out that firm 1's profit rises by a lesser amount than firm 2's profit falls, thus the aggregate profit is lower. (see e.g. 2.1.B in [Gib92] for more details)

1.1.2 Congestion Network Game

The Cournot competition suggests two interesting observations: the entire network can be worse off with a Stackelberg leader than the standard Nash setting, and a sequence of "self-improving" moves could lead to an equilibrium in the market. Next, we consider a congestion network game which has been studied recently by researchers in computer science as a basic foundation of routing information in a computer network. Our concentration will be on the loss of network efficiency in a model with a Stackelberg leader from the model where all players simultaneously compete for their minimal costs.

We begin by imagining there are two players that wish to route a flow from the same

source to the same destination through a shared network. Suppose that each player can split its flow arbitrarily across all links in which each player experiences a cost of using each link that is proportional to the sum of the total flow sharing that link. Thus, the total cost experienced by each player is the sum over all links of the cost of using each link. Again we assume that both players are selfish, want to minimise their own cost without regard to the cost that the other player might suffer from the decision it makes. If we let the game runs with both players making their decision simultaneously and independently, similar to the Cournot competition, we expect the network to eventually reach the Nash equilibrium after a long enough sequence of flow changes.

Suppose that player 1 has a Stackelberg leadership. Being selfish, player 1 may find that it pays to route more flow on "cheaper" links, forcing player 2 to route less flow on those links, but more flow on "more expensive" links than it does in the simultaneous setting.

For a better explanation, imagine an example of the network that has two parallel links. The cost of using link 1 equals the total flow on it, and the cost of using link 2 equals twice the total flow on it plus 1. Each player has one unit of flow to route. Assume that player 1 routes the fractions f_1^1 and f_2^1 of its flow on link 1 and link 2 respectively. Similarly, assume that player 2 routes the fractions f_1^2 and f_2^2 of its flow on link 1 and link 2 respectively. The total cost experienced by player 1 is f_1^1 times the total flow on link 1, plus f_2^1 times the total flow on link 2. The total cost experienced by player 2 is f_1^2 times the total flow on link 1, plus f_2^2 times the total flow on link 2. The social cost is the sum of player 1's cost and player 2's cost.

The Nash equilibrium in this setting is the point when both players route 7/9 of their flow on link 1 and the rest on link 2. When player 1 has the Stackelberg leadership, it could have achieved its cost at the Nash equilibrium, but chooses to play differently if that means its individual cost will be lower. It finds that overusing link 1 will force player 2 to move some of its flow from link 1 to link 2, which will result in a lower cost for player 1. Specifically, player 1 routes 5/6 on link 1 inducing player 2 to route 3/4 on link 1, i.e. 1/36 less than it does at the Nash equilibrium. Consequently, in the simultaneous-move setting, the cost of both player 1 and player 2 are 1.63, and the social cost is 3.26. In the sequential-move setting, the cost of player 1 is reduced to 1.62, but the cost of player 2 increases to 1.65, as a result, the social cost increases to 3.27. In summary, the cost of player 1 falls in the sequential-move setting by a lesser amount than the cost of player 2 rises in the sequential-move setting.

We have just reaffirmed that there may be an additional cost in a Nash equilibrium when one of the players is allowed to have a Stackelberg leadership, over and above the standard simultaneous-setting Nash equilibrium. In this thesis, we study this additional cost. Specifically, we consider a two-player game in a network with n parallel links where

each link has an non-decreasing convex latency function. Each player has a quantity of flow that can be split across available links. All players are assumed to be selfish and will route their flow to minimise their personal cost. In order to gain a better understanding of this principle, we study the computation of Nash equilibrium in these networks, thus developing techniques and mathematical tools for quantifying the worst additional social cost due to a selfish Stackelberg leader in this setting.

1.1.3 Nash Equilibrium

Nash equilibrium is arguably one of the most fundamental concepts in game theory [Osb94]. It is well known to be named after John Nash for his contribution in the 1951 paper [Nas51] that proved that, in any game with a finite number of actions, at least one equilibrium exists. Thereafter, Nash equilibrium has been studied extensively in many contexts by many researchers of many different fields. For the past few decades, researchers in theoretical computer science have been trying to create a foundational understanding of the selfish behaviour of users in the Internet. It seems that a natural tool to analyse this class of problems may come from that of the noncooperative game theory, and an appropriate solution concept is that of Nash equilibrium.

However, the original notion of Nash equilibrium studied by Nash has limitations in representing the present-day Internet. First, the classical Nash equilibrium that arises in a game has a finite set of players, while in the today Internet, there are so many users that it could be deemed as infinitely many user network. Second, in the classical Nash equilibrium, all players are assumed to have "perfect-information", that is every player knows what every other players do. This is impossible with the astronomical scale of the Internet and the limited amount of time for each player to choose a path for transferring data.

There are a few alternative approaches that we can see being a mathematical representation of the Internet. For example "Selfish routing" initialised by Roughgarden and Tardos [RT02] considers a simpler network in which there are an infinite number of infinitesimally small players. A Nash equilibrium in this setting is often called "Wardrop's first principle" which states that all users choose a minimal-cost path to route their traffic (we briefly discuss the computation of Nash equilibrium in this setting in Section 2.3). Another possible alternative is an approximate Nash equilibrium of a model where each user is restricted to some constraints, e.g. a user only aware of other "nearby" users' traffic activities.

1.1.4 The Problem Statements

Interesting classes of noncooperative games are congestion games [Ros73a] and its equivalent model of *potential games* [MS96]. Recent works on Nash equilibrium in congestion games can be briefly summarised as follows:

1. Quality of Nash equilibrium:
   (a) the existence and uniqueness of Nash equilibria with distinct cost functions in a system of selfish players.
   (b) the loss of social welfare due to selfish, uncoordinated behaviour by comparing the social cost of Nash equilibrium with the socially optimal cost.
2. Computation of a Nash equilibrium:
   (a) via "unrestricted algorithms".
   (b) via algorithms that simulate selfish behaviour.

In this thesis, we study two of those topics: 1b and 2b. Regarding the problem of the quality of Nash equilibrium, it is known that a Nash equilibrium is suboptimal [RT02]. Furthermore, as we have shown in examples of Cournot competition and congestion network game, there can also be an additional cost to the Nash equilibrium of a network that has a Stackelberg leader. We are interested to know how much is the additional social cost because of a selfish leader, specifically in two-player games?

Regarding the problem of the computation of Nash equilibrium, we study a simple generic algorithm for local optimisation called *Randomised Local Search* that can construct a Nash equilibrium. It is known that there are efficient algorithms for finding a Nash equilibrium (e.g. [FKK$^+$02, EDKM07]). The main reason we are interested in studying Randomised Local Search is because it can be realised by a simple distributed network of selfish users that have no central control and only interact via the effect they have on the latency functions of links, hence simulates a real network of internet users.

For this problem, the model will be slightly different from that of the problem of the quality of Nash equilibrium. We concentrate on a finite number of players, rather than just two players usually assumed in the previous problem. Moreover, each player chooses a single link to route its flow, rather than splitting its flow across available links. This is called *pure strategy*. An opposite strategy to pure strategy is *mixed strategy* in which each player decides on a probability for each pure strategy. By restricting to pure strategy, we always have, at least, one pure Nash equilibrium in our setting [FKK$^+$02, EDKM07].

Consider the congestion game shown in Subsection 1.1.2, Randomised Local Search (RLS) could construct a Nash equilibrium as follows. At each step, RLS selects a

player and a link uniformly at random. The player moves its flow to that link if its resulting cost is lower. It is known (e.g. [EDKM07, FKK+02]) that an algorithm with the sequence of such "self-improving" moves in this particular setting must eventually reach a Nash equilibrium. Clearly, RLS will not make a self-improving move on every attempt, however, we show that RLS has a high enough probability of selecting a successful attempt to construct a Nash equilibrium quickly.

For this topic, we will be concentrating on a potential game framework. In a potential game, there is a potential function which maps a current state of the game to a real number (in our situation, a state would mean the assignment of flows to links). Thus, in our RLS, the move happens when the resulting potential is lower. If the change to player costs and that to the potential function caused by a move of a single player flow is related by a factor that only depends on that player, the game is called "weighted potential game" which is what we will be focussing on. (For other types of potential games see previous works later in this chapter or [MS96].)

Our focus will be on the load-balancing model introduced by Koutsoupias and Papadimitriou [KP99], that is intended to model a simple version of the Internet of users and service providers. A load balancing game is essentially equivalent to a two nodes connected by parallel links. We consider a scenario where each player controls a non-negligible flow.

The problems being studied in this thesis can be summarised as follows:

- How much is the loss of social welfare due to a selfish leader, especially in the two-player scenario? The problem is extensively studied in Chapters 2–4.
- Does RLS converge quickly to a Nash equilibrium in a load balancing game? In Chapter 5, we present convergence rate bounds of RLS to address this problem.

1.2 Our Contributions

Our contributions can be categorised into two main topics. In Subsection 1.2.1 we describe the results in Chapter 3 and 4, and the results, which are described in Subsection 1.2.2, are based on the study in Chapter 5.

To describe our results more precisely, we must be more formal about our model of selfish routing in a congestion network. We consider a parallel-link network in which there are n links, all of which directing from a common source node to a common sink node. Each link has a latency function as a function of the link congestion on it. We assume that all latency functions are non-negative, non-decreasing and convex. There are m players, each of which controls a non-negligible flow to be routed from the source to the sink. In Chapter 3 and 4, we assume player flows are splittable, while unsplittable

flow is studied in Chapter 5. For the splittable flow scenario, each player experiences the cost that is the sum, over all links used by that player, of the amount of that player's flow on that link multiplied by the latency of using that link, while, for the unsplittable flow case, each player experiences the total cost of the link which that player uses.

We assume that every player will always choose a routing strategy, namely an assignment of flow, that will minimise its cost. As the route which is chosen by one player affects the congestion (and hence the latency) experienced by others, it is natural to apply the theory of a non-cooperative game. Following the conventions of the non-cooperative game, the network is at Nash equilibrium if no player can reduce its cost by unilaterally changing its routing strategy. The network is called *system equilibrium* (SE) [CP91] when the network has a minimal-possible total cost.

1.2.1 Bounding the Price of Selfish Stackelberg Leadership

In Chapters 3 and 4, our focus is on a network of two players, each with a non-negligible splittable flow. We call a player a *leader* if it has a Stackelberg leadership, that is, a leader chooses its routing strategy before others. The leader is not allowed to change its routing strategy afterwards. The other player(s) is called *follower(s)*. The followers react to the leader's strategy, reaching the Nash equilibrium relative to it. In the case of two-player games in which we will be study mainly in Chapter 4, the follower will treat the leader's flow as constant, and will try to achieve a system equilibrium relative to the modified network. In other words, the follower will behave like an optimal flow relative to the modified network after the leader has played. The network is at *Stackelberg equilibrium* when the flow of the followers reaches a Nash equilibrium, or, in the case of two-player games, the follower reach a system equilibrium. We call the Stackelberg equilibrium where the leader's cost is minimal the *selfish Stackelberg leadership equilibrium* (which will often be shortened to *SSL equilibrium*).

The main objective is to obtain bounds on the additional social cost caused when one of the players becomes a leader. We define the worst ratio between the social cost at Nash equilibria and that at SSL equilibria, the *price of selfish Stackelberg leadership* (which will often be shortened to the *price of SSL*).

Even though, a game of two players is one of the most basic models studied in non-cooperative game theory, it is arguably one of the most studied games. Many classical examples are two-player games, for instance Cournot competition which we have shown earlier, the prisoner's dilemma which famously illustrates that Nash equilibrium is sub-optimal, and the battle of sexes which shows that there could be more than one pure Nash equilibria.

In Chapter 2, we give the basic definitions and preliminary technical results needed in

the rest of this thesis. The technical results, which are shown in Section 2.2–2.4, include techniques for finding a system equilibrium, a Nash equilibrium and a SSL equilibrium. By using the techniques, we demonstrate the social costs of several alternative network set-ups to not only show the positive price of SSL, but also compare and contrast to other problems studied in related works.

In Chapter 3, we study lower bounds on the price of SSL for various latency function forms. From examples shown in Chapter 2, we notice that, different sets of coefficients of latency functions and a player fraction can generate different prices of SSL. This has raised a few questions which lead us to the experiments in Chapter 3. In Chapter 3, we optimise the price of SSL with respect to coefficients of latency functions and a fraction of player flows in simple restricted networks. The latency functions considered are affine linear, quadratic, cubic and quartic. We solve the optimisation problem by using a local search algorithm that is based on Hill climbing (see e.g. [RN03] for more detail on Hill climbing). We found that, for the network of two players, both with access to two parallel links, each of which has affine linear latency functions, the price of SSL can be as high as 1.075. For polynomial latency functions, we found the price of SSL of 1.091, 1.135 and 1.161 for quadratic, cubic and quartic latency functions respectively. For an asymmetric network that has two private links which only the owner of that link can use, and one shared link; every link has a linear latency function, we show the price of SSL of 1.074. These prices of SSL establish lower bounds.

In Chapter 4, we study upper bounds on the price of SSL. Our focus will be on linear latency functions. If the latency functions are homogenous—homogenous linear functions are in the form of $f(x) = ax$ where $a > 0$, not only is there no price of SSL, but also all players split their flows in a uniform way. For the case of affine linear latency functions, we show that the price of SSL is at most multiplicative constant (thus, independent of the number of links) at most 1.322. This upper bound corresponds to the lower bound of a multiplicative factor of 1.075.

1.2.2 Bounding the Convergence Rate of Randomised Local Search in a Load-balancing Game with Variable-capacity Resources

In contrast, in Chapter 5, we are concerned with an algorithm that can construct a Nash equilibrium and the time it takes to reach a Nash equilibrium, rather than the quality of Nash equilibrium. Our focus is still on the parallel-link network. However there are some crucial differences between this chapter and the previous chapters. First, we consider m players in this chapter while two-player games are considered in the previous chapter. Moreover player flows are restricted to be integer, and the total flow is not necessarily one as previously assumed in the previous chapters. And finally player flows

are unsplittable, that is each player chooses a single link to route its flow.

We assume that all player flows are positive integer in the range $\{1, \ldots, f_{\max}\}$ where $f_{\max}$ denotes the ratio of the maximal to the minimal player flow. Every link in the network has a capacity which is assumed to be a positive integer in the range $\{1, \ldots, c_{\max}\}$ where $c_{\max}$ denotes the ratio of the maximal to the minimal capacity. This model is equivalent to the parallel-link network in which each link has a homogeneous linear latency function of the form $f(x) = ax$, where the coefficient a is a fraction of one over the capacity of that link.

We study a simple generic local search called Randomised Local Search (RLS). An *attempt* is the process via RLS of selecting a single player and a single link uniformly at random. If the selected player can reduce its cost by transferring its flow to the selected link, then that attempt is *successful*. We call a successful attempt a *move*. We study the number of attempts RLS takes to reach a Nash equilibrium.

We show that RLS can be realised in a distributed setting and present a distributed version of RLS that can be executed by all players simultaneously in non-cooperative games.

Two upper bounds are proved for the expected number of attempts for Nash convergence via RLS. Firstly, we quickly obtained an upper bound of $O(m^3 n (f_{\max})^2 (c_{\max})^2)$. Secondly, with a more detailed analysis, we prove an alternative upper bound of $O\left(n(mc_{\max}f_{\max})^2(n + c_{\max}f_{\max})\right)$. We consider the later bound to be better in a sense that the number if players m decreases from cubic to quadratic, and could be linear in the distributed setting.

1.3 Previous Work

In this section, we describe and compare some previous works related to the problems studied in this thesis.

Selfish Routing

The term "selfish routing" was initialised in computer science mainly by Roughgarden and Tardos in [Rou05a, RT02] to represent a model of non-cooperative, selfish users simultaneously active in transportation networks. However, the original concept was formally formulated as far as the work of Beckmann et al. in 1955 [BMW55]. Most of the early works considered the selfish routing in transportation models, until Rosenthal [Ros73a, Ros73b] described how the transportation model can be naturally generalised to a more abstract setting, and introduced the name "congestion game" for the model in 1973.

Even though, a selfish routing in congestion games has been studied extensively thereafter, it was not until 1999 that the idea of quantifying the cost of selfish routing was proposed by Koutsoupias and Papadimitriou [KP99]. The main idea is to measure how much selfishness can degrade the overall performance of a system. The term *coordination ratio* was used as a tool to measure the degradation in that paper, but later Papadimitriou replaced it with the new term, the "price of anarchy" in [Pap01]. The term mathematically represents the ratio of the cost in the worst possible Nash equilibrium to the best possible social cost.

The concept of the price of anarchy of selfish routing has been studied in numerous variants and generalisations. Two of the earliest works that studied the concept are [KP99, RT02]. Koutsoupias and Papadimitriou [KP99] studied a simple parallel-link network (though it was called a load-balancing network in that paper) which is one of the most commonly studied network topologies in computer science (e.g. [CV02, KLO97b, KP99, MS01, ORS93, Rou04]). In that work, each user chooses a probability based on the set of links (specifying the probability that the user will route all of its flow on a given link). Each user wishes to minimise the expected cost it will experience, while the global objective is to minimise the maximum expected latency. They obtained a tight analysis of the price of anarchy in two-node, two-link networks and obtained partial results for two-node networks with three or more parallel links.

Roughgarden and Tardos [RT02] studied the price of anarchy in a *nonatomic* model, in which there is an infinite number of players, each of whom controls a negligible amount of traffic. For a general network, they proved that, if the latency of each link is a linear function of its congestion, then the price of anarchy is at most 4/3. For the more general setting in which the edge latency functions are assumed only to be continuous and nondecreasing in the edge congestion, they showed that the price of anarchy is unbounded; however, the social cost of selfish users is no more than the total cost incurred by optimally routing twice as much traffic (see the book [Rou05a] by Roughgarden for a good overview of the price of anarchy).

The Price of Selfish Stackelberg Leadership

In contrast with the previous works on selfish routing that study games in the simultaneous setting, namely every player play a game simultaneously, we are interested in the model of having one of the players acts as a leader who will select its strategy before others. If the leader hold its strategy fixed, the other players react to the modified network, reaching the Nash equilibrium relative to it. This model is described in the game theory community as a Stackelberg game, and its corresponding equilibrium is

known as a Stackelberg equilibrium. The Stackelberg game is named after von Stackelberg [von34] who originally considered the model of two competing companies. He introduced the idea of two-stage game; the leader choosing a strategy in the first stage, and the follower choosing a strategy in the second stage, with full knowledge of the leader's strategy.

Many works have studied the Stackelberg games and the corresponding Stackelberg equilibria in congestion games, for example [CS07, ES90, KS06, KPS05, KK07, KLO97a, KM02, Rou04, Swa07, YZM07]. Problems that recent works have considered in the Stackelberg game include

- What is the loss (or gain) of efficiency of the system with a Stackelberg leader?
- What is the impact of the following leader behaviours:
    1. A selfish leader—a leader whose objective is to minimise its own cost.
    2. A benign leader—a leader whose objective is to achieve the lowest possible social cost of the system.
    3. A malicious leader—a leader whose objective is to maximise the social cost of the system.

Most of the recent works on the Stackelberg games considered the benign leader in the context of network flow as a tool for mitigating the degradation due to selfish users, for instance [CS07, KS06, KPS05, KLO97a, Rou04, Swa07]. The fraction of the flow that is controlled by the leader is routed so as to minimise the social cost in the presence of followers who minimise their own cost. One of the earlier works that studied this problem was by Korilis et al. [KLO97a]. They considered a model of a finite number of players, each with a splittable flow in a parallel-link network with linear latency functions. They proposed the leader's guaranteed-optimal strategy and methodology to derive it. Roughgarden [Rou04] investigated the same parallel-link model, but with infinitely many infinitesimal followers. He showed that it is $\mathcal{NP}$-hard to find an optimal strategy for the benign leader in this setting. Furthermore, he proved that if an α fraction of the overall traffic can be globally regulated, then for an arbitrary class of latency functions, there is a strategy that ensures that the price of anarchy is at most $1/\alpha$. For the same model, Kaporis and Spirakis [KS06] analysed the least portion of a total flow that the leader must have in order to enforce the optimal system cost.

In contrast to other recent works on the Stackelberg games in congestion networks, we are interested in the impact of a selfish leader on the networks. We consider a basic model in which there are one leader and one follower, each with a non-negligible

splittable flow[1]. We study an additional social cost occurred when the leader chooses the strategy that will minimise the leader's cost, by predicting what the followers will do afterward to maximise their own costs.

We denote the additional social cost induced by a leader with the price of selfish Stackelberg leadership. The concept of the price of SSL, as far as we know, has never been formally studied even for a simple two-player game studied in this thesis. However, there have been observations of this additional cost, for example, in Cournot competition, Gibbon [Gib92] pointed out that a follower profit falls more than a leader profit rises when compare to the standard Nash equilibrium.

Nash equilibrium

In this thesis, we study Nash equilibrium of the atomic splittable model. Nash equilibrium is a solution concept of a game involving two or more players. The original concept was first developed by Cournot in 1838 for pure strategies in a two-player game. Back then Nash equilibrium was called Cournot equilibrium. It was a long time after then before von Neumann and Morgenstern [vM44] introduced Nash equilibrium for mixed strategies and showed that a mixed-strategy Nash equilibrium will exist for any zero-sum game[2] with a finite set of actions in 1944. In 1951, Nash [Nas51] had a breakthrough result by showing that a mixed strategy Nash equilibrium for any game with a finite set of actions must exist.

Many researchers have pursued existence and uniqueness results of Nash equilibrium for the atomic model and several extensions. Orda et al. [ORS93] showed that the Nash equilibrium of a network of two nodes connected by parallel links each of which has a convex latency function always exists. This fact is used in out thesis to guarantee the existence of Nash equilibrium. They further proved that if latency functions are increasing functions, the Nash equilibrium is unique. They proved their theorems using the classical existence theorem of [Ros65]. Independently, Harker [Har85] proved the same facts using the theory of variational inequalities. Altman et al. [ABJS00] extended the uniqueness results to general networks for a restricted class of latency functions. Haurie and Marcott [HM85] showed that the Nash equilibrium for the nonatomic model is the unique limit of any sequence of Nash equilibria obtained for a sequence of the atomic splittable games in which the number of players is finite and tends to infinity.

[1] A model where there are finitely many players each with non-negligible spittable flow is sometimes called an *atomic* model, and a player in the model is called atomic player.

[2] A game in which a player's gain or loss is exactly balanced by the losses or gains of the other player(s). If the total gains of the players are added up, and the total losses are subtracted, they will sum to zero.

The Convergence Rate of RLS

In Chapter 5, We consider a simple generic algorithm called Randomised Local Search (RLS) since it is similar to "randomised local search" that is studied in other contexts. Because of its generic and simple to implement, RLS is used in many problems, especially as a generic optimisation method ([GW03, NW07, Weg03, WW05]). For example, given a binary string of length m with a function $f : \{0,1\}^m \to \mathbb{R}$ to be optimised, RLS can be used to flip repeatedly a random bit. The change is accepted whenever f is higher for the modified string. In the context of minimum spanning tree [NW07] and maximum matching [GW03], a variant of RLS is used where two bits may be flipped simultaneously. In our setting, an assignment is an element of $\{1,\ldots,n\}^m$ where m is the number of players and n is the number of links. RLS repeatedly replaces a random entry of this vector by a random element of $\{1,\ldots,n\}$, and accepts the new assignment if its potential value of f is lower. RLS is a variant of the more extensively studied (1+1) Evolutionary Algorithm of [DJW02].

We study a congestion game in a framework of a potential game which was introduced by Monderer and Shapley [MS96]. In a potential game, there is a potential function which maps a flow configuration to a real number. Comparing the change in player's cost with that in the potential function as a result of that player deviates its route, Monderer and Shapley categorised a class of potential games as follows. In an exact potential game, the change are identical. In a weighted potential game, the changes are related by a factor that depends only on the player. In an ordinal potential game, the changes are in the same direction, while, in a generalised potential game, a decrease in a players cost implies a decrease in the potential function (but not vice versa). In this thesis, we borrow the potential function from [EDKM07] in which the game is proved to be a weighted potential game.

Our focus in this topic is on pure strategies in a parallel-link network network with arbitrary capacities. It was shown by Fotakis et al. [FKK$^+$02] how to find a pure-strategy Nash equilibrium in polynomial time with respect to the number of players in the model. In the same setting, Feldmann et al. [FGL$^+$03] termed an algorithm that constructs a Nash equilibrium from an arbitrary state without increasing the social cost, a Nashification algorithm. They showed that an algorithm that allows an unsatisfied player to move using its best strategy may take time exponential in the number of players, for both identical link and links with arbitrary capacities.

We consider an integer flow setting, i.e. all players flow are positive integer. Evendar at el. [EDKM07] proved an upper bound for the time an self-improving algorithm takes to reach a Nash equilibrium. The bound is of $O(W^2 c_{\max}^2)$ where W denotes the maximal total weight of players. Note that W would be equivalent to $mf_{\max}$ in our

notation system.

With regard to random algorithms constructing Nash equilibria, Even-dar at el. [EDKM07] studies "Random" algorithm, in which at each step a player is randomly selected; then its flow is moved to the lowest-cost link. They obtained an upper bound of $O(W+m)$ for identical links. In the same model, Goldberg [Gol04] studied RLS and proved a bound of $O(n^2 m f_{\max})$ for the integer flow setting. The work in this topic in this thesis is motivated by [Gol04] and somewhat an attempt to extend [Gol04]. For unrestricted-flow, Goldberg also proved an upper bound of $O(f_{\max}^2 n^4 m^5 \log(mn))$.

1.4 Thesis Outline

The rest of the thesis is organised as follows. The notations of the model and the basic definitions are formally given in Chapter 2. This is followed by the definitions and computational methods of the three equilibrium concepts; System equilibrium, Nash equilibrium and SSL equilibrium. We end Chapter 2 with Pigou's example, in which we study a simple two-node, two-parallel-link network in various set-ups.

The methods used to compute the equilibria described earlier are used in Chapter 3 in our optimisation program to find the lower bound of the price of SSL. We give several examples which shows that the price of SSL in a simple two-parallel-link networks with latency functions that are linear, quadratic, cubic and quartic, and a simple asymmetric network where two links are private links and one link is a shared link: every links have linear latency functions. The upper bounds on the price of SSL for the two players in a parallel-link network with linear latency functions are proved in Chapter 4. We end the chapter with the results showing that the price of SSL for homogeneous linear latency functions is one and Nash flow and SSL flow are identical and unique.

We study RLS in load balancing games in Chapter 5. We show a distributed version of RLS before proving the upper bounds on the number of attempts RLS take to reach a Nash equilibrium for players with integer flow.

Finally, the conclusions and possibilities for future works are discussed in Chapter 6.

Chapter 2

Preliminaries

In this chapter, we present the notations, basic definitions and properties needed for the rest of the thesis. In Section 2.1, we give the formal definitions and notations of the models. We discuss the characteristics and techniques for finding a system equilibrium, a Nash equilibrium and a SSL equilibrium in Section 2.2, 2.3 and 2.4 respectively. Finally, in Section 2.5, we present a simple example that whereby using techniques for finding the equilibria shown in the earlier sections, demonstrates the social cost of several alternative network set-ups. The social costs will not only show the additional social cost that could arise in a SSL equilibrium over that in a Nash equilibrium, but are also used to compare and contrast the problem studied in this thesis with the previous works discussed in Section 1.3.

2.1 The Model

We consider a directed network $G = (N, E)$, where N is a set of *nodes* (or *vertices*) and E is a set of *links* (or *edges*). We consider a special case where there are two nodes in N, where we call one of the nodes a *source* and the other node a *sink*. A link is a directed link that connects the source node to the sink node. We denote n the number of links in E, i.e. $|E| = n$. For each link $j \in E$, we define a *latency* function $\ell_j : \mathbb{R}^+ \longrightarrow \mathbb{R}^+$ as the cost of using link j which is a function of the load on j. We will generally assume that latency functions are non-negative, continuous and non-decreasing.

We denote a set of *noncooperative players* by M and the number of players by m. Each player routes its flow from the source to the sink by splitting its flow over the set of available links. For each player $i \in M$, $f^i \in \mathbb{R}^+$ denotes an amount of flow belonging to player i, (in the related literature, flow is sometimes called "weight" or "demand"), and f^i_j denotes a non-negative flow of player i assigned to link j. Let $\mathbf{f}^i = (f^i_1, \ldots, f^i_n)$ denote a *strategy* (or a *player flow*) of player i that represents a partition of f^i amongst all n

links in E. Let a *flow* $\mathbf{f} = (\mathbf{f}^1, \ldots, \mathbf{f}^m)$ denote a combination of player strategies, one for each player (A flow is sometimes called a *player profile*, or a *solution*). Let $f \in \mathbb{R}^+$ denote an amount of the total flow in the system. A flow $\mathbf{f}$ is *feasible* if $f = \sum_{i \in M} f^i$. Furthermore, let $\mathbf{w}$ denote a weight vector $(f^1, \ldots, f^m)$ that specifies how much flow each player has.

Given a total flow f, $f_j \in \mathbb{R}^+$ denotes the portion of f on link $j \in E$, i.e. $f = \sum_j f_j$. We usually assume (by re-scaling as necessary) that the total amount of flow is one (unless we explicitly state another quantity). Regarding the latency functions ℓ_j, if f_j is the flow routed on j then the latency of link j is $\ell_j(f_j)$. It is sometimes useful to let ℓ_j denote $\ell_j(f_j)$.

We often denote one of the players as player 1 and the other as player 2. Also, we call the leader player 1, and the follower player 2 in the case where one of them is a leader.

Definition 2.1.1. *The* cost experienced by a player *with respect to a flow* $\mathbf{f} = (\mathbf{f}^1, \ldots, \mathbf{f}^m)$ *is the sum, over all links used by that player, of the amount of that player's flow on that link multiplied by the cost of using that link. (The cost of the link is of course affected by the other players' strategies.) The cost of player i under the flow* $\mathbf{f}$ *is denoted by*

$$C^i(\mathbf{f}) = \sum_{j \in E} f_j^i \ell_j(f_j). \tag{2.1}$$

We often use C^i for the cost of player i, dropping $\mathbf{f}$, if the reference to the flow is clear from the context or unnecessary. It is useful sometimes to denote $C_j^i = f_j^i \ell_j(f_j)$ the contribution to player i's cost from link j, thus

$$C^i = \sum_{j \in E} C_j^i.$$

We define a *social cost* $C(\mathbf{f})$ with respect to a flow $\mathbf{f}$ as the sum of the individual players' costs:

$$C(\mathbf{f}) = \sum_{i \in M} C^i(\mathbf{f}). \tag{2.2}$$

Again, we often use C instead of $C(\mathbf{f})$. A social cost can also be expressed as the sum over all links, of the flow on that link multiplied by the cost of using that link:

$$C = \sum_{j \in E} f_j \ell_j(f_j). \tag{2.3}$$

With respect to latency functions, our results usually apply to certain subclasses of latency functions, that have been considered in the literature and can be listed with a

formal definition in an incremental order of general form as follows.

- Homogeneous linear function is a linear function of the form $\ell_j(f_j) = a_j(f_j)$ where $a_j > 0$.
- Affine linear latency function is a linear function of the form $\ell_j(f_j) = a_j(f_j) + b_j$ where a_j,$b_j \geq 0$.
- Polynomial function has a general form of $\ell_j(f_j) = a_{j,d}(f_j)^d + a_{j,d-1}(f_j)^{d-1} + \ldots + a_{j,1}(f_j) + a_{j,0}$ where d is nonnegative integer and $a_{j,0}, \ldots, a_{j,d}$ are constant coefficients where $a_{j,d} \neq 0$. In particular we consider
  - Quadratic function: $d = 2$.
  - Cubic function: $d = 3$.
  - Quartic function: $d = 4$.
- Convex function.

Since all the above functions are convex, we give a formal definition of convexity as follows.

Definition 2.1.2. *[NW99]* ***Convexity***

1. *$S \in \mathbb{R}^n$ is a convex set if the straight line segment connecting any two points in S lies entirely inside S. Formally, for any two points $x \in S$ and $y \in S$, we have $\alpha x + (1-\alpha)y \in S$ for all $\alpha \in [0,1]$.*
2. *f is a convex function if its domain is a convex set and if for any two points x and y in this domain, the graph of f lies below the straight line connecting $(x, f(x))$ to $(y, f(y))$ in the space $\mathbb{R}^{n+1}$. That is, we have*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y),$$

for all $\alpha \in [0,1]$.

For the rest of this thesis, we will assume that for every link $j \in E$, $f_j \cdot \ell_j(f_j)$ is convex. Under this assumption, every cost function $C(\cdot)$ is convex. This gives us a useful property which states that any local optimal solution is a global solution in an optimisation problem where the objective function is convex (see e.g. Nocedal [NW99]). This becomes very useful for the problem of optimising $C(\cdot)$.

We usually assume that links are indexed in increasing order of marginal costs at zero flow, i.e. $\ell_j(0)$. For example, links are indexed in increasing order of b_j for networks with linear latency functions.

2.2 Optimal Flows

In this subsection, we note some basic facts and behaviours of optimal flows—flows with minimal-possible social costs. In [CP91], an optimal flow is called "System Equilibrium" (SE). We let $\mathbf{f}^{\mathrm{SE}}$ denote an optimal flow.

The objective of $\mathbf{f}^{\mathrm{SE}}$ is to optimise the social cost $C(\mathbf{f}) = \sum_{j \in E} f_j \ell_j(f_j)$ of a network. Intuitively, when $\mathbf{f}^{\mathrm{SE}}$ has reached any locally optimal outcome, moving flow from one link to another can only increase the cost of $\mathbf{f}^{\mathrm{SE}}$. That means that the marginal benefit of removing flow from any link is at most the marginal cost of adding flow to any other link. Since we assume all cost functions are convex, any local minimum is a global minimum. This condition should be necessary and sufficient for $\mathbf{f}^{\mathrm{SE}}$ to be globally optimal. Before we formally formulate this idea, let us define a *marginal cost* of a given latency function $\ell(\cdot)$.

Definition 2.2.1. *If $\ell(\cdot)$ is a differentiable latency function, then the corresponding* marginal cost function *is $\ell'(\cdot)$ defined by*

$$\ell'(x) = \frac{\partial}{\partial x}(x \cdot \ell(x)). \tag{2.4}$$

Given the definition of marginal cost function, we are ready to formalise an important characteristic of the optimal flow in the following proposition. (The proof will be omitted. See [BMW55] for a complete proof).

Proposition 2.2.2. *[BMW55] Suppose we are given a network G. A flow $\mathbf{f}$ is an optimal flow of G if and only if for every j and $k \in E$ with $f_j > 0$, $\ell'_j(f_j) \leq \ell'_k(f_k)$.*

Briefly, Proposition 2.2.2 implies that, for a parallel-link network, the marginal cost of using link j is equal to the marginal cost of using link k if both j and k have non-zero flow in the optimal flow in the network.

Next, it will be useful, especially for Chapter 4, to consider a special case of affine linear latency networks. Recall that a linear function is of the form $a_j f_j + b_j$, for a_j and $b_j \geq 0$. The following lemma shows a characteristic of $\mathbf{f}^{\mathrm{SE}}$ when the latency functions of a network are linear. Recall that $\mathbf{w}$ denotes a weight vector $(f^1, \ldots, f^m)$.

Lemma 2.2.3. *[RT02] Suppose we have a network G with affine latency functions $\ell_j(f_j) = a_j f_j + b_j$, and a weight vector $\mathbf{w}$ to be routed on G. An optimal flow of G has the property that $2a_j f_j + b_j$ is the same for all links j on which $f_j > 0$.*

Proof. From Proposition 2.2.2, $\mathbf{f}^{\mathrm{SE}}$ is globally optimal if and only if for all j and $k \in E$ that are used, $\ell'_j(f_j) = \ell'_k(f_k)$. By Definition 2.2.1, for every link j and k, we have

$$f_j \frac{\partial \ell_j(f_j)}{\partial f_j} + \ell_j(f_j) = f_k \frac{\partial \ell_k(f_k)}{\partial f_k} + \ell_k(f_k).$$

In the linear context where $\ell_j(f_j) = a_j f_j + b_j$, we are saying that for all j, $f_j a_j + a_j f_j + b_j$ is the same, i.e. the result follows. □

One consequence of the above observation is that all latencies end up within a factor of 2 of each other (for links that carry non-zero flow).

Corollary 2.2.4. *Let G be a network with affine linear latency functions. If $\mathbf{f}$ is an optimal flow of G then the maximum latency of links used is at most twice the minimal latency of links used, in $\mathbf{f}$.*

Proof. Suppose j is a maximal latency link and k is a minimal latency link in an optimal flow of a given network G. If both j and k carry flow, then, from Lemma 2.2.3, $2a_j f_j + b_j = 2a_k f_k + b_k$. The ratio of the latency of j to that of k is

$$\frac{a_j f_j + b_j}{a_k f_k + b_k} = \frac{2a_j f_j + b_j - a_j f_j}{a_k f_k + b_k} = \frac{a_k f_k + b_k + (a_k f_k - a_j f_j)}{a_k f_k + b_k} \leq 2.$$

□

Another useful consequence from Lemma 2.2.3 is that the latency difference between the two links that are used in $\mathbf{f}^{\mathrm{SE}}$ will end up being half of the difference of their marginal costs at zero flow b.

Lemma 2.2.5. *Let G be a network with affine linear latencies in the form $\ell_j(f_j) = a_j f_j + b_j$, and suppose links j and k both carry flow in an optimal flow $\mathbf{f}^{SE}$ of G. Assume $j < k$ and $b_j < b_k$, and f_j^{SE} denotes a flow of $\mathbf{f}^{SE}$ on link $j \in E$. Then $\ell_k(f_k^{SE}) - \ell_j(f_j^{SE}) = \frac{1}{2}(b_k - b_j)$.*

Proof. If j and k both carry flow, then we noted from Lemma 2.2.3 that $2a_j f_j + b_j$ is the same for all j. Hence we have,

$$\begin{aligned} 2\left(\ell_k(f_k^{\mathrm{SE}}) - \ell_j(f_j^{\mathrm{SE}})\right) &= (2a_k f_k^{\mathrm{SE}} + 2b_k) - (2a_j f_j^{\mathrm{SE}} + 2b_j) \\ &= (2a_k f_k^{\mathrm{SE}} + b_k) - (2a_j f_j^{\mathrm{SE}} + b_j) + b_k - b_j \\ &= b_k - b_j. \end{aligned}$$

□

Note that Lemma 2.2.5 implies the fact (shown in [HTW06]) that in the homogeneous linear case (where all b_j's are zero) the latencies are equal in an optimal flow. They also showed that flows at Nash equilibrium will behave the same. We study the homogeneous

linear case in details later in Section 4.5 and show that flows at SSL equilibrium have the same property.

One consequence from Lemma 2.2.5 is that, if an optimal flow is routed through a set of links, then, given our assumption that links are indexed in increasing order of b_j, their latencies $\ell_j(f_j^{\mathrm{SE}})$ are sorted in increasing order of j.

Corollary 2.2.6. *Let G be a parallel-link network with linear latency functions, and suppose that links are labelled in increasing order of b_j, i.e. $j < k$ if $b_j \leq b_k$. If $\mathbf{f}$ is an optimal flow of G then the latencies $\ell_j(f_j)$ are sorted in increasing order of j, i.e. $\ell_j(f_j) \leq \ell_k(f_k)$ if $b_j \leq b_k$.*

Proof. Suppose, for a contradiction proof, that there are two links j and k such that if $\mathbf{f}$ is an optimal flow of a given network G and $b_j \leq b_k$ then $\ell_j(f_j) > \ell_k(f_k)$.

From Lemma 2.2.5, $\ell_j(f_j) - \ell_k(f_k)$ equals $(1/2)(b_j - b_k)$. However, from the assumption, $(b_j - b_k) \leq 0$ which implies that $\ell_j(f_j) \leq \ell_k(f_k)$. Hence the result follows. □

2.2.1 How to Find an Optimal Flow

In this subsection, we demonstrate how to mathematically compute an optimal flow of a given network G.

For general latency functions, we can find an optimal flow by solving the problem $\operatorname{argmin}_{\mathbf{f}} C(\mathbf{f})$. Using linear programming technique and by assuming all cost functions are convex and differentiable, the solution to $\operatorname{argmin}_{\mathbf{f}} C(\mathbf{f})$ can occur either at its *stationary points* or at points on its domain boundary. A stationary point is a point where the derivative is equal to zero. (see e.g. [MT03] for more details).

Let us demonstrate this with a network of two parallel links. For this case, the problem of finding the optimal flow $\mathbf{f}^{\mathrm{SE}} = (f_1^{\mathrm{SE}}, f_2^{\mathrm{SE}})$ can be reduced to the problem of finding only f_1^{SE}, since, knowing f_1^{SE}, we can obtain f_2^{SE} with $1 - f_1^{\mathrm{SE}}$. Since we assume that the total flow is one, the optimal flow must be $f_1^{SE} = 0$ or $f_1^{\mathrm{SE}} = 1$ or the critical point where $\partial C(\mathbf{f})/\partial f_1 = 0$. The following steps summarise how to compute an optimal flow:

1. Let for $(f_1)^*$ be the solution to $\partial C(\mathbf{f})/\partial f_1 = 0$.
2. If $(f_1)^*$ is feasible, i.e. $0 \leq (f_1)^* \leq f$ then $f_1^{\mathrm{SE}} = (f_1)^*$.
3. If $(f_1)^*$ is infeasible, i.e. $f_1^{\mathrm{SE}} > 1$ or $f_1^{\mathrm{SE}} < 0$, then either $(f_1^{\mathrm{SE}}) = 0$ or $(f^{\mathrm{SE}}) = 1$ whichever has the lower cost.

Alternatively, we can use Proposition 2.2.2 to find an optimal flow. Again, let us demonstrate this with a network of two parallel links.

1. Compute the marginal cost function $\ell_1'(f_1)$ and $\ell_2'(f_2)$.
2. Choose f_1^{SE} and f_2^{SE} such that $\ell'(f_j^{\text{SE}}) = \ell'(f_k^{\text{SE}})$.

Roughly speaking, to find an optimal flow using Proposition 2.2.2 is to solve for the optimal flow on link j f_j^{SE} such that the marginal costs of every link j that carries non-zero flow f_j^{SE} are equal, i.e. $\ell_j'(f_j^{\text{SE}})$ are equal for every $j \in E$ for which $f_j^{\text{SE}} > 0$.

Specifically for a network with linear latency functions, we can use the property stated in Lemma 2.2.5 to find an optimal flow. Assuming that links are sorted in the increasing order of b_j, we describe the process of obtaining an optimal flow in a network of n parallel links as follows.

Imagine an optimal flow increasing from 0 to 1. Initially all the flow will be put on link 1, i.e. the link with the lowest constant cost. If link 1 is a fixed cost link then all the flow will be allocated to link 1. Or else, according to Lemma 2.2.5, once the flow is increased to the point that the latency in link 1 is half the difference of the constant cost between link 2 and link 1, that is $\ell_1(f_1^{\text{SE}}) = (b_2 - b_1)/2$, the optimal flow starts using link 2. At this point, the amount of flow of f^{SE} has increased to $(b_2 - b_1)/(2a_1)$ and assigned only to link 1. If link 2 is a fixed cost all further flow will be routed on link 2. Otherwise, further flow will be assigned to both of the first two links, at the rate that will make those links continue to be increased with equal latency. And once the latency of link 2 increases to $(b_3 - b_2)/2$, the optimal flow starts using link 3. The process continues until all of f^{SE} is assigned.

The optimal flow prefers links with low constant cost. The following observation indicates which links will carry the optimal flow.

Observation 2.2.7. *Let G be a parallel-link network with linear latency functions. Let j be the link with the maximal index in the optimal flow of G which has $f_j^{SE} > 0$. Then*

- *$j = j'$ where j' is maximal with $\sum_{k=1}^{j'} (b_{j'} - b_k)/2a_{j'} \leq f$ and k is not a fixed latency link, i.e. $\ell_k = a_k f_k + b_k$ where $a_k \neq 0$. Otherwise,*
- *$j = j'$ where j' is minimal with $\ell_{j'} = b_{j'}$.*

Example 2.2.8. Suppose we are given a two-parallel-link network. Link 1 has the latency function of $\ell_1(f_1) = 2f_1$ and link 2 has the latency function of $\ell_2(f_2) = 1$. We need to route a flow of one unit from a source to a destination through link 1 and link 2 with the minimal possible cost. From Definition 2.2.1, the marginal cost of link 1 is $4f_1$ and 1 for link 2. Hence, if we route a quarter of the flow on link 1 and the rest on link 2 then, by Proposition 2.2.2, the flow is socially optimal.

Remark 2.2.9. We should point out that the computation of an optimal flow for network with linear latency functions can be done efficiently (in polynomial time). That is, we solve the problem $C(\mathbf{f})/\partial\mathbf{f}$ by linear programming.

2.3 Nash Flows

In this section, we investigate a flow at Nash equilibrium (NE). We shall denote a flow at Nash equilibrium (which we sometimes call Nash flow) by $\mathbf{f}^{\mathrm{NE}}$.

Definition 2.3.1. *Let G be a network of m players. A flow $\mathbf{f}^* = \left((\mathbf{f}^1)^*, \ldots, (\mathbf{f}^m)^*\right)$ of G is at* Nash equilibrium *if, for every player i,*

$$\begin{aligned} &C^i\left((\mathbf{f}^1)^*, \ldots, (\mathbf{f}^{i-1})^*, (\mathbf{f}^i)^*, (\mathbf{f}^{i+1})^*, \ldots, (\mathbf{f}^m)^*\right) \\ &\quad \leq C^i\left((\mathbf{f}^1)^*, \ldots, (\mathbf{f}^{i-1})^*, \mathbf{f}^i, (\mathbf{f}^{i+1})^*, \ldots, (\mathbf{f}^m)^*\right) \end{aligned} \tag{2.5}$$

for every feasible strategy $\mathbf{f}^i$.

We can look at a Nash equilibrium as the best strategy that each player can play based on the given set of strategies of the other players. Such the strategy is known as *best response strategy* which we give the formal definition in the following definition.

Definition 2.3.2. *A strategy of player i $(\mathbf{f}^i)^*$ is a* best response *for a given flow $\mathbf{f} = \left(\mathbf{f}^1, \ldots, \mathbf{f}^m\right)$ if and only if*

$$C^i\left(\mathbf{f}^1, \ldots, \mathbf{f}^{i-1}, (\mathbf{f}^i)^*, \mathbf{f}^{i+1}, \ldots, \mathbf{f}^m\right) \leq C^i\left(\mathbf{f}^1, \ldots, \mathbf{f}^{i-1}, \mathbf{f}^i, \mathbf{f}^{i+1}, \ldots, \mathbf{f}^m\right) \tag{2.6}$$

for every feasible strategy $\mathbf{f}^i$ of player i.

Corollary 2.3.3. *A flow $\mathbf{f}$ is at* Nash equilibrium *if and only if every player i routes using the best response strategy $(\mathbf{f}^i)^*$.*

From Definition 2.3.3, finding a Nash equilibrium is equivalent to every player i solving the problem

$$\underset{\mathbf{f}^i}{\operatorname{argmin}}\, C^i\left(\mathbf{f}^1, \ldots, \mathbf{f}^{i-1}, \mathbf{f}^i, \mathbf{f}^{i+1}, \ldots, \mathbf{f}^m\right) \tag{2.7}$$

while holding other strategies $\mathbf{f}^k$ fixed for every $k \neq i$.

Following from [HTW06], since we assume that cost functions are convex, we can state the convex optimality condition that a Nash flow must satisfy. Consider a link j and a player i. The cost of i on j is $f_j^i \cdot \ell_j(f_j)$. Increasing i's flow on j increases the cost of i at the rate of $\ell_j(f_j) + f_j^i \cdot \ell_j'(f_j)$ where $\ell_j'(f_j)$ is a marginal cost function of ℓ_j as defined in Definition 2.2.1.

Lemma 2.3.4. *[HTW06] A flow $\mathbf{f}$ is at Nash equilibrium, for all links j and k and for all players i that have $f_j^i > 0$, if the following inequality holds,*

$$\ell_j(f_j) + f_j^1 \cdot \ell_j'(f_j) \leq \ell_k(f_k) + f_k^1 \cdot \ell_k'(f_k).$$

It is crucial that, when we talk about a Nash equilibrium, we should clarify the existence and uniqueness of it. The following proposition was first proved by Harker [Har85] using the theory of variational inequalities, and subsequently shown by Orda et al. in [ORS93]. The proof relies on the assumption that the cost function must be convex. We shall omit the proof (for the complete proof see [ORS93]).

Proposition 2.3.5. *[HM85, ORS93] Let G be a parallel-link network for m atomic-splittable players with continuous, differentiable, convex cost functions. A Nash flow of G must exist.*

Remark 2.3.6. A network in our setting could admit more than one Nash flow (See Example 2.3.7), it is easy to see that all Nash flow must have the same cost. Borrowing term from [Rou02], we claim that Nash flow in our setting is *essentially unique*.

Example 2.3.7. A network of links parallel links. Both links have a constant cost of 1. It is easy to see that, regardless of how many players in the network, there are more than one Nash flows. However every Nash flow has a social cost of exactly 1.

Next, let us consider the Nash equilibrium of a flow $\mathbf{f}$ of a special case of two players. Given a network of two players: player 1 and player 2, a flow $\mathbf{f}^* = \left((\mathbf{f}^1)^*, (\mathbf{f}^2)^*\right)$ is at Nash equilibrium if,

$$C^1\left((\mathbf{f}^1)^*, (\mathbf{f}^2)^*\right) \leq C^1\left(\mathbf{f}^1, (\mathbf{f}^2)^*\right) \text{ and } C^2\left((\mathbf{f}^1)^*, (\mathbf{f}^2)^*\right) \leq C^2\left((\mathbf{f}^1)^*, \mathbf{f}^2\right) \tag{2.8}$$

for every feasible strategies $\mathbf{f}^1$ and $\mathbf{f}^2$.

This is equivalent to solving two problems, one for each player. Assuming that player 2 will play with the feasible strategy $(\mathbf{f}^2)'$, player 1 solves the problem

$$\operatorname*{argmin}_{\mathbf{f}^1} C^1\left(\mathbf{f}^1, (\mathbf{f}^2)'\right).$$

Similarly, assuming player 1 will play with the feasible strategy $(\mathbf{f}^1)'$, player 2 solves the problem

$$\operatorname*{argmin}_{\mathbf{f}^2} C^2\left((\mathbf{f}^1)', \mathbf{f}^2\right).$$

Suppose that player 1 and player 2 obtain $(\mathbf{f}^1)^*$ and $(\mathbf{f}^2)^*$ respectively from solving the above problems. $(\mathbf{f}^1)^*$ (respectively $(\mathbf{f}^2)^*$) is essentially the best response to $(\mathbf{f}^2)'$ (respectively $(\mathbf{f}^1)'$). If $(\mathbf{f}^1)^* = (\mathbf{f}^1)'$ and $(\mathbf{f}^2)^* = (\mathbf{f}^2)'$ then the condition in (2.8) is satisfied, hence the flow $((f^1)^*, (f^2)^*)$ is a Nash flow.

We have so far considered the case of atomic players. It is useful, especially for the final section of this chapter, to formulate a formal definition of a Nash equilibrium of a flow of nonatomic players. Following from [CP91], let $\mathbf{f}^{\mathrm{UE}}$ denote a flow of infinitely

many players, each of which controls a negligible fraction of the total flow (We sometimes refer to this type of flow as UE flow). Each player in $\mathbf{f}^{\mathrm{UE}}$ follows the Wardrop's user equilibrium principle in its routing decision, which states that, at Nash equilibrium, the latencies in all links actually used are equal and less than those which would be experienced by a single player on any unused link. The following definition formally summarised this idea.

Definition 2.3.8. *[RT02] Let G be a network of nonatomic players. A flow $\mathbf{f}$ of G is at Nash equilibrium if and only if for every links j and $k \in E$ with $f_j > 0$, $\ell_j(f_j) \leq \ell_k(f_k)$.*

2.3.1 How to Find a Nash Flow

In this subsection, we demonstrate how to compute a Nash flow $\mathbf{f}^{\mathrm{NE}}$ for a two-player game in a network with two parallel links. As mentioned earlier, finding the Nash flow $\mathbf{f}^{\mathrm{NE}} = \left((\mathbf{f}^1)^{\mathrm{NE}}, (\mathbf{f}^2)^{\mathrm{NE}}\right)$ of the two-player flow is equivalent to player 1 solving the problem

$$\operatorname*{argmin}_{\mathbf{f}^1} C^1\left(\mathbf{f}^1, (\mathbf{f}^2)^{\mathrm{NE}}\right),$$

and player 2 solves the problem

$$\operatorname*{argmin}_{\mathbf{f}^2} C^2\left((\mathbf{f}^1)^{\mathrm{NE}}, \mathbf{f}^2\right).$$

Since, in two-link networks, knowing the amount of a player flow on one link obviously grants us the player flow on the other link, we shall use f_1^i instead of $\mathbf{f}^i$ for the strategy of player i. Since when solving the problem $\operatorname{argmin}_{\mathbf{f}^i} C^i$ for player i, the other player's flow is fixed, we can use the derivative technique that we described for the problem of finding an optimal flow.

Thus, for a two-player case, we will have two conditions so that, if $\mathbf{f}^{\mathrm{NE}}$ satisfies, both conditions will be necessary and sufficient for $\mathbf{f}^{\mathrm{NE}}$ to be a Nash flow. Let us demonstrate this with a simple two parallel-link network with linear latency functions in the following example.

Example 2.3.9. Suppose we are given a network of two parallel links. Link 1 has the latency of $\ell_1 = 2f_1$ and link 2 has the latency of $\ell_2 = 1$. In addition, we are given a flow of two players, each of which has a $1/2$ unit of flow to route from the sink node to the source node using link 1 and/or link 2. First, we solve the problem $\operatorname{argmin}_{\mathbf{f}^1} C^1\left(\mathbf{f}^1, \mathbf{f}^2\right)$. From Definition 2.1.1, player 1's cost is

$$C^1 = f_1^1 \cdot 2(f_1^1 + f_1^2) + (\frac{1}{2} - f_1^1) \cdot 1$$

because $f_2^1 = 1/2 - f_1^1$. The partial derivative of C^1 with respect to f_1^1 is

$$\frac{\partial C^1}{\partial f_1^1} = 4f_1^1 + 2f_1^2 - 1.$$

Equating the result to zero, we get

$$4f_1^1 + 2f_1^2 - 1 = 0 \tag{2.9}$$

We then solve the problem $\operatorname{argmin}_{\mathbf{f}^2} C^2\left(\mathbf{f}^1, \mathbf{f}^2\right)$. The cost of player 2 can be written as

$$C^2 = f_1^2 \cdot 2(f_1^1 + f_1^2) + (\frac{1}{2} - f_1^2) \cdot 1,$$

because $f_1^2 = 1/2 - f_1^2$. The partial derivative of C^2 with respect to f_1^2 is

$$\frac{\partial C^1}{\partial f_1^1} = 4f_1^2 + 2f_1^1 - 1.$$

Setting the result to zero, we have

$$4f_1^2 + 2f_1^1 - 1 = 0 \tag{2.10}$$

For a flow of both players to be the Nash flow, both (2.9) and (2.10) must be satisfied. Solving both equations, yields $f_1^1 = f_1^2 = 1/6$. Because both f_1^1 and f_1^2 are feasible (positive and less than the corresponding player's flow) the flow $((1/6, 1/3), (1/6, 1/3))$ is the Nash flow.

2.4 Stackelberg Routing

In this section, we consider another concept of equilibrium. Let us suppose that one of the players in M acts as a leader that will have a priority to route its flow before others. After the leader chooses its strategy, the rest of the players—followers—react to the strategy, trying to minimise their individual cost with latency functions that have been modified by the leader, reaching the Nash equilibrium relative to it. In game theory, this is the concept called Stackelberg games. To clarify, in Stackelberg games, the leader is not necessarily selfish. As we have mentioned earlier in the Previous Works section, there are other related works that consider both benevolent leaders and malicious leaders (see Section 1.3 for more details).

Our focus is on a selfish leader. If the leader's strategy that will optimise the leader's cost after all players have played is $\mathbf{f}^1 = \left(f_1^1, \ldots, f_n^1\right)$, the followers will take the

leader's flow as fixed and minimise their cost with respect to the modified cost—namely $\tilde{\ell}_j(f_j) = \ell_j(f_j + f_j^1)$ for each j. A flow is at selfish Stackelberg leadership equilibrium (or SSL equilibrium) in two cases. The first is where the number of players $m > 2$, after the leader has played, the followers form an $m-1$ Nash equilibrium relative to $\tilde{\ell}_j(f_j)$. The second where there is a single follower, the follower forms an optimal flow relative to $\tilde{\ell}_j(f_j)$. Note that we mainly study SSL equilibrium in the second case. We denote such a flow as selfish Stackelberg leadership flow (or SSL flow or $\mathbf{f}^{\mathrm{SSL}}$). We formalise the idea of the SSL equilibrium in the following definition.

Definition 2.4.1. *Given a network G, in which player 1 is a leader and the rest of the players (2 to m) are followers, all of which route their flow simultaneously. Let player 1's strategy be $\mathbf{f}^1 = \left(f_1^1, \ldots, f_n^1\right)$ that optimise C^1, and let $\tilde{\ell}_j(f_j) = \ell_j(f_j + f_j^1)$ for each link $j \in E$. The flow $\mathbf{f}$ is at a selfish Stackelberg leadership equilibrium if and only if the flow $\mathbf{f}'$ of players 2 to m is at Nash equilibrium, or, in the case where $m = 2$, $\mathbf{f}'$ of player 2 is at system equilibrium, with respect to a latency $\tilde{\ell}_j(f_j)$ for each $j \in E$.*

Since after the leader has played the modified latency functions retain the property to be continuous, differentiable and convex, the existence of the SSL equilibria is guaranteed by Proposition 2.3.5.

Proposition 2.4.2. *Let G be a parallel-link network of m atomic-splittable players with continuous differentiable convex cost functions. The flow of G at SSL equilibrium must exist.*

2.4.1 The Price of Selfish Stackelberg Leadership

Now that we have defined the SSL equilibrium and the Nash equilibrium, we are ready to give the formal definition of the price of selfish Stackelberg leadership that we use to quantify the additional social cost that occurs when one of the players acts as a leader. Informally, the price of SSL is the worst ratio between the social cost at SSL equilibria and that of SSL equilibria.

Definition 2.4.3. *Let G be a parallel-link network of m atomic-splittable players. Let $\mathbf{f}^{SSL}$ and $\mathbf{f}^{NE}$ respectively be the SSL flow and the Nash flow of G. If $\mathbf{F}$ is the set of all feasible flows of G, the* price of selfish Stackelberg leadership *in G is defined as*

$$\sup_{\mathbf{f}^{SSL}, \mathbf{f}^{NE} \in \mathbf{F}} \frac{C\left(\mathbf{f}^{SSL}\right)}{C\left(\mathbf{f}^{NE}\right)}. \tag{2.11}$$

Remark 2.4.4. The definition focuses on the additional social cost of SSL setting over Nash setting; so we do not define it as $C(\mathbf{f}^{\mathrm{SSL}})/C(\mathbf{f}^{\mathrm{SE}})$ where $\mathbf{f}^{\mathrm{SE}}$ denote the optimal

flow of the corresponding $\mathbf{f}$. However in later chapters, specifically in Chapter 4, we often use the ratio $C(\mathbf{f}^{\mathrm{SSL}})/C(\mathbf{f}^{\mathrm{SE}})$ to upper bound the price of SSL.

2.4.2 How to Find a SSL Flow

In this subsection, we demonstrate how to compute a SSL flow for a network of two parallel links and two players. To compute the SSL flow $\mathbf{f}^{\mathrm{SSL}}$, we first have to determine what the second player will do.

The Last Player's Behaviour

We have noted earlier that the last player in a Stackelberg game will act similar to an optimal flow. Suppose that player 1 routes with the strategy $\mathbf{f}^1 = (f_1^1, \ldots, f_n^1)$. After player 1 has played, the latency function on each link has changed to $\tilde{\ell}_j(f_j) = \ell_j(f_j + f_j^1)$. Thus, player 2 now has to solve the problem

$$\operatorname*{argmin}_{\mathbf{f}^2} \sum_{j \in E} f_j^2 \cdot \tilde{\ell}_{(f_j)}.$$

Solving the above problem, player 2 obtains $\mathbf{f}^2$ which essentially is the best response strategy of player 2 with respect to any player 1's strategy $\mathbf{f}^1$.

Alternatively, we can use Proposition 2.2.2, and find the strategy of player 2 such that all of the marginal costs of the links used are equal.

Proposition 2.4.5. *Let G be a parallel-link network with two players. In the SSL setting, assume that the leader routes its flow with $\mathbf{f}^1$, and let $\tilde{\ell}_j(f_j) = \ell_j(f_j + f_j^1)$ for all links j. Player 2's cost is minimised if for every pair of links j and k with $f_j^2 > 0$, $\tilde{\ell}'_j(f_j^2) \leq \tilde{\ell}'_k(f_k^2)$.*

The next step would be to determine what the first player would do.

The First Player's Behaviour

Let us denote $\left(\mathbf{f}^2\right)'(\mathbf{f}^1)$ the solution to the problem $\operatorname{argmin}_{\mathbf{f}^2} \sum_{j \in E} f_j^2 \cdot \tilde{\ell}_{(f_j)}$ for any given $\mathbf{f}^1$. The objective of player 1 now is to choose $\mathbf{f}^1$ such that the combined flow gives the minimal-possible cost of player 1. That is, player 1 solves the problem

$$\operatorname*{argmin}_{\mathbf{f}^1} C^1(\mathbf{f}^1, \left(\mathbf{f}^2\right)'(\mathbf{f}^1)).$$

The next example shows how to find the SSL flow in the two parallel link network with linear latency functions.

Example 2.4.6. Consider the network in Example 2.3.9. That is, we are given a network of two parallel links where link 1 has the latency function of $2f_1$ and link 2 has the latency function of 1. We are also given a flow of two players, each of which has $1/2$ unit of flow to route from the source to the sink in the network. Suppose player 1 routes its flow before player 2. We find the SSL flow in the network.

Firstly, we consider player 2's behaviour. If player 1 routes with the strategy $\mathbf{f}^1 = (f_1^1, f_2^1)$, the cost of player 2 is

$$C^2 = f_1^2 \cdot 2(f_1^1 + f_1^2) + (\frac{1}{2} - f_1^2) \cdot 1,$$

because $f_2^2 = 1/2 - f_1^2$. The partial derivative of C^2 with respect to f_1^2 is

$$\frac{\partial C^2}{\partial f_1^2} = 4f_1^2 + 2f_1^1 - 1.$$

Equating the result to zero before solving for f_1^2, we have

$$f_1^2 = \frac{1 - 2f_1^1}{4}. \tag{2.12}$$

Secondly, we consider what player 1 will do. Player 1 knows that if it routes with f_1^1 then player 2 will respond with f_1^2 according to (2.12). Hence, the cost of player 1 is

$$C^1 = f_1^1 \cdot 2(f_2^1 + \frac{1 - 2f_1^1}{4}) + (\frac{1}{2} - f_1^1) \cdot 1.$$

by $f_2^1 = 1/2 - f_1^1$ and (2.12). The partial derivative of C^1 with respect to f_1^1 is

$$\frac{\partial C^1}{\partial f_1^1} = 2f_1^1 - \frac{1}{2}.$$

Setting the derivative to zero and then solve for f_1^1, we obtain f_1^1 of $1/4$. Furthermore, from (2.12), f_1^2 is $1/8$. Hence the SSL flow is $((1/4, 1/4), (1/8, 3/8))$.

Combining Examples 2.3.9 and 2.4.6, we show in the following example the price of SSL of the network discussed in the previous example.

Example 2.4.7. Consider the social costs obtained in Example 2.3.9 and Example 2.4.6. The social cost of the SSL flow is $29/32$ and that of the Nash flow is $8/9$. Thus, we have the price of SSL of approximately 1.019.

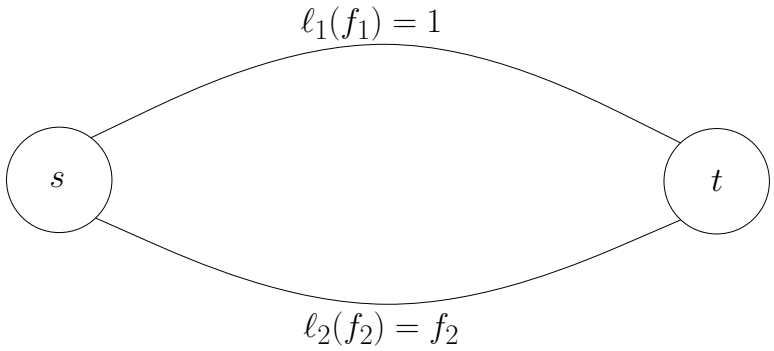


Figure 2.1: Pigou's example shows a simple two-node, two-parallel-link network. A latency function $\ell(x)$ describes the cost experienced by a player for using that link.

2.5 Pigou's Example

In this section, we illustrate the definitions, characterisations and techniques of the previous sections in a simple network. We hope that this will not only motivate the main problem studied in this thesis, but also distinguish between our work and other works studying Stackelberg routing games. We consider the network depicted in Figure 2.1. The network was first studied by Pigou in 1920 [Pig20].

Suppose that the latency function of link 1 is $\ell_1(f_1) = 1$ and the latency of link 2 is $\ell_2(f_2) = f_2$. Suppose the total traffic flow is 1, and, in the case where there are two players, player 1's flow f^1 is $2/5$ and player 2's flow f^2 is $3/5$. Assume that all players aim to minimise an individual cost, unless stated otherwise, of routing a flow from s to t by splitting their flow across the links. We compute the routing strategies, which are equivalent, to obtain f_1 for the one player game and f_1^1 and f_1^2 for the two-player game for the following set-ups:

1. A system equilibrium.
2. A user equilibrium.
3. Two atomic players in a Nash equilibrium.
4. Two atomic players in the selfish Stackelberg leadership setting.
5. An atomic leader and a UE follower.
6. A benevolent atomic leader and an atomic follower.
7. A benevolent atomic leader and a UE follower.
8. A malicious atomic leader and an atomic follower.

9. A malicious atomic leader and a UE follower.

We investigate each set-up in the order listed. In each set-up, we compute the routing strategy for each player, its individual cost and the social cost. Table 2.1 summarises the results at the end of this section.

2.5.1 A System Equilibrium

In this first scenario, we compute the optimal flow of the network. Since the latencies of the network are affine linear, from Lemma 2.2.3, the strategy of the optimal flow is to make the terms $2a_j + b_j$ the same for both links. That is, $2f_2$ from link 2 must equal to 1 from link 1. Routing half of the flow on link 1 and the other half on link 2 equalises the terms. Hence the optimal flow is $(1/2, 1/2)$, which gives the social cost of

$$C\left(\mathbf{f}^{\mathrm{SE}}\right) = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}.$$

2.5.2 A User Equilibrium

Next, we consider the nonatomic setting. That is, we investigate the UE flow which has an infinite number of players, each of which controls a negligible fraction of the overall flow. By Definition 2.3.8, the UE flow $\mathbf{f}^{\mathrm{UE}}$ is at Nash equilibrium if and only if $\mathbf{f}^{\mathrm{UE}}$ routes all its flow along link 2 which makes the latency of link 1 and link 2 equal.

Alternatively, we can describe $\mathbf{f}^{\mathrm{UE}}$ as follows. For each selfish nonatomic player to ensure the minimal individual cost possible, it routes its flow on link 2. This is because the cost of using link 2 is never worse than the cost of using link 1 even when link 2 is loaded with all the traffic. (The cost of using link 1 is always 1 and the cost of using link 2 when the entire flow is routed on it is 1.) And it is even better when there are some foolish players routing their flow on link 1.

In conclusion, by Definition 2.1.1, the social cost is,,

$$C(\mathbf{f}^{\mathrm{UE}}) = f_1 \cdot \ell_1(f_1) + f_2 \cdot \ell_2(f_2) = 0 \cdot 1 + 1 \cdot 1 = 1.$$

Note that, for any UE flow that is bigger than one in this network, one unit of that flow will be routed on link 2 and the rest will be on link 1. This is because if the flow on link 2 is less than the flow on link 1 then some players on link 1 will be better off deviating to link 2, and if already one unit of flow is on link 2, then any player deviating from link 1 to link 2 will be worse off.

2.5.3 Two Atomic Players at Nash Equilibrium

In this subsection, we consider the scenario in which there are two atomic-splittable players. Recall that player 1 has 2/5 of the overall flow and the rest of the flow belongs to player 2. To find the Nash equilibrium for this set-up, by Definition 2.3.3, basically we want to find a pair of feasible strategies $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ respectively for player 1 and player 2 such that

$$C^1\left((f_1^1)^{\mathrm{NE}}, (f_1^2)^{\mathrm{NE}}\right) \leq C^1\left(f_1^1, (f_1^2)^{\mathrm{NE}}\right)$$

and

$$C^2\left((f_1^1)^{\mathrm{NE}}, (f_1^2)^{\mathrm{NE}}\right) \leq C^2\left((f_1^1)^{\mathrm{NE}}, f_1^2\right)$$

for every feasible strategy f_1^1 and f_1^2.

That is, we must solve the optimisation problems $\operatorname{argmin}_{f_1^1} C^1(f_1^1, (f_1^2)^{\mathrm{NE}})$ while holding $(f_1^2)^{\mathrm{NE}})$ fixed and $\operatorname{argmin}_{f_1^2} C^2((f_1^1)^{\mathrm{NE}}, f_1^2)$ while holding $(f_1^1)^{\mathrm{NE}})$. The cost of player 1 and player 2 can be explicitly written as

$$C^1(f_1^1, (f_1^2)^{\mathrm{NE}}) = f_1^1 \cdot 1 + (f^1 - f_1^1)(f^1 - f_1^1 + f^2 - (f_1^2)^{\mathrm{NE}}),$$

and

$$C^2((f_1^1)^{\mathrm{NE}}, f_1^2) = f_1^2 \cdot 1 + (f^2 - f_1^2)(f^1 - (f_1^1)^{\mathrm{NE}} + f^2 - f_1^2).$$

Substituting $f^1 = 2/5$ and $f^2 = 3/5$, we get

$$C^1 = f_1^1 + (\frac{2}{5} - f_1^1)(1 - f_1^1 - (f_1^2)^{\mathrm{NE}}),$$

and

$$C^2 = f_1^2 + (\frac{3}{5} - f_1^2)(1 - (f_1^1)^{\mathrm{NE}} - f_1^2).$$

The partial derivative of C^1 and C^2 with respect to f_1^1 and f_1^2 respectively are

$$\begin{aligned}\frac{\partial C^1}{\partial f_1^1} &= 1 + (\frac{2}{5} - f_1^1)(-1) + (-1)(1 - f_1^1 - (f_1^2)^{\mathrm{NE}}) \\ &= 2f_1^1 + (f_1^2)^{\mathrm{NE}} - \frac{2}{5}, \end{aligned} \tag{2.13}$$

and

$$\begin{aligned}\frac{\partial C^2}{\partial f_1^2} &= 1 + (\frac{3}{5} - f_1^2)(-1) + (-1)(1 - (f_1^1)^{\mathrm{NE}} - f_1^2) \\ &= 2f_1^2 + (f_1^1)^{\mathrm{NE}} - \frac{3}{5}. \end{aligned} \tag{2.14}$$

Setting (2.13) and (2.14) to zero, we then can solve for $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$. We have

$$(f_1^1)^{\mathrm{NE}} = \frac{2 - 5(f_1^2)^{\mathrm{NE}}}{10}, \tag{2.15}$$

and

$$(f_1^2)^{\mathrm{NE}} = \frac{3 - 5(f_1^1)^{\mathrm{NE}}}{10}. \tag{2.16}$$

In order for the pair of strategies $(\mathbf{f}^1)^{\mathrm{NE}}$ and $(\mathbf{f}^2)^{\mathrm{NE}}$ to be at Nash equilibrium, both (2.15) and (2.16) must be satisfied. Solving the pair of equations, we get $(f_1^1)^{\mathrm{NE}} = 1/15$ and $(f_1^2)^{\mathrm{NE}} = 4/15$. Thus we get the following costs:

$$C^1 = \frac{1}{15} \cdot 1 + \frac{1}{3}(\frac{1}{3} + \frac{1}{3}) = \frac{13}{45} \approx 0.289,$$

and

$$C^2 = \frac{4}{15} \cdot 1 + \frac{1}{3}(\frac{1}{3} + \frac{1}{3}) = \frac{22}{45} \approx 0.489.$$

And the social cost is

$$C = \frac{13}{45} + \frac{22}{45} = \frac{7}{9} \approx 0.778.$$

2.5.4 Two Atomic Players in the SSL Setting

In this subsection, we consider the same set-up as in the previous Subsection 2.5.3, except that, in this subsection the game is played in the SSL setting—one of the players acts as a selfish leader, routing its flow first and has to commit to the strategy it chooses. The other player then allocates its flow to minimise its cost according to the first player's choice. We consider two scenarios in this setting. One is when player 1 is the leader and the other is when player 2 is the leader. We determine the SSL flow $\mathbf{f}^{\mathrm{SSL}}$ which as noted earlier it is sufficient to find f_1^1 and f_1^2 at SSL equilibrium.

Player 1 as a leader

Suppose player 1 is a leader. To find what player 1 will do, we have to first determine what player 2's strategy is. Player 2 will have to solve the problem, given that player 1 previously chooses strategy f_1^1, $\operatorname{argmin}_{f_1^2} C^2(f_1^1, f_1^2)$. The cost of player 2 is

$$C^2(f_1^1, f_1^2) = f_1^2 \cdot 1 + (f^2 - f_1^2)(f^1 - f_1^1 + f^2 - f_1^2).$$

Substituting $f^1 = 2/5$ and $f^2 = 3/5$, we get

$$C^2 = f_1^2 + (\frac{3}{5} - f_1^2)(1 - f_1^1 - f_1^2).$$

Taking the partial derivative of C^2 with respect to f_1^2, we get

$$\begin{aligned}\frac{\partial C^2}{\partial f_1^2} &= 1 + (\frac{3}{5} - f_1^2)(-1) + (-1)(1 - f_1^1 - f_1^2) \\ &= 2f_1^2 + f_1^1 - \frac{3}{5}.\end{aligned}$$

Setting the result to zero then solving for f_1^2, we have

$$f_1^2 = \frac{3 - 5f_1^1}{10}. \tag{2.17}$$

Secondly, we determine f_1^1 in the SSL flow. Player 1 solves the problem $\operatorname{argmin}_{f_1^1} C^1(f_1^1, f_1^2)$, knowing that if player 1 routes with f_1^1, player 2 will respond with f_1^2 according to (2.17). The cost of player 1 is

$$C^1(f_1^1, f_1^2) = f_1^1 \cdot 1 + (f^1 - f_1^1)(f^1 - f_1^1 + f^2 - f_1^2).$$

Substituting (2.17) for f_1^2, $f^1 = 2/5$ and $f^2 = 3/5$, we get

$$C^1 = f_1^1 + (\frac{2}{5} - f_1^1)(\frac{7 - 5f_1^1}{10}).$$

The derivative of C^1 with respect to f_1^1 is

$$\begin{aligned}\frac{\partial C^1}{\partial f_1^1} &= 1 + (\frac{2}{5} - f_1^1)(\frac{1}{2}) + (-1)(\frac{7 - 5f_1^1}{10}) \\ &= f_1^1 + \frac{1}{10}\end{aligned}$$

Setting the derivative to zero and solving for f_1^1, we obtain the infeasible f_1^1 of $-1/10$. Since the cost of player 1 is convex, the feasible f_1^1 in the SSL flow must be equal to either 0 or $2/5$. When f_1^1 equals to 0 yielding f_1^2 of $3/10$ (using (2.17)) which gives the cost of player 1 of $0 \cdot 1 + (2/5)(2/5 + 3/10) \approx 0.28$. When f_1^1 equals to $2/5$ yielding f_1^2 of $1/10$ which gives the cost of player 1 of $2/5 \cdot 1 + 0 \cdot (0 + 1/2) = 0.4$. Hence, the SSL flow is $(0, 3/10)$. And we have the cost of player 2 of $(3/10) \cdot 1 + 3/10(2/5 + 3/10) = 0.51$. Finally, the social cost is $0.28 + 0.51 \approx 0.79$.

Player 2 as a leader

Suppose player 2 is a leader. Firstly, we determine what player 1 will do. Given that player 2 previously routes with a strategy f_1^2, player 1 solves the problem $\operatorname{argmin}_{f_1^1} C^1(f_1^1, f_1^2)$. The cost of player 1 is

$$C^1(f_1^1, f_1^2) = f_1^1 \cdot 1 + (f^1 - f_1^1)(f^1 - f_1^1 + f^2 - f_1^2).$$

By substituting $f^1 = 2/5$ and $f^2 = 3/5$, we get

$$C^1 = f_1^1 + (\frac{2}{5} - f_1^1)(1 - f_1^1 - f_1^2).$$

Taking a partial derivative of C^1 with respect to f_1^1, we get

$$\begin{aligned}\frac{\partial C^1}{\partial f_1^1} &= 1 + (\frac{2}{5} - f_1^1)(-1) + (-1)(1 - f_1^1 - f_1^2) \\ &= 2f_1^1 + f_1^2 - \frac{2}{5}.\end{aligned}$$

Setting the result to zero then solving for f_1^1, we have

$$f_1^1 = \frac{2 - 5f_1^2}{10}. \tag{2.18}$$

Secondly, player 2 solves the problem $\operatorname{argmin}_{f_1^2} C^2(f_1^1, f_1^2)$. Substituting (2.18) into C^2, we get the cost of player 2 as

$$\begin{aligned}C^2 &= f_1^2 + (\frac{3}{5} - f_1^2)(1 - \frac{2 - 5f_1^2}{10} - f_1^2) \\ &= f_1^2 + (\frac{3}{5} - f_1^2)(\frac{8 - 5f_1^2}{10}).\end{aligned}$$

The partial derivative of C^2 with respect to f_1^2 is

$$\begin{aligned}\frac{\partial C^2}{\partial f_1^2} &= 1 + (\frac{3}{5} - f_1^2)(-\frac{1}{2}) + (-1)(\frac{8 - 5f_1^2}{10}) \\ &= y - \frac{1}{10}.\end{aligned}$$

Equating the result to zero then solving the equation, we get $f_1^2 = 1/10$ therefore,

by (2.18), $f_1^1 = 3/20$. Thus, we have the following costs:

$$C^1 = \frac{3}{20} \cdot 1 + \frac{1}{4}(\frac{1}{4} + \frac{1}{2}) = \frac{27}{80} \approx 0.337,$$

and

$$C^2 = \frac{1}{10} \cdot 1 + \frac{1}{2}(\frac{1}{4} + \frac{1}{2}) = \frac{19}{40} = 0.475.$$

And the social cost is

$$C = \frac{27}{80} + \frac{19}{40} = \frac{13}{16} \approx 0.812.$$

2.5.5 An Atomic Leader and a UE Follower

In this setting, we consider the scenario in which the leader (player 1) is an atomic player and player 2 is a UE flow. Again, we first determine what the follower will do. If player 1 has 2/5 of the total flow then, by Definition 2.3.8, for *every* strategy of player 1, player 2 routes all of its flow on link 2.

For the strategy of player 1 at SSL equilibrium, player 1 solves the problem $\operatorname{argmin}_{f_1^1} C^1(f_1^1, f_1^2)$. By substituting $f_1^2 = 0$, $f^1 = 2/5$ and $f^2 = 3/5$, we have the cost of player 1 as

$$C^1 = f_1^1 \cdot 1 + (\frac{2}{5} - f_1^1)(1 - f_1^1).$$

The partial derivative of C^1 with respect to f_1^1 is

$$\begin{aligned}\frac{\partial C^1}{\partial f_1^1} &= 1 + (\frac{2}{5} - f_1^1)(-1) + (-1)(1 - f_1^1) \\ &= 2f_1^1 - \frac{2}{5}.\end{aligned}$$

Setting the derivative to zero then Solving the equation, we obtain $f_1^1 = 1/5$. The resulting flow is $((1/5, 1/5), (0, 3/5))$. We have the following costs:

$$C^1 = \frac{1}{5} \cdot 1 + \frac{1}{5}(\frac{1}{5} + \frac{3}{5}) = \frac{9}{25} = 0.36,$$

and

$$C^2 = 0 \cdot 1 + \frac{3}{5}(\frac{1}{5} + \frac{3}{5}) = \frac{12}{25} = 0.48.$$

The social cost is

$$C = \frac{9}{25} + \frac{12}{25} = \frac{21}{25} = 0.84.$$

2.5.6 A Benevolent Atomic Leader and an Atomic Follower

In this subsection, we consider the situation similar to Subsection 2.5.4, in which there are two atomic players in the SSL setting. However, in this subsection, we assume that the leader is benevolent. That is, the leader's objective is to allocate its flow to entice the other player to route its flow such that the social cost is minimal. Korilis et al. [KLO97a] studied the more general setting in which there are finitely many atomic followers. And, for this setting, they presented the leader strategy that can always enforce the optimal flow.

We first consider the situation where player 1 is a leader; then we consider the situation where player 2 is a leader.

Player 1 as a benevolent leader

Suppose player 1 is a leader. Recall from Example 2.5.1 that the optimal flow is $(1/2, 1/2)$. If player 1 routes its flow such that it does not change the latency difference between link 1 and link 2, i.e. $f_1^1 = 2/5$, then player 2, who will treat player 1's flow as constant, will act as an optimal flow. By Lemma 2.2.5, player 2 will route 1/10 on link 1 which makes the combined flow an optimal flow. Therefore we get the following costs:

$$C^1 = \frac{2}{5} \cdot 1 + 0(0 + \frac{1}{2}) = \frac{2}{5} = 0.4,$$

and

$$C^2 = \frac{1}{10} \cdot 1 + \frac{1}{2}(0 + \frac{1}{2}) = \frac{7}{20} = 0.35.$$

And the social cost is

$$C = \frac{2}{5} + \frac{7}{20} = \frac{3}{4} = 0.75.$$

Player 2 as a benevolent leader

Suppose player 2 is a leader. Similar to when player 1 is a leader, player 2 will try yo allocate its flow such that the latency different between link 1 and link 2 stays the same after its turn. That is, player 2 routes 1/2 unit of flow on link 1. Player 1 will act like an optimal flow, and, by Lemma 2.2.5, will route all of its flow on link 2. Consequently

the combined flow is an optimal flow. Thus, we get the following costs

$$C^1 = 0 \cdot 1 + \frac{2}{5} \cdot (\frac{2}{5} + \frac{1}{10}) = \frac{1}{5} = 0.2,$$

and

$$C^2 = \frac{1}{2} \cdot 1 + \frac{1}{10}(\frac{2}{5} + \frac{1}{10}) = \frac{11}{20} = 0.55.$$

And the social cost is

$$C = \frac{1}{5} + \frac{11}{20} = \frac{3}{4} = 0.75.$$

2.5.7 A Benevolent Atomic Leader and a UE Follower

In this subsection, we consider a similar setting as the previous subsection, but with player 2 as a UE flow. This set-up is commonly studied by many recent works in Stackelberg games for example [KS06, Rou04, Swa07].

As usual, we consider what player 2 will do first. For every strategies f_1^1, link 2 is always better for player 2. Hence, player 2 routes all its flow on link 2. Player 1 aim to minimise the social cost. After player 2 has played, the social cost is

$$\begin{aligned} C &= (f_1^1 + 0) \cdot 1 + (2/5 - f_1^1 - 3/5) \cdot (2/5 - f_1^1 - 3/5) \\ &= f_1^1 + (1 - f_1^1)^2. \end{aligned}$$

The partial derivative of C with respect to f_1^1 is

$$\frac{\partial C}{\partial f_1^1} = 2f_1^1 - 1.$$

Setting the derivative to zero then solving for f_1^1, we get an infeasible f_1^1 of $1/2$. Let us rewrite C as the following

$$C = (f_1^1)^2 - f_1^1 + 1$$

which essentially is a decreasing convex function for f_1^1 in the range $[0, 2/5]$. Therefore the feasible strategy of player 1 that minimise the social cost must be one of its border, namely f_1^1 equals to either 0 or $2/5$. Since when f_1^1 equals to $2/5$ the social cost is lower than that when f_1^1 equals to 0, player 1 routes all its flow on link 1. In conclusion, we

get the following costs

$$C^1 = \frac{2}{5} \cdot 1 + 0(0 + \frac{3}{5}) = \frac{2}{5} = 0.4,$$

and

$$C^2 = 0 \cdot 1 + \frac{3}{5} \cdot (0 + \frac{3}{5}) = \frac{9}{25} \approx 0.36.$$

And the social cost is

$$C = \frac{2}{5} + \frac{9}{25} = \frac{19}{25} = 0.76.$$

2.5.8 A Malicious Atomic Leader and an Atomic Follower

So far, we have considered a benevolent leader and a selfish leader. In this subsection, we consider a malicious leader. The objective of the leader is to seek, independently of its own cost, to degrade the cost of the total network. In this subsection, we consider the scenario in which player 1 is the malicious leader, and player 2 is an atomic follower. This is similar to the SSL setting in a sense that player 1 predicts what player 2 do then maximise the social cost with regard to the prediction of what player 2 will do. Firstly, we consider what player 2 will do. The cost of player 2 is

$$C^2(f_1^1, f_1^2) = \frac{3}{5} \cdot 1 + (\frac{3}{5} - f_1^2)(\frac{2}{5} - f_1^1 + \frac{3}{5} - f_1^2),$$

where $f^1 = 2/5$ and $f^2 = 3/5$. Taking the partial derivative of C^2 with respect to f_1^2, we have

$$\begin{aligned} \frac{\partial C^2}{\partial f_1^2} &= 1 + (\frac{3}{5} - f_1^2)(-1) + (-1)(1 - f_1^1 - f_1^2) \\ &= 2f_1^2 + f_1^1 - \frac{3}{5}. \end{aligned}$$

Setting the derivative to zero then solving for f_1^2, we have

$$f_1^2 = \frac{3 - 5f_1^1}{10}. \tag{2.19}$$

Secondly, we determine f_1^1 which is the solution to $\text{argmax}_{f_1^1} C(f_1^1, f_1^2)$ where f_1^2 is obtained from (2.19). Substituting (2.19) for f_1^2, the social cost C becomes

$$\begin{aligned} C &= (f_1^1 + f_1^2) \cdot 1 + (1 - f_1^1 - f_1^2)^2 \\ &= (f_1^1 + \frac{3 - 5f_1^1}{10}) \cdot 1 + (1 - f_1^1 - \frac{3 - 5f_1^1}{10})^2. \end{aligned}$$

The partial derivative of C with respect to f_1^1 is

$$\frac{\partial C}{\partial f_1^1} = \frac{5f_1^1 - 2}{10}$$

Setting the result to zero then solving for f_1^1, we have $f_1^1 = 2/5$, which implies that $f_1^2 = 3/10$. The cost of player 1 and player 2 are

$$C^1 = 0 \cdot 1 + (\frac{2}{5} - 0)(1 - 0 - \frac{3}{10}) = 0.28,$$

and

$$C^2 = \frac{3}{10} \cdot 1 + (\frac{3}{5} - \frac{3}{10})(1 - 0 - \frac{3}{10}) = 0.51.$$

And the social cost if given by

$$C = (0 + \frac{3}{10}) \cdot 1 + (1 - 0 - \frac{3}{10})^2 = 0.79.$$

2.5.9 A Malicious Atomic Leader and a UE Follower

In this final subsection, we still consider the malicious leader, instead, with a UE flow as a follower. We know that for every strategies of player 1, player 2 will always route all of its flow on link 2. Therefore for player 1 to maximise the social cost, it similarly routes all of its flow on link 2. Clearly the cost of player 1 is 2/5, and the cost of player 2 is 3/5. And the social cost if 1.

2.5.10 Comments

In this subsection, we give some comments based on the results shown in Table 2.1.

1. The ratio of the social cost of the user equilibrium to that of the system equilibrium is 4/3. This ratio is known as the price of anarchy. Roughgarden and Tardos [RT02] showed that the ratio of 4/3 is the tight bound for the price of anarchy for a general network with linear latency functions.

2. The social costs of the SSL flow both when player 1 is a leader and when player 2 is a leader, are higher than the social cost of the Nash flow. In the SSL flow, the leaders over-use link 2 forcing the followers to move some of their flows to link 1. As a consequence, it turns out that the leaders' costs are lower, but the followers' costs increase with greater amount, thus increasing the social costs. This fact was previously noticed by earlier works, for example Gibson mentioned it in [Gib92] for a Cournot competition. For a congestion network game, we initially found this

observation in [CP91] by Catoni and Pallottino. They studied a network of three parallel links where two of them are private links and one is a shared link. Note that the network we studied here is somewhat more restrict than that in [CP91] in that there are no private links.

3. The ratio of the social cost of when player 1 is a leader to that of the Nash equilibrium, and the social cost of player 2 is a leader and that of the Nash equilibrium are 1.015 and 1.044 respectively. These ratios demonstrate the price of selfish Stackelberg leadership studied in this thesis.

4. The fraction of the total flow a leader has can effect how much the leader benefits. As it shows in the comparison between when player 1 is a leader and when player 2 is a leader. Player 2, who has the bigger fraction of the total flow, reduces more of its per-unit cost than player 1 does. That is, when player 1 is a leader, it reduces $(0.009)/(2/5) \approx 0.0225$ while player 2 reduces $(0.014)/(3/5) \approx 0.0233$ in the set-up 4.2.

5. Comparing the cost of the leader when a follower is a UE flow to that of when a follower is an atomic player shows that having an atomic follower can be better off for the selfish leader. We believe this is in fact true in general, in a sense that an atomic follower reacts more responsively to a leader's strategy than a UE follower as shown in a few examples in which a UE flow completely disregards a leader's strategy.

6. For a benevolent leader, with regard to the leader's objective, an atomic follower is preferred. When a follower is a UE flow, the leader's strategy is insignificant to the follower. As a result, the the leader cannot ensure that the combined file is optimal. This is previously noticed by Roughgarden [Rou04].

7. The idea of a benevolent leader has been studied by many recent works as a tool to mitigate selfish users, for example Roughgarden [Rou04] who studied a network in which a follower is a UE flow, and Korilis et al. [KLO97a] and Orda et al. [ORS93] who studied a network with atomic followers.

2.5.11 Conclusion

A few conclusions can be drawn from the study in this subsection. First of all, we have reaffirmed that there are price of SSL (Comment 3). Moreover, we have observed that, in this particular example, when a leader has more flow in the system, the price of SSL gets higher. Clearly, this observation is not always true, as when the leader flow get much higher than the follower flow, the price of SSL will get closer to 1.

Another interesting observation from this study when comparing benefits that leaders obtain between infinitely many followers and a single follower, is that, a leader will prefer atomic followers because they react more responsively to leader's strategies. This implies that leaders can enforce strategies easier than infinitely many followers which often completely neglect leader's strategies.

Set-up Description	f_1^1	f_1^2	f_1	C^1	C^2	Social Cost
1. System equilibrium	-	-	1/2	-	-	0.75
2. User equilibrium	-	-	0	-	-	1
3. Two atomic players in NE	1/15	4/15	1/3	0.289	0.489	0.778
4.1 Two atomic players in SSL (player 1 as a leader)	0	3/10	3/10	0.28	0.51	0.79
4.2 Two atomic players in SSL (player 2 as a leader)	3/20	1/10	1/4	0.337	0.475	0.812
5. Atomic leader and UE follower	1/5	0	1/5	0.36	0.48	0.84
6.1. Benevolent atomic leader and atomic follower (player 1 as a leader)	2/5	1/10	1/2	0.4	0.35	0.75
6.2 Benevolent atomic leader and atomic follower (player 2 as a leader)	0	1/2	1/2	0.2	0.55	0.75
7. Benevolent atomic leader and UE follower	2/5	0	2/5	0.4	0.36	0.76
8. Malicious atomic leader and atomic follower	0	3/10	3/10	0.28	0.51	0.79
9. Malicious atomic leader and UE follower	0	0	0	2/5	3/5	1

Table 2.1: Routing strategies and their corresponding costs in the Pigou's example

Chapter 3

The Price of Selfish Stackelberg Leadership: Lower Bound from Examples

In Chapter 2, we have demonstrated that in a network of two parallel links and two selfish players each with a splittable flow, there might be an additional cost arising when allowing one of the players to have a Stackelberg leadership, in comparison to the social cost of Nash equilibrium. In this chapter, we show furthermore how much the additional cost can be in that simple network setting. In other words, what is the maximal price of SSL? We look at the problem of optimising the price of SSL of the following network configurations: a two-parallel-link network with latency functions that are linear, quadratic, cubic and quartic, and a three-parallel-link network in which two of the links are private links and one is a shared link, and each link has a linear latency function. For each configuration, we show an example that demonstrates the maximal price of SSL we obtained. These examples offer lower bounds for the price of SSL. We show in the next chapter an upper bound to match the lower bound of the network with two parallel links, each of which has a linear latency function.

3.1 Overview

Computing the price of SSL from examples shown in the previous chapter, we have noticed that with different coefficients of latency functions and a different fraction of player flows, prices of SSL are different. For example, in Example 2.4.7 where the latency of link 1 is $\ell_1(f_1) = 2f_1$ and the latency of link 2 is $\ell_2(f_2) = 1$, and each player has $1/2$ unit of flow, we obtained the price of SSL of 1.019. However, in the Pigou's Example where $\ell_1(f_1) = f_1$, $\ell_2(f_2) = 1$, the leader flow is $3/5$ and the follower flow is

2/5, we obtained the different price of SSL of 1.044 (comment 3 in Subsection 2.5.10). So it raises the following questions that lead us to the problem studied in this chapter:

- How to choose coefficients in linear latency functions and a fraction of player flows that have maximised the price of SSL?
- How much worse can the price of SSL be for some networks that have more general latency functions? e.g. quadratic etc.

In this chapter, we look at the problem of optimising the price of SSL with respect to the coefficients and the fraction of player flows of simple networks with different forms of latency functions. Our focus is on simple networks of various network configurations. We consider both symmetric and asymmetric networks. A symmetric network is a network in which every players have access to all the links, while an asymmetric network is a network in which each player does not have an access to all the links. We consider the simplest network of its kind—that is we study a network of two parallel links for a symmetric network, and a network with one shared link and two private links for an asymmetric network. We assume throughout this chapter that there are two selfish players; player 1 and player 2, each with a splittable flow to be routed through the networks. Player 1 (the leader in the SSL setting) controls a fixed flow of one unit while player 2 controls $f^2 > 0$ units of flow (as can be assumed by re-scaling).

We study networks with four different forms of latency function. For the symmetric network, we consider the networks with latency functions that are linear, quadratic, cubic and quartic. For the asymmetric network, we consider the network with linear latency functions. Given coefficients of latency functions and player 2's flow of each network configuration, we optimise the price of SSL. This can be described by a simple example as follows. Suppose we are given a network of two nodes connecting by two parallel links with the latencies $\ell_1(f_1) = a_1 \cdot (f_1)$ and $\ell_2(f_2) = b_2$ for some $a_1, b_2 > 0$. In the network, there are two players in which player 1's flow is fixed at one and player 2's flow is $f^2 > 0$. The aim is to try to find a set of constants a_1, b_2 and f^2 such that the price of SSL is maximised.

To solve the optimisation problem, we use an adapted Hill climbing technique (see for example [RN03] for more details). Hill climbing technique is appealing for several reasons. The most important one is that it is relatively simple to implement. Although there are other more advanced algorithms that may be more efficient, in our situations hill climbing works reasonably well. We discuss the main algorithm which uses the adapted Hill climbing in Section 3.2.1.

Aside from the optimisation technique, another important problem to consider is the computation of the price of SSL for a network of two players. That is, given a specific

network model with a set of coefficients and a fraction of player flows, we determine the price of SSL. By Definition 2.4.3, the computation of the price of SSL is essentially the problem of computing the social costs of a Nash flow and that of a SSL flow. We describe how to obtain player strategies at Nash equilibrium and SSL equilibrium in Section 3.3. Algorithm for finding the price of SSL is discussed in Subsection 3.2.2.

The maximal price of SSL we obtained for each network configuration can be summarised as follows (Table 3.1 at the end of the chapter provides the summation in a table form). In Section 3.3.1, we consider a symmetric network of two nodes, two parallel links shown in Figure 3.1. We study the network with affine linear latency functions of which the highest price of SSL we obtained, which is shown in Example 3.3.1, is approximately 1.071. We study the network where the latency of the one of the link is quadratic and that of the other link is constant. The maximal price of SSL we obtained is a multiplication factor 1.14 which is shown in Example 3.3.2. For the network where one of the link has a cubic latency function and the other link has a constant latency function, we obtained the price of SSL of approximately 1.177. For the network in which one of the link has a quartic latency function and the other one has a constant latency function, we obtained the price of SSL of 1.323. In Subsection 3.3.2, we consider an asymmetric network of two nodes three parallel links where each link has a linear latency function shown in Figure 3.4. The network has two private links each of which is used exclusively by the assigned player, and one shared link. We obtain the price of SSL of approximately 1.074 which is shown in Example 3.3.5.

Remark 3.1.1. Since the networks with quadratic, cubic and quartic latency functions we consider have some restrictions, the maximal price of SSL obtained for them are not necessary the maximal for the network with general latency functions in each setting. However, we have noticed from some simple examples and from the network with linear latency functions that in a network of two parallel links, the price of SSL tends to be higher if one of the link has a constant latency function which is the reason why in our set-ups for the networks with quadratic, cubic and quartic latency functions, we have one of the link be a constant latency function link.

Remark 3.1.2. The price of SSL of 1.071 obtained in Example 3.3.1 corresponds to an upper bound of 1.322 given in Section 4.4.

3.2 Algorithm Detail

Let G be a parallel-link network in which each link has a general latency function form of $\ell_j(f_j) = a_{j,d}(f_j)^d + a_{j,d-1}(f_j)^{d-1} + \ldots + a_{j,1}(f_j) + a_{j,0}$ where d is nonnegative integer

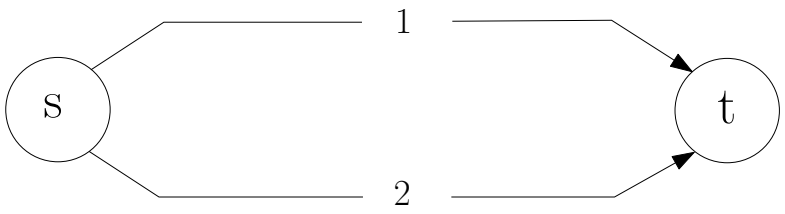


Figure 3.1: A simple symmetric network of two nodes, two directed links.

and $a_{j,0}, \ldots, a_{j,d}$ are constant coefficients. Let σ denote a vector of constants $(a_{j,d}, f^2)$ for all d of all j, and f^2 corresponds to the player 2's flow. Recall that by rescaling we can assume player 1's flow fixed at 1. The basic idea of the algorithm is to perform a local search for σ until we find the maximal price of SSL or a higher price of SSL can be found from our set-up. The program are separated into two main parts. One is to control a local search, and the other is to compute the price of SSL. The former is managed in the main algorithm shown in Figure 3.2, and the later is achieved in the POS algorithms shown in Figure 3.3.

3.2.1 The Main Algorithm

The objective of the main algorithm is to alter parameters, i.e. coefficients of latency functions and player 2's flow (recall that player 1's flow is fixed at 1), that initially generated at random, such that we move toward a set of parameters that produces a higher price of SSL.

Initially, we randomly generate σ (thus probably a bad one). We compute the price of SSL of every neighbours of σ. A neighbour of σ is σ with one of the constants changed by the value *step* given from the input. If k is the number of constants in σ, we compute the price of SSL of 2^k neighbours of σ. The computation of the price of SSL is handle with the function POS(). Subsequently, we move to the neighbour with the highest price of SSL if it is greater than that of σ. By repeating this process, we are essentially improving the solution a little bit each time.

Unfortunately, similar to Hill climbing, there are a few disadvantages on the main algorithm that we have to overcome. One of the commonly known disadvantage is that sometimes we reach a point where no better neighbour can be found even though it is not an optimal point. This is known in the field of optimisation algorithm as a plateau which occurs when we reach a "flat" part of a search space, i.e. a part where the quality of all the solutions are all very close together. To get out of this flat point we consider different neighbours that can be further away from a current σ and neighbours that have more changes to its constants. That is, instead of considering neighbours that

Input: *step*, *loop*.

Repeatedly, do the following for the number *loop* times:

1. Randomly generate $\sigma = (a_{j,d}, f^2)$.
2. $\sigma^* = 0$.
3. Let k be the number of members in σ.
4. Repeatedly do the following:
   (a) A neighbour of σ is σ that has one of the constants added or subtracted with *step*.
   (b) Let N be an array of size 2^k for all neighbours of σ.
   (c) Let S be an array of size 2^k.
   (d) From $i = 0$ to $2^k - 1$
   $S[i] = \text{POS}(N[i])$.
   (e) Let $\sigma' = \max(S)$.
   (f) If $\sigma' > \sigma$, $\sigma = \sigma'$.
   (g) Else, repeat the following steps for *loop* times or until σ', defined below, is found:
   - Let σ' be σ which has all constants, each is added with a random value in the range of $[-2step, 2step]$.
   - If $\text{POS}(\sigma') > \text{POS}(\sigma)$, $\sigma = \sigma'$ then return to step 4a.
5. If $\sigma' > \sigma^*$, $\sigma^* = \sigma'$.

Return σ^*.

Figure 3.2: A main algorithm for optimising the price of SSL.

have only one constant changed with fixed value *step*, we consider neighbours that each has every constants change by a random value in the range $[-2step, 2step]$. This process is repeated for the number of *loop*. If still no neighbour with a higher price of SSL is found, we restart with a new initial σ.

step indicates how much difference the neighbours of σ we consider. It is also used in the process of trying to get out of a flat point. Hence, *step* is an important parameter to try to avoid the plateau problem. If the *step* is set too small, we are more likely to experience the plateau problem and the algorithm can take a very long time to run. If the *step* is too big, we could miss out on paths that might lead to an optimal solution. For each network configuration, we test three different values of *step*—0.001, 0.01 and 0.1.

Another disadvantage to the main algorithm is that there is no guarantee that the

solution we obtain is the global optimum. To try to overcome this problem, we repeat the algorithm as many times as possible. By repeating the algorithm, we increase the chance of reaching an optimal point. The input variable *loop* is used to indicate how many times we run the algorithm. We use $loop = 10,000$ for symmetric networks with linear and quadratic latency functions, and an asymmetric network with linear latency functions. We use $loop = 1000$ for the network with cubic and quartic latency functions.

The value of the initial random σ is also very important. Intuitively, we want to start with values σ as close to an optimal solution as possible. The constants $a_{j,d}$ and f^2 in σ are initialised randomly from a reasonable range. For example, for $a_{j,d}$ of a link that has only a fixed cost, $a_{j,d}$ should not be initialised with such a high value that both players only route their flow on the other link at both Nash equilibrium and SSL equilibrium.

3.2.2 Algorithm for Finding the Price of SSL for Two-player Games

Recall that, for a network of two links, the strategy of player 1 $\mathbf{f}^1 = (f_1^1, f_1^2)$, and the strategy of player 2 $\mathbf{f}^2 = (f_1^2, f_2^2)$, can be decreased to f_1^1 and f_1^2 respectively since f_2^1 and f_2^2 can be obtained with $f^1 - f_1^1$ and $f^2 - f_1^2$.

1. Computing Player strategies at Nash equilibrium.

   We formulate the expressions of player strategies at Nash equilibrium in terms of the constants of σ. As mentioned in the previous chapter, finding the Nash equilibrium of two-player games is equivalent to finding the optimal strategies for each player while holding the other player's strategy fixed. By solving the problems, $\partial C^1/\partial f_1^1 = 0$ and $\partial C^2/\partial f_1^2 = 0$ we obtained the optimal strategies. Let us denote for the rest of this chapter the optimal strategies for player 1 and player 2 that are computed using the derivatives with $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ respectively. This means that the flow $\left((f_1^1)^{\mathrm{NE}}, (f_1^2)^{\mathrm{NE}}\right)$ is at Nash equilibrium. In addition, by solving $\partial C^1/\partial f_1^1 = 0$ and $\partial C^2/\partial f_1^2 = 0$ individually, we obtain the optimal strategies for each player with respect to the other player's strategy. Let us denote for the rest of this chapter the optimal strategy of player 1 with respect to any given f_1^2 with $(\overline{f_1^1})$. Similarly, let $(\overline{f_1^2})$ denote the optimal strategy of player 2 with respect to any f_1^1.

2. Computing Player strategies at SSL equilibrium.

   We wish to find the strategies $(f_1^1)^{\mathrm{SSL}}$ and $(f_1^2)^{\mathrm{SSL}}$ such that the cost of player 1 is minimal and $(f_1^2)^{\mathrm{SSL}}$ is the strategy for player 2 that optimises its cost with respect to $(f_1^1)^{\mathrm{SSL}}$ (see Section 2.4). Since $(f_1^2)^{\mathrm{SSL}}$ depends on $(f_1^1)^{\mathrm{SSL}}$, the

Given $\sigma = (a_{j,d}, f^2)$,

1. Compute $(f_1^1)^{\text{NE}}$ and $(f_1^2)^{\text{NE}}$ using the expressions shown later in this chapter.
   (a) If $0 \leq (f_1^1)^{\text{NE}} \leq 1$ and $0 \leq (f_1^2)^{\text{NE}} \leq f^2$,
      i. Let $C^{\text{NE}} = C\left((f_1^1)^{\text{NE}}, (f_2^2)^{\text{NE}}\right)$
   (b) Else if $\left((f_1^1)^{\text{NE}} < 0 \text{ or } (f_1^1)^{\text{NE}} > 1\right)$ and $\left((f_1^2)^{\text{NE}} < 0 \text{ or } (f_1^2)^{\text{NE}} > f^2\right)$,
      i. Let $C^{\text{NE}} = C^{\text{SSL}}$.
   (c) Else,
      - If $(f_1^1)^{\text{NE}} > 1$, $(f_1^1)' = 1$ or if $(f_1^1)^{\text{NE}} < 0$, $(f_1^1)' = 0$.
      - Let $(\overline{f_1^1}) = (f_1^1)^{\text{NE}}$.
      - Let $(f_1^1)^* = (\overline{f_1^1})$.
      - If $(f_1^2)^{\text{NE}} > f^2$, $(f_1^2)' = f^2$ or if $(f_1^2)^{\text{NE}} < 0$, $(f_1^2)' = 0$.
      - Let $(\overline{f_1^2}) = (f_1^2)^{\text{NE}}$.
      - Let $(f_1^2)^* = (\overline{f_1^2})$.
      - Repeatedly do the following until $(f_1^1)' = (f_1^1)^*$ and $(f_1^2)' = (f_1^2)^*$,
        i. If $(f_1^1)' \neq (f_1^1)^*$,
           A. Compute $(\overline{f_1^2})$ with respect to $(f_1^1)'$.
           B. If $(\overline{f_1^2}) > f^2$, $(f_1^2)' = f^2$, or if $(\overline{f_1^2}) < 0$, $(f_1^2)' = 0$.
           C. Let $(f_1^1)^* = (f_1^1)'$ and $(f_1^2)^* = (\overline{f_1^2})$.
        ii. Else if $(f_1^2)' \neq (f_1^2)^*$,
           A. Compute $(\overline{f_1^1})$ with respect to $(f_1^2)'$.
           B. If $(\overline{f_1^1}) > 1$, $(f_1^1)' = 1$, or if $(\overline{f_1^1}) < 0$, $(f_1^1)' = 0$.
           C. Let $(f_1^1)^* = (\overline{f_1^1})$ and $(f_1^2)^* = (f_1^2)'$.
      - Let $C^{NE} = C((f_1^1)', (f_1^2)')$.
2. Let $(C^1)^*$ be a high number, and $(f_1^1)^{\text{SSL}} = 0$ and $(f_1^2)^{\text{SSL}} = 0$.
3. For $f_1^1 = 0$ to 1, $f_1^1 = f_1^1 + 0.01$:
   (a) Compute $(\overline{f_1^2})$ with respect to f_1^1.
   (b) If $(\overline{f_1^2}) > f^2$, $(\overline{f_1^2}) = f^2$, or if $(\overline{f_1^2}) < 0$, $(\overline{f_1^2}) = 0$.
   (c) Let $(C^1)' = C^1\left(f_1^1, (\overline{f_1^2})\right)$.
   (d) If $(C^1)' < (C^1)^*$, $(C^1)^* = (C^1)'$, $(f_1^1)^{\text{SSL}} = f_1^1$ and $(f_1^2)^{\text{SSL}} = (\overline{f_1^2})$.
4. Let $C^{\text{SSL}} = C\left((f_1^1)^{\text{SSL}}, (f_1^2)^{\text{SSL}}\right)$.

Return $C^{\text{SSL}}/C^{\text{NE}}$.

Figure 3.3: The POS algorithm computes the price of SSL for a given σ.

problem becomes one variable problem. It is unnecessary (and can be quite complicated) to formulate the explicit expressions for SSL equilibrium similar to Nash equilibrium. The only expression required is the optimal strategy of player 2 with respect to any given f_1^1, i.e. $(\overline{f_1^2})$ which we will already obtain in the process of computing Nash equilibrium.

POS algorithm

POS algorithm depicted in Figure 3.2.2 compute a price of SSL for any given set of parameters. The computation is done via two steps, first is to compute the social cost of SSL flow and then the social cost of NE flow. The detail description of POS() is as follows.

Firstly, POS() computes a social cost at Nash equilibrium. As mentioned above, if both $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ are feasible, it simply computes $C((f_1^1)^{\mathrm{NE}}, (f_1^2)^{\mathrm{NE}})$ for the social cost at Nash equilibrium. If both $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ are infeasible, we return the price of SSL of 1.

If one of the strategies is infeasible, we readjust the strategies as follows. For $i = \{1, 2\}$, if $f_1^i > f^i$ then set f_1^i to f^i, and if $f_1^i < 0$ then set f_1^i to 0. Step 1c reassures that we obtain the feasible strategies $(f_1^1)'$ and $(f_1^2)'$, such that $C^1((f_1^1)', (f_1^2)') \leq C^1(f_1^1, (f_1^2)')$ and $C^2((f_1^1)', (f_1^2)') \leq C^1((f_1^1)', f_1^2)$ which, by Definition 2.3.3, guarantees that $((f_1^1)', (f_1^2)')$ is at Nash equilibrium.

Secondly, POS() computes the social cost at SSL equilibrium. We use a brute force search for feasible strategies $(f_1^1)^{\mathrm{SSL}}$ and $(f_1^2)^{\mathrm{SSL}}$, such that $C^1((f_1^1)^{\mathrm{SSL}}, (f_1^2)^{\mathrm{SSL}})$ where $(f_1^2)^{\mathrm{SSL}}$ is computed from the expression $(\overline{f_1^2})$, which implies that $(f_1^2)^{\mathrm{SSL}}$ is the strategy of player 2 which optimises its cost with respect to $(f_1^1)^{\mathrm{SSL}}$.

3.3 The Computation of the Price of SSL

As mentioned earlier, we find four expressions; $(f_1^1)^{\mathrm{NE}}$, $(f_1^2)^{\mathrm{NE}}$, $(\overline{f_1^1})$ and $(\overline{f_1^2})$, for each network configurations.

3.3.1 Symmetric Networks

In this section, we focus on the symmetric networks of one single source node and one single destination node connected by two parallel links, shown in Figure 3.1.

Affine Linear Latency Functions

We consider the network with a non-decreasing linear latency function of the form $\ell_j(f_j) = a_j f_j + b_j$ where $a_j, b_j \geq 0$. Since the price of SSL is 1 in a network, where every

link has homogeneous latency function (as we will show in Section 4.5), our focus here is on a scenario where at least one latency function is a nonhomogeneous linear, i.e. at least one of the links has $b_j > 0$.

Firstly, we obtain player strategies $(f_1^1)^{\text{NE}}$ and $(f_1^2)^{\text{NE}}$ by solving the equations $\partial C^1/\partial f_1^1 = 0$ and $\partial C^2/\partial f_1^2 = 0$. We consider the cost experienced by player 1:

$$C^1(f_1) = \sum_{j \in E} f_j^1(a_j(f_j^1 + f_j^2) + b_j)$$

which (for two links) can be explicitly written as

$$f_1^1\left(a_1(f_1^1 + f_1^2) + b_1\right) + (1 - f_1^1)\left(a_2(1 - f_1^1 + f^2 - f_1^2) + b_2\right)$$

where we substitute f_2^1 and f_2^2 with $1 - f_1^1$ and $f^2 - f_1^2$ respectively. Differentiating C^1 with respect to f_1^1, we obtain

$$\frac{\partial C^1}{\partial f_1^1} = 2(a_1 + a_2)f_1^1 + (a_1 + a_2)f_1^2 + b_1 - b_2 - (2 + f^2)a_2.$$

Equating the derivative to zero then solving for $(\overline{f_1^1})$ we have

$$(\overline{f_1^1}) = \frac{-b_1 + b_2 - a_1 \cdot f_1^2 + a_2(2 - f_1^2 + f^2)}{2(a_1 + a_2)}. \tag{3.1}$$

Next we consider the cost of player 2:

$$C^2(f_2) = \sum_{j \in E} f_j^2(a_j(f_j^1 + f_j^2) + b_j)$$

which can be explicitly written as

$$f_1^2\left(a_1(f_1^1 + f_1^2) + b_1\right) + (f^2 - f_1^2)\left(a_2(1 - f_1^1 + f^2 - f_1^2) + b_2\right)$$

where similarly we substitute f_2^1 and f_2^2 with $1 - f_1^1$ and $f^2 - f_1^2$ respectively. Differentiating C^2, we get

$$\frac{\partial C^2}{\partial f_1^2} = b_1 - b_2 + a_1 f_1^1 + 2a_1 \cdot f_1^2 + a_2(-1 + f_1^1 + 2f_1^2 1 - 2f^2)$$

Setting the derivative to zero then solving for $(\overline{f_1^2})$ we have

$$(\overline{f_1^2}) = \frac{a_2 - b_1 + b_2 - a_1 \cdot f_1^1 - a_2 f_1^1 + 2a_2 f^2}{2(a_1 + a_2)} \tag{3.2}$$

Solving the pair of (3.1) and (3.2), we obtain the strategies for a flow at Nash equilibrium as

$$(f_1^1)^{\mathrm{NE}} = \frac{3a_2 - b_1 + b_2}{3(a_1 + a_2)} \quad \text{and} \quad (f_1^2)^{\mathrm{NE}} = \frac{3a_2 \cdot f^2 - b_1 + b_2}{3(a_1 + a_2)}.$$

The following example shows a lower bound on the price of SSL in a network with linear latency functions.

Example 3.3.1. Consider the network of two nodes and two parallel links shown in Figure 3.1. Let us suppose that the latency of the upper link is $\ell_1(f_1) = 2.46f_1 + 0.01$ and the latency of the lower link is $\ell_2(f_2) = 4.92$. Player 1 controls a flow of one unit while player 2 controls a flow of 0.59 unit. The flow at Nash equilibrium is $(0.70, 0.59))$ with the social cost of approximately 5.58. The flow at SSL equilibrium is $(1.00, 0.50))$ with the social cost of approximately 6.00. Hence the price of SSL in this example is ≈ 1.075.

Quadratic Latency Functions

We consider the network model for the following configuration. From Figure 3.1, the latency of the upper link is $\ell_1(f_1) = (f_1)^2 + a(f_1)$ where $a \geq 0$ and the latency of the lower link is $\ell_2(f_2) = c$ where $c > 0$. This is generally achieved by rescaling. The assumption that some link has only a fixed cost means that the result may not be optimal for general quadratic functions.

The cost experienced by player 1 in this case is

$$C^1(f_1) = f_1^1((f_1^1 + f_1^2)^2 + a(f_1^1 + f_1^2)) + (1 - f_1^1)c.$$

Taking the derivative of C^1 with respect to f_1^1 we get

$$-c + a(f_1^1 + f_1^2) + (f_1^1 + f_1^2)^2 + f_1^1(a + 2(f_1^1 + f_1^2)). \tag{3.3}$$

Setting the result to zero then solving for f_1^1, we get

$$f_1^1 = \frac{1}{3}\left(-a - 2f_1^2 \pm \sqrt{a^2 + 3c + af_1^2 + (f_1^2)^2}\right).$$

Since f_1^1 must be positive, thus the only feasible $(\overline{f_1^1})$ is

$$(\overline{f_1^1}) = \frac{1}{3}\left(-a - 2f_1^2 + \sqrt{a^2 + 3c + af_1^2 + (f_1^2)^2}\right).$$

Next, we consider the cost of player 2,

$$C^2(f_2) = f_1^2((f_1^1 + f_1^2)^2 + a(f_1^1 + f_1^2)) + (f^2 - f_1^2)c.$$

Taking the derivative of C^2 with respect to f_1^2, we get

$$-c + a(f_1^1 + f_1^2) + (f_1^1 + f_1^2)^2 + f_1^2(a + 2(f_1^1 + f_1^2)). \tag{3.4}$$

Setting the result to zero, then solving for f_1^2, we get

$$(\overline{f_1^2}) = \frac{1}{3}\left(-a - 2f_1^1 \pm \sqrt{a^2 + 3c + af_1^1 + (f_1^1)^2}\right).$$

Similarly, since f_1^2 must be positive, the only valid result from the above equation is

$$(\overline{f_1^2}) = \frac{1}{3}\left(-a - 2f_1^1 + \sqrt{a^2 + 3c + af_1^1 + (f_1^1)^2}\right).$$

Setting (3.3) and (3.4) to zero then solving for f_1^1 and f_1^2, we get

$$f_1^1 = f_1^2 = \frac{1}{16}\left(-3a \pm \sqrt{9a^2 + 32c}\right).$$

Since $a, c \geq 0$, the only feasible solution is

$$(f_1^1)^{\mathrm{NE}} = (f_1^2)^{\mathrm{NE}} = \frac{1}{16}\left(-3a + \sqrt{9a^2 + 32c}\right).$$

Example 3.3.2. Modifying the network from Example 3.3.1, let us suppose that the latency of the upper link is $\ell_1(f_1) = f_1^2 + 0.01f_1$ and the latency of the lower link is $\ell_2(f_2) = 1.38$. Player 1 controls a flow of one unit while player 2 controls a flow of 0.35 unit. The flow at Nash equilibrium is $\mathbf{f}^{\mathrm{NE}} = (0.45, 0.35))$ with the social cost of approximately 1.277. The flow at SSL equilibrium is $\mathbf{f}^{\mathrm{SSL}} = (0.65, 0.28))$ with the social cost of approximately 1.393. Hence the price of SSL in this example is ≈ 1.091.

Cubic Latency Functions

Let us modify the network Figure 3.1 by letting the upper link have a latency $\ell_1(f_1) = (f_1)^3 + a(f_1)^2$ for $a \geq 0$, and the lower link to have a latency $\ell_2(f_2) = c$ for $c > 0$.

Now we obtain the Nash flow by solving the problems $\partial C^1/\partial f_1^1 = 0$ and $\partial C^2/\partial f_1^2 = 0$, for f_1^1 and f_1^2. The cost of player 1 is

$$C^1 = f_1^1((f_1^1 + f_1^2)^3 + a(f_1^1 + f_1^2)) + (1 - f_1^1)c.$$

Differentiating and equating the result to zero, we get

$$-c + a(f_1^1 + f_1^2) + (f_1^1 + f_1^2)^3 + f_1^1\left(a + 3(f_1^1 + f_1^2)^2\right) = 0. \tag{3.5}$$

Solving (3.5), we obtain $(\overline{f_1^1})$ as

$$\begin{aligned}(\overline{f_1^1}) = &\frac{-24a + 9(f_1^2)^2}{62^{2/3}\left(432c + 216af_1^2 + 54(f_1^2)^3 + 2\sqrt{\left(24a - 9(f_1^2)^2\right)^3 + 729\left(8c + 4af_1^2 + (f_1^2)^3\right)^2}\right)^{1/3}} \\ &+ \frac{\left(432c + 216af_1^2 + 54(f_1^2)^3 + 2\sqrt{\left(24a - 9(f_1^2)^2\right)^3 + 729\left(8c + 4af_1^2 + (f_1^2)^3\right)^2}\right)^{1/3}}{122^{1/3}} \\ &- \frac{3f_1^2}{4}\end{aligned}$$

Next, we consider the cost experienced by player 2:

$$C^2 = f_1^2((f_1^1 + f_1^2)^3 + a(f_1^1 + f_1^2)) + (f^2 - f_1^2)c$$

Differentiating and equating the result to zero, we get

$$-c + a(f_1^1 + f_1^2) + (f_1^1 + f_1^2)^3 + f_1^2\left(a + 3(f_1^1 + f_1^2)^2\right) = 0 \tag{3.6}$$

Solving (3.6), we obtain

$$\begin{aligned}(\overline{f_1^2}) = &\frac{-24a + 9(f_1^1)^2}{62^{2/3}\left(432c + 216af_1^1 + 54(f_1^1)^3 + 2\sqrt{\left(24a - 9(f_1^1)^2\right)^3 + 729\left(8c + 4af_1^1 + (f_1^1)^3\right)^2}\right)^{1/3}} \\ &+ \frac{\left(432c + 216af_1^1 + 54(f_1^1)^3 + 2\sqrt{\left(24a - 9(f_1^1)^2\right)^3 + 729\left(8c + 4af_1^1 + (f_1^1)^3\right)^2}\right)^{1/3}}{122^{1/3}} \\ &- \frac{3f_1^1}{4}\end{aligned}$$

Solving (3.5) and (3.6), we obtain the expressions for $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ as

$$(f_1^1)^{\mathrm{NE}} = (f_1^2)^{\mathrm{NE}} = \frac{5^{2/3}a - \sqrt[3]{5}(\sqrt{5} + \sqrt{a^3 + 5c^2} - 5c)^{2/3}}{10\sqrt[3]{\sqrt{5}\sqrt{a^3 + 5c^2} - 5c}}.$$

Example 3.3.3. Consider the modification of Example 3.3.2 by replacing the latency function of the upper bound from the quadratic function to the cubic function $\ell_1(f_1) = (f_1)^3$ and the latency of the lower link to $\ell_2(f_2) = 4.95$. Again, player 1 has one unit of

flow while player 2 has 0.628 units. At Nash equilibrium, player 2 puts all of his flow on the upper link. Player 1 routes the same amount on the upper link. At SSL equilibrium, player 1 routes all of his flow on the upper link, pushing player 2 to put 0.39 unit of his flow on the lower link. As a result, the social cost of Nash and at SSL equilibrium are ≈ 4.323 and ≈ 4.913 respectively. Hence the price of SSL is at least 1.135.

Quartic Latency Functions

Our final result for a symmetric network is the network with a latency function of a degree of, at most, four. Let us suppose that, in the network shown in Figure 3.1, the upper link has a latency $\ell_1(f_1) = (f_1)^4 + a(f_1)^3$ where $a \geq 0$ and the lower link has a latency $\ell_2(f_2) = c$. First we obtain the expression f_1^1 and f_1^2 at the Nash equilibrium by solving the two equations $\partial C^1/\partial f_1^1 = 0$ and $\partial C^2/\partial f_1^2 = 0$. The cost experienced by player 1 is

$$C^1 = f_1^1((f_1^1 + f_1^2)^4 + a(f_1^1 + f_1^2) + (1 - f_1^1)c.$$

Differentiating C^1 and then equating the result to zero, we get

$$-c + (f_1^1 + f_1^2)^2 \left(5(f_1^1)^2 + 6f_1^1 f_1^2 + (f_1^2)^2 + a(4f_1^1 + f_1^2)\right) = 0. \tag{3.7}$$

Solving (3.7) for $(\overline{f_1^1})$, we obtain

$$\begin{aligned}(\overline{f_1^1}) = \frac{1}{5}(-a - 4f_1^2) - \frac{1}{10\sqrt{3}}(&\sqrt{(-90f_1^2(a + 2f_1^2) + 12(a + 4f_1^2)^2 - (53^{2/3}(20c} \\ &- 3a^2(f_1^2)^2))/g + 53^{1/3}g)) + \frac{1}{2}\sqrt{(-\frac{12}{5}f_1^2(a + 2f_1^2) + \frac{8}{25}(a + 4f_1^2)^2 + (20c} \\ &- 3a^2(f_1^2)^2)/(53^{1/3}g) - \frac{1}{53^{2/3}}g + (4\sqrt{3}(4a^3 + 3a^2 f_1^2 - 3a(f_1^2)^2 - 4(f_1^2)^3)) \\ &/(25\sqrt{(-90f_1^2(a + 2f_1^2) + 12(a + 4f_1^2)^2 - (53^{2/3}(20c - 3a^2(f_1^2)^2))/g + 53^{1/3}g)}))\end{aligned}$$

where

$$\begin{aligned}g = (&-72a^2c - 36acf_1^2 - 72c(f_1^2)^2 - 9a^3(f_1^2)^3 + 2\sqrt{3}(c(72a^2c(f_1^2)^2 + 108a^5(f_1^2)^3 \\ &+ 432ac(f_1^2)^3 + 27a^4(16c + 7(f_1^2)^4) + 16c(125c + 27(f_1^2)^4) \\ &+ 108a^3(4cf_1^2 + (f_1^2)^5)))^{1/2})^{1/3}.\end{aligned}$$

Next, we consider the cost of player 2

$$C^2 = f_1^2((f_1^1 + f_1^2)^4 + a(f_1^1 + f_1^2)) + (f^2 - f_1^2)c.$$

Taking the derivative of C^2 with respect to f_1^2 and set the result to zero, we obtain

$$-c + (f_1^1 + f_1^2)^2 \left((f_1^1)^2 + 6f_1^1 f_1^2 + 5(f_1^2)^2 + a(f_1^1 + 4f_1^2)\right) = 0. \tag{3.8}$$

Solving 3.8 for $(\overline{f_1^2})$, we get

$$\begin{aligned}(\overline{f_1^2}) = \frac{1}{5}(-a - 4f_1^1) - \frac{1}{10\sqrt{3}}(\sqrt{(-90f_1^1(a + 2f_1^1) + 12(a + 4f_1^1)^2 - (53^{2/3}(20c} \\ - 3a^2(f_1^1)^2))/g + 53^{1/3}g)) + \frac{1}{2}\sqrt{(-\frac{12}{5}f_1^1(a + 2f_1^1) + \frac{8}{25}(a + 4f_1^1)^2 + (20c} \\ - 3a^2(f_1^1)^2)/(53^{1/3}g) - \frac{1}{53^{2/3}}g + (4\sqrt{3}(4a^3 + 3a^2 f_1^1 - 3a(f_1^1)^2 - 4(f_1^1)^3)) \\ /(25\sqrt{(-90f_1^1(a + 2f_1^1) + 12(a + 4f_1^1)^2 - (53^{2/3}(20c - 3a^2(f_1^1)^2))/g + 53^{1/3}g)))\end{aligned}$$

where

$$\begin{aligned}g = (-72a^2c - 36acf_1^1 - 72c(f_1^1)^2 - 9a^3(f_1^1)^3 + 2\sqrt{3}(c(72a^2c(f_1^1)^2 + 108a^5(f_1^1)^3 \\ + 432ac(f_1^1)^3 + 27a^4(16c + 7(f_1^1)^4) + 16c(125c + 27(f_1^1)^4) \\ + 108a^3(4cf_1^1 + (f_1^1)^5)))^{1/2})^{1/3}.\end{aligned}$$

Then, solving (3.7) and (3.8) yields the expressions $(f_1^1)^{\mathrm{NE}}$ and $(f_1^2)^{\mathrm{NE}}$ of which both are equal to

$$\begin{aligned}&\frac{1}{48}(-5a - (25a^2 + (24(-16c + (-25a^2c + \sqrt{c^2(625a^4 + 4096c)})^{2/3}))/(-25a^2c \\ &+ \sqrt{c^2(625a^4 + 4096c)})^{1/3})^{1/2} + 24((25a^2)/288 + (2c)/(3(-25a^2c \\ &+ \sqrt{c^2(625a^4 + 4096c)})^{1/3}) - 1/24(-25a^2c + \sqrt{c^2(625a^4 + 4096c)})^(1/3) + (125a^3)/ \\ &(6912((25a^2)/576 + (-16c + (-25a^2c + \sqrt{c^2(625a^4 + 4096c)})^{2/3})/(24(-25a^2c \\ &+ \sqrt{c^2(625a^4 + 4096c)})^{1/3}))^{1/2}))^{1/2}).\end{aligned}$$

Example 3.3.4. Consider a network from Example 3.3.2 where we change the upper link's latency function and the lower link's latency to $\ell_1(f_1) = (f_1)^4 + 0.17f_1$ and $\ell_2(f_2) = 1.96$, respectively. Player 1 as usual controls a flow of one unit, while player 2 controls a flow of 0.33 unit. At Nash equilibrium, player 2 puts all of his flow on the upper link and player 1 puts 0.43 of its flow on the upper link and the rest of the flow on the lower link. At SSL equilibrium, player 1 put 0.71 of its flow on the upper link, pushing player 2 to route 0.09 unit of its flow on the lower link. As a result, the social cost at the Nash equilibrium and at the SSL equilibrium are approximately 1.427 and 1.657 respectively. The price of SSL is approximately 1.161.

3.3.2 Asymmetric Network

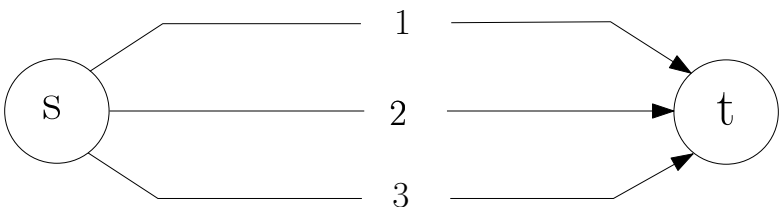


Figure 3.4: A simple three-link asymmetric network in which 1 and 3 are private links, and 2 is a shared link.

We now consider the asymmetric network in Figure 3.4 in which two nodes are connected with three parallel links. Each link has a linear latency function in the form $\ell_j(f_j) = a_j(f_j) + b_j$ for $j \in \{1, 2, 3\}$ where $a_j, b_j \geq 0$. Again player 1 is assumed to control one unit of flow and player 2 is assumed to control $f^2 > 0$ unit of flow. We wish to determine the player strategies $(f_1^1)^{\text{NE}}$, $(f_3^2)^{\text{NE}}$, $(\overline{f_1^1})$ and $(\overline{f_3^2})$. The cost experienced by player 1 is

$$C^1 = f_1^1(a_1(f_1^1) + b_1) + (1 - f_1^1)(a_2(1 - f_1^1 + f^2 - f_3^2) + b_2).$$

where we substitute f_2^1 and f_2^2 with $1 - f^1$ and $f^2 - f_3^2$ respectively. Differentiating C^1 with respect to f_1^1 and setting the result to zero, we get

$$b_1 - b_2 + 2a_1 f_1^1 + a_2(-2 + 2f_1^1 - f^2 + f_3^2) = 0. \tag{3.9}$$

Solving (3.9) for $(\overline{f_1^1})$, we get

$$(\overline{f_1^1}) = \frac{-b_1 + b_2 + a_2(2 + f^2 - f_3^2)}{2(a_1 + a_2)}.$$

The cost of player 2 is given by

$$C^2 = (f^2 - f_3^2)(a_2(1 - f_1^1 + f^2 - f_3^2) + b_2) + f_3^2(a_3(f_3^2) + b_3).$$

Differentiating C^2 with respect to f_1^2 and setting the result to zero, we get

$$-b_2 + b_3 + 2a_3 f_3^2 + a_2(-1 + f_1^1 - 2f^2 + 2f_3^2) = 0. \tag{3.10}$$

Solving (3.10) for $(\overline{f_3^2})$, we have

$$(\overline{f_3^2}) = \frac{a_2 + b_2 - b_3 - a_2 f_1^1 + 2a_2 f^2}{2(a_2 + a_3)}.$$

Solving (3.9) and (3.10) for $(f_1^1)^{\text{NE}}$ and $(f_3^2)^{\text{NE}}$, we obtain

$$(f_1^1)^{\text{NE}} = \frac{3a_2^2 + 2a_3(-b_1 + b_2) + a_2(-2b_1 + b_2 + b_3 + 2a_3(2 + f^2))}{4a_1(a_2 + a_3) + a_2(3a_2 + 4a_3)}$$

and

$$(f_1^2)^{\text{NE}} = \frac{2a_1(a_2 + b_2 - b_3 + 2a_2 f^2) + a_2(b_1 + b_2 - 2b_3 + 3a_2 f^2)}{4a_1(a_2 + a_3) + a_2(3a_2 + 4a_3)}.$$

Example 3.3.5. Consider a network in Figure 3.4 in which the top link, middle link and bottom link have a latency of $\ell_1(f_1) = 2.75$, $\ell_2(f_2) = f_2$ and $\ell_3(f_3) = 3.5$ respectively. Player 1 controls a one unit flow, while player 2 controls a 1.42 unit of flow. At Nash equilibrium, player 1 routes a 0.33 fraction of its flow on link 1, while player 2 routes all of its flow on link 2. The social cost at Nash equilibrium is approximately 5.27. At SSL equilibrium, player 1 routes all of its flow in link 2 and player 2 routes 1.25 units of flow on link 2. The social cost at the SSL equilibrium is approximately 5.66, which results in the price of SSL of approximately 1.074.

Remark 3.3.6. We show in the next chapter that there is no price of SSL if a parallel-link network only has homogeneous linear latency functions. However, for asymmetric networks, the leader can take advantage even in a network where the all latency functions are homogeneous linear, as we show in the next example.

Example 3.3.7. Consider a network in Figure 3.4 in which $\ell_1(f_1) = f_1$, $\ell_2(f_2) = 2f_2$ and $\ell_3(f_3) = f_3$. Player 1 and player 2 each have one unit of flow. We find a flow (f_1^1, f_3^2) at Nash equilibrium and SSL equilibrium Using the expressions formulated above, we have a Nash flow of (0.75,0.75) and SSL flow of (0.76, 0.71). The social cost of the Nash flow is 1.625 and the social cost of the SSL flow is 1.644. Hence we have the price of SSL of 1.012.

3.4 Discussion

In Table 3.1, we have summarised the maximal price of SSL we obtained for each of the network configurations studied. Our results show some lower bounds for the price of SSL. The multiplication factor 1.075 for the network with linear latency functions offers a lower bound corresponding with an upper bound of 1.322, as we will show in the next chapter.

Description	$\ell_1(f_1)$	$\ell_2(f_2)$	$\ell_3(f_3)$	Price of SSL (approximately)
Affine linear	$a_1 f_1 + b_1$	$a_2 f_2 + b_2$	-	1.075
Quadratic	$(f_1)^2 + a f_1$	c	-	1.091
Cubic	$(f_1)^3 + a(f_1)^2$	c	-	1.135
Quartic	$(f_1)^4 + a(f_1)^3$	c	-	1.161
Asymmetric with linear	$a_1 f_1 + b_1$	$a_2 f_2 + b_2$	$a_3 f_3 + b_3$	1.074

Table 3.1: The results of the price of SSL for networks with various latency function forms with some restrictions. All coefficients are assume to be non-negative.

So far, we have not found any examples of a parallel network for the case of two players, which would suggest that the price of SSL is unbounded. In contrast to the price of anarchy of a network of two players where one player is atomic and the other is nonatomic, Roughgarden showed that the price of anarchy can be arbitrarily large, even for a simple two parallel links model for general latency functions. In that particular example the price of SSL is not unbounded.

It can be seen from the results that, as we allow a more general class of latency functions, the price of SSL increases. This suggests that the price of SSL depends crucially on the "steepness" of the network cost functions. An interesting open problem on this subject would be how big can the price of SSL be, for unrestricted convex latency functions?

Chapter 4

Upper Bounding the Price of Selfish Stackelberg Leadership

So far, we have demonstrated in several examples in Chapters 2 and 3 that there can be an additional social cost if there exists a player with Stackelberg leadership when compared to the standard Nash setting, even in a simple two parallel links model with linear latency functions.

In this chapter, we investigate upper bounds on the price of selfish Stackelberg leadership (price of SSL), namely the worst possible ratio of the social cost that occurs when one of the players is a selfish leader to that of the Nash equilibrium. In particular, we focus on the model of two atomic players each with a splittable flow in a parallel-link network with linear latency functions. We consider both when latency functions are general linear and when latency functions are restricted to homogeneous linear.

4.1 Overview

The main objective of this chapter is to prove upper bounds on the price of SSL. That is, we prove the bounds on the worst ratio between a social cost of a SSL flow and that of a Nash flow. We concentrate on a simple model of two atomic splittable players in a network with n parallel links each of which has a latency function of the form $\ell_j(f_j) = a_j f_j + b_j$ where $a_j, b_j \geq 0$. We assume that both players are selfish and want to minimise their individual costs. Furthermore, we assume that the total amount of flow in the network is one. We prove that there are constant upper bounds that are independent of the number of links in the network. Note that all of the notations and the definitions used in this chapter are previously described in Chapter 2.

Our main approach to obtain the upper bounds is to bound the social cost of a SSL flow with a combination of upper bounds on player individual costs, while the social

cost of a Nash flow is lower bounded by an optimal cost. To prove the upper bounds on player individual costs, we have to know what players' strategies are in the SSL flow. Finding optimal strategies for players are difficult, especially for a leader, we analyse two simple strategies, one for each player, that guarantee some upper bounds. That is, if the leader plays with the proposed strategy then regardless of what the follower will do, the leader's cost is at most the upper bound, and vice versa. We describes the strategies in Section 4.2. Theses strategies are not necessary optimal for corresponding players, but the upper bounds for player costs obtained when they are used can guarantee the upper bounds for the costs when optimal strategies are used.

In Section 4.2, we prove an upper bound on the price of SSL of 2. In Section 4.3, with a slightly more careful analysis on the upper bounds on player costs, we show an upper bound on the price of SSL of 4/3. In Section 4.4, we present our main result showing an upper bound of slightly less than 4/3, or 1.322 to be precise, by dint of a more complicated analysis and also by considering the *aloof strategy* of Roughgarden [Rou04]. This upper bound together with the lower bound of 1.075 shown in Example 3.3.1 in a network of two parallel links, demonstrates that the price of SSL is in a narrow range of [1.075, 1.322].

In Section 4.5, we consider the model with homogeneous linear functions, namely linear functions where every b_j is zero (or all b_j are identical by normalising). We prove that a SSL flow is unique and identical to a Nash flow, consequently, the price of SSL is 1 in such a model. Finally, we conclude the chapter with a conclusion and some open problems in Section 4.6.

4.2 Two Strategies.

In this section, we consider two strategies, one for each player in a Stackelberg game, that provide a suboptimal performance guarantee. The first strategy is for a leader (or as we normally refer to as player 1). It was used by Korilis et al. [KLO97a] for a benevolent leader as a routing strategy in the same model where there is a single follower. They proved that a leader using this strategy can always enforce an optimal flow.

Strategy 1:

Strategy 1, depicted in Figure 4.1, is for player 1. The general idea is that player 1 acting like a benevolent leader, tries to enforce an optimal flow ($\mathbf{f}^{\mathrm{SE}}$). Provided that player 1 can achieve this, it follows immediately that player 1's cost is, at most, the optimal cost, which is the total cost for both players. We describe the strategy as follows.

1. Compute an optimal flow $\mathbf{f}^{\mathrm{SE}}$.
2. Let $\widehat{n} \leq n$ be the maximal with $f^{\mathrm{SE}}_{\widehat{n}} > 0$.
3. Compute an optimal flow $(\mathbf{f}^{\mathrm{SE}})^2$ where $(f^{\mathrm{SE}})^2 = f^2$.
4. Let $j^* \leq \widehat{n}$ be the maximal with $(f^{\mathrm{SE}})^2_{j^*} > 0$.
5. For $j = j^* + 1$ to $\widehat{n}$, set $f^1_j = f^{\mathrm{SE}}_j$.
6. For $j = 1$ to j^*, set $f^1_j = \left(f^1 - \sum_{j=j^*+1}^{\widehat{n}} f^1_j\right) / \left(a_j \sum_{j=1}^{j^*} 1/a_j\right)$.

Figure 4.1: Strategy 1 for player 1

The first step of the strategy is for player 1 to compute $\mathbf{f}^{\mathrm{SE}}$. The second step is to divide the links used in $\mathbf{f}^{\mathrm{SE}}$ into two subgroups; one is the group of links that will be used by both player 1 and 2 in the SSL flow, and the other is the group of links that will be used only by player 1. The list of links that will be used by player 2 can be obtained by computing the optimal flow relative to f^2, in other words, computing player 2's strategy as if player 1 is not present in the system. Recall that links are indexed in increasing order of b_j and an optimal flow preserves the order of links (Corollary 2.2.6). Let j^* be the maximal indexed link that player 2 will use and $\widehat{n}$ be the maximal indexed link used in $\mathbf{f}^{\mathrm{SE}}$, then we claim that $\{1, \ldots, j^*\}$ represents the group of links that will be used by both players and $\{j^* + 1, \ldots, \widehat{n}\}$ represents the group of links that will be used only by player 1.

For links in $\{j^* + 1, \ldots, \widehat{n}\}$, player 1 fills the latencies to the level that should occur in $\mathbf{f}^{\mathrm{SE}}$. Then, for links in $\{1, \ldots, j^*\}$, player 1 ensures that the difference between the latencies of every pairs of links after it has played stay the same. That is, player 1 splits the rest of its flow to all the links in this group such that every link is increased with the same amount of latency. The fraction of flow player 1 routes to link $j \in \{1, \ldots, j^*\}$ can be computed as follows. After allocating flow to links in $\{j^* + 1, \ldots, \widehat{n}\}$, the remaining flow of player 1 is $f^1 - \sum_{j=j^*+1}^{\widehat{n}} f^1_j$. Player 1 aims to increase every link with equal

latency, i.e. $a_i f_i^1 = a_j f_j^1$ for all $i, j \in \{1, \ldots, j^*\}$. We compute f_j^1 where $j \in \{1, \ldots, j^*\}$,

$$\begin{aligned} f^1 - \sum_{j=j^*+1}^{\widehat{n}} f_j^1 &= \sum_{j=1}^{j^*} \frac{a_j f_j^1}{a_j} \\ &= \frac{a_1 f_1^1}{a_1} + \ldots + \frac{a_j f_j^1}{a_j} + \ldots + \frac{a_{j^*} f_{j^*}^1}{a_{j^*}} \\ &= \frac{a_k f_k^1}{a_1} + \ldots + \frac{a_j f_j^1}{a_j} + \ldots + \frac{a_j f_j^1}{a_{j^*}} \qquad (4.1) \\ f_j^1 &= \frac{1}{a_k \sum_{j=1}^{j^*} 1/a_j} \left(f^1 - \sum_{j=j^*+1}^{\widehat{n}} f_j^1 \right), \end{aligned}$$

where 4.1 comes from the fact that every link is increased with equal latency, i.e. $a_j f_j^1$ are equal for every j in $\{1, \ldots, j^*\}$. By increasing latencies equally, player 1 ensures that player 2 routes its flow like $\mathbf{f}^{\mathrm{SE}}$ for $\{1, \ldots, j^*\}$. Hence the differences between links after player 2 has played are the same as that in $\mathbf{f}^{\mathrm{SE}}$. In summary, $f_j^1 + f_j^2 = f_j^{\mathrm{SE}}$ for every j in $\{1, \ldots, j^*\}$ and $f_j^1 = f_j^{\mathrm{SE}}$ for every j in $\{j^* + 1, \ldots, \widehat{n}\}$.

Lemma 4.2.1. *Given a flow of two players in a parallel-link network with linear latency functions, if player 1 uses Strategy 1 then player 1's cost in SSL equilibria is at most the cost of the optimal flow, i.e.* $C^1 \leq C^{SE}$.

Strategy 2 :

We consider Strategy 2 for a follower (or as we normally refer to as player 2) depicted in Figure 4.2. The aim of Strategy 2 is for player 2 to try to create an optimal flow by acting similarly to a UE flow. The strategy can be described as follows.

Player 2 first computes an optimal flow. Let $\widehat{n}$ be the maximal indexed link that is used in $\mathbf{f}^{\mathrm{SE}}$. Knowing how much flow player 1 has allocated on each link, form link 1 to link $\widehat{n}$, player 2 gradually distributes its flow to the the minimal latency link, i.e. link j with minimal $\ell_j(f_j^1 + f_j^2)$, but not to exceed the level of latencies in the optimal flow, i.e. $f_j^1 + f_j^2 \leq f_j^{SE}$.

Clearly, if player 2 routes its flow with Strategy 2, its cost is, at most, the optimal cost. This is because if player 1 does not allocate its flow more than the level that occurs in the optimal flow, then, using Strategy 2, player 2 will ensure that the combined flow is optimal. If there are some links that player 1 routes more than the optimal flow then there is enough room on the remaining links for player 2 to use those without making their flow more than the optimal flow.

1. Compute an optimal flow $\mathbf{f}^{\mathrm{SE}}$.
2. Let $\widehat{n} \leq n$ be the maximal with $f^{\mathrm{SE}}_{\widehat{n}} > 0$. in $\mathbf{f}^{\mathrm{SE}}$ and still have $f^1_j < f^{\mathrm{SE}}_j$ such that $\ell_1(f^1_1) \leq \ldots \leq \ell_n(f^1_n)$.
3. For $j = 1$ to $\widehat{n}$, set f^2_j such that $\ell_j(f^2_j + f^1_j) \leq \ell_j(f^2_{j+1} + f^1_{j+1})$ and $f^2_j + f^1_j \leq f^{\mathrm{SE}}_j$.

Figure 4.2: Strategy 2 for player 2

Lemma 4.2.2. *Given a flow of two players in a parallel-link network with linear latency functions, if player 2 uses Strategy 2 then the cost of player 2 at SSL equilibria is at most the cost of the optimal flow, i.e.* $C^2 \leq C^{SE}$.

Theorem 4.2.3. *Given a flow of two players in a parallel-link network with linear latency functions, the price of selfish Stackelberg leadership is at most 2.*

Proof. Recall that the price of SSL is the worst ratio of a social cost in a Stackelberg game and that in Nash equilibria. By using Lemma 4.2.1 and Lemma 4.2.2, the social cost in a Stackelberg game is at most $2C^{\mathrm{SE}}$. The social cost in Nash equilibria is at least C^{SE}, consequently, the price of SSL is at most $2C^{\mathrm{SE}}/C^{\mathrm{SE}} \leq 2$. □

Remark 4.2.4. It can simply be seen that, in two-player Stackelberg games in a parallel-link network, player 1 can always enforce an optimal flow by using Strategy 1. Although we assume here that the total flow is 1, Strategy 1 works with any positive total flow and with any fraction of flow a leader controls as Korilis et al. proved in [KLO97a]. This is clearly not necessary true for Strategy 2 for player 2.

4.3 Quick Upper Bound of 4/3

In the previous section, we oversimplify the upper bounds on players cost by using the costs of the combined flow as the bounds on individual costs. In fact, to get closer to true bounds, each player upper bound should be the cost of the combined flow minuses the minimal of the other player's cost, or the cost where that player routing all its flow on maximal latency links. Using this approach, we determine improved upper bounds for player 1 and player 2 in Lemma 4.3.2 and Lemma 4.3.3 respectively. Again using the combination of upper bounds on player costs, we prove the upper bound on the price of SSL of 4/3 in Theorem 4.3.4. Note that we still assume that player 1 uses Strategy 1 and player 2 uses Strategy 2.

Remark 4.3.1. We shall denote the minimal latency of links used in $\mathbf{f}^{\mathrm{SE}}$ by $\ell_{\min}$, and the maximal by $\ell_{\max}$. Since the total flow is always assumed to be one, we have that $\ell_{\min} \leq C^{\mathrm{SE}} \leq \ell_{\max}$. Obviously this also applies to the cost of player 1 and 2, i.e. $\ell^1_{\min} \leq C^1 \leq \ell^1_{\max}$ and $\ell^2_{\min} \leq C^2 \leq \ell^2_{\max}$ if ℓ^1 and ℓ^2 denote links used by player 1 and 2 respectively.

Lemma 4.3.2. *Given a parallel-link network with linear latency functions. In a Stackelberg game, player 1's cost is at most* $\min\{2f^1\ell_{\min},\, C^{SE} - (1-f^1)\ell_{\min}\}$.

Proof. Applying Strategy 1, player 1 ensures that the combined flow is socially optimal. Thus, player 1's cost is the optimal cost minus player 2's cost which is at least all of player 2's flow multiplied by the minimal latency, i.e. $(1-f^1)\ell_{\min}$ (recall that the total flow is one unit). Hence player 1's cost is at most $C^{\mathrm{SE}} - (1-f^1)\ell_{\min}$.

Alternatively, player 1's cost cannot be any higher than the cost of it putting all of its flow on the maximal latency links. Furthermore, from Corollary 2.2.4, the maximal latency of an optimal flow $\ell_{\max}$ is at most twice the minimal latency $\ell_{\min}$ for links that get used. Hence player 1's cost is at most $2f^1\ell_{\min}$.

Combining those two results, we have player 1's cost is at most $\min\{2f^1\ell_{\min},\, C^{\mathrm{SE}} - (1-f^1)\ell_{\min}\}$. □

Lemma 4.3.3. *Given a parallel-link network with linear latency functions. In a Stackelberg game, player 2's cost is at most* $\min\{C^{SE} - f^1\ell_{\min},\, 2(1-f^1)\ell_{\min}\}$.

Proof. In a Stackelberg game, we assume that player 2 uses Strategy 2. This implies that every link used by player 2 is at most ℓ_{max} which is at most $2\ell_{min}$ by Corollary 2.2.4. The cost of player 2 is maximised when all of player 2's flow is routed on maximal latency links, i.e. $C^2 \leq (1-f^1)2\ell_{min}$.

Alternatively, player 2's cost is at most the optimal cost minus player 1's cost. There are two cases to be considered.

Case 1: Player 2 can ensure that the minimum latency of the combined flow of all the links is at least $\ell_{\min}$. Hence we can pessimistically assume that player 1's cost is at least $f^1\ell_{\min}$ which means that player 2's cost is at most $C^{\mathrm{SE}} - f^1\ell_{\min}$.

Case 2: Player 2 *cannot* ensure that the minimum latency of the combined flow is at least $\ell_{\min}$. Then it uses links that all have latency less than $\ell_{\min}$. Thus its cost is less than $(1-f^1)\ell_{\min}$ which is less than the above bounding $2(1-f^1)\ell_{\min}$.

Combining those results, player 2's cost is at most $\min\{C^{\mathrm{SE}} - f^1\ell_{\min},\, 2(1-f^1)\ell_{\min}\}$. □

We are now ready to show the upper bound of 4/3 on the price of SSL.

Theorem 4.3.4. *In a two-player, n parallel-link network with linear latency functions, the price of SSL is at most 4/3.*

Proof. Suppose we are given a two-player, parallel-link network with linear latency functions. In a Stackelberg game, we assume that player 1 uses Strategy 1, and player 2 uses Strategy 2. Recall that an optimal cost is in the range between between $\ell_{\min}$ and $2\ell_{\min}$ (from Remark 4.3.1 and Corollary 2.2.4). There are two cases to be considered.

- The optimal cost is at least $(3/2)\ell_{\min}$. In this case, from Lemma 4.3.2 and 4.3.3, the social cost in a Stackelberg game is upper bounded by $2f^1\ell_{\min} + 2(1-f^1)\ell_{\min} = 2\ell_{\min}$. Hence the price of SSL is at most $(2\ell_{\min})/\left((3/2)\ell_{\min}\right) \leq 4/3$.
- The optimal cost is less than $(3/2)\ell_{\min}$. Then, similarly from Lemma 4.3.2 and 4.3.3, the social cost in a Stackelberg game is upper bounded by $C^{\mathrm{SE}} - (1 - f^1)\ell_{\min} + C^{\mathrm{SE}} - f^1\ell_{\min} = 2C_{\mathrm{SE}} - f^1\ell_{\min}$. Hence the price of SSL is upper bounded by $(2C^{\mathrm{SE}} - \ell_{\min})/(C^{\mathrm{SE}}) \leq 4/3$.

The combination of the above results proves the theorem.

□

4.4 Upper Bound of Less than 4/3

In this section, we prove that there is an upper bound on the price of SSL of less than 4/3. To do that, we improve the upper bound of player 1's cost in Lemma 4.3.2, in which the cost is bounded under the pessimistic assumption so that, in a situation where it creates an optimal flow, it is possible for all of player 1's flow to be routed on the maximal latency link and all of player 2's flow to be routed on the minimal latency link.

The outline of the proof of the main theorem in this section is as follows. We define the *diverse latency property* that an optimal flow may or may not have, depending on the latency functions of the links. We prove the upper bound in the case that the optimal flow has the diverse latency property, then we use an alternative proof in the case that it does not have the property. When the optimal flow has the property, we insist that player 1 uses the strategy called the *aloof strategy* [Rou04] that corresponds more with what selfish player 1 would do instead of Strategy 1.

Up to this point, we have shown that the price of SSL is at most 4/3 by pessimistically assuming that player 1 and 2 use the suboptimal Strategy 1 and 2 respectively in a Stackelberg game to enforce an optimal flow. With this approach, player costs depend on the optimal cost and the minimum latency of links used in the optimal flow. In addition, we know that the optimal cost is in the range between ℓ_{min} and $2\ell_{min}$, hence,

in Lemma 4.4.1 by assuming player 1 and 2 use Strategy 1 and 2 respectively, we show upper bounds on the price of SSL as C^{SE} changes in the range $[\ell_{\min}, 2\ell_{\min}]$. Similarly, with this approach, upper bounds on the price of SSL depend on the fraction of player 1. As the fraction of player 1 changes, we show in Lemma 4.4.3 bounds on the price of SSL. Figure 4.3 depicts the basic ideas by demonstrating upper bounds on the social cost of SSL equilibria as C^{SE} and f^1 vary. The lemmas will be useful in the next section when we prove that the price of SSL is at most 4/3.

Lemma 4.4.1. *In a two-player, parallel-link network with linear latency functions, let $C^{SE} = \gamma\ell_{\min}$ where $1 \leq \gamma \leq 2$. Then*

- *if $\gamma \geq \frac{3}{2}$ then the price of selfish Stackelberg leadership $\leq \frac{2}{\gamma}$;*
- *if $\gamma < \frac{3}{2}$ then the price of selfish Stackelberg leadership $\leq \frac{2\gamma-1}{\gamma}$.*

Proof. When the optimal cost is at least $(3/2)\ell_{\min}$, the social cost in the SSL setting is bounded by $2\ell_{\min}$ (Figure 4.3(b) and Corollary 2.2.4). Hence the price of SSL is upper bounded by

$$\frac{2\ell_{\min}}{C^{\text{SE}}} \leq \frac{2\ell_{\min}}{\gamma\ell_{\min}} \leq \frac{2}{\gamma}.$$

If the optimal cost is less than $(3/2)\ell_{\min}$ then the social cost in the SSL setting is bounded by $2C^{\text{SE}} - \ell_{\min}$ (Figure 4.3(c)) in which the social cost is player 1's cost of $C^{\text{SE}} - (1 - f^1)\ell_{min}$ plus player 2's cost of $C^{\text{SE}} - f^1\ell_{min}$. Hence the upper bound of the price of SSL is given by

$$\frac{2C^{\text{SE}} - \ell_{\min}}{C^{\text{SE}}} \leq 2 - \frac{\ell_{\min}}{\gamma\ell_{\min}} \leq \frac{2\gamma - 1}{\gamma}.$$

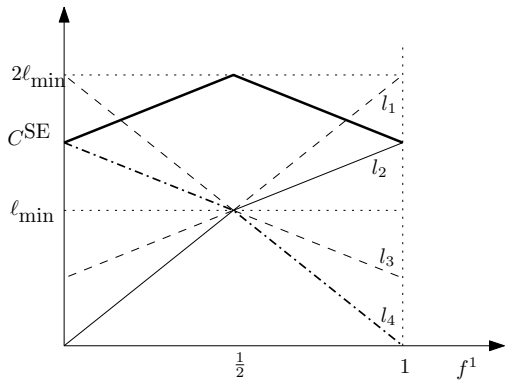
□

Remark 4.4.2. As suggested in Figure 4.3, the ratio between the upper bound for the social cost and the optimal cost is maximised when the optimal cost is $(3/2)\ell_{\min}$.

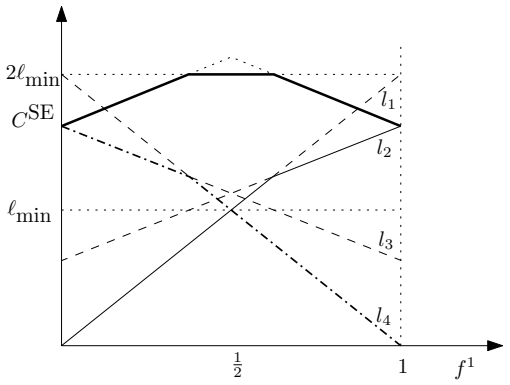
Lemma 4.4.3. *Suppose we have a two-player, parallel-link network. Let $f^1 = \alpha$ where $0 \leq \alpha \leq 1$ then:*

- *if $\alpha \geq \frac{1}{2}$ then the price of selfish Stackelberg leadership $\leq 1 + \frac{2(1-\alpha)}{3}$;*
- *if $\alpha < \frac{1}{2}$ then the price of selfish Stackelberg leadership $\leq 1 + \frac{2\alpha}{3}$.*

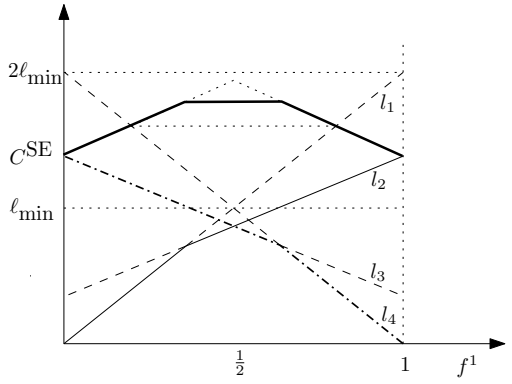
Proof. As mentioned above (Remark 4.4.2), the ratio between C^{SSL} and C^{SE} is maximised when $C^{\text{SE}} = (3/2)\ell_{\min}$. Firstly, we consider the case when $\alpha \geq 1/2$. In this case, player 1's cost and player 2's cost in the SSL setting are bounded by $C^{\text{SE}} - (1 - f^1)\ell_{\min}$



(a) Graph of the upper bound of C^{SSL} when $C^{\mathrm{SE}} = (3/2)\ell_{\min}$



(b) Graph of the upper bound of C^{SSL} when $C^{\mathrm{SE}} > (3/2)\ell_{\min}$



(c) Graph of the upper bound of C^{SSL} when $C^{\mathrm{SE}} < (3/2)\ell_{\min}$

Figure 4.3: Assuming that player 1 and 2 route flow with Strategy 1 and 2, the graphs show the upper bound on C^{SSL} as C^{SE} and f^1 of the corresponding flows vary. The graphs depict the basic ideas for Lemma 4.4.1 and Lemma 4.4.3. Dashed lines denote player costs as follows: for C^1, $l_1 = f^1\ell_{\max}$ and $l_2 = C^{\mathrm{SE}} - (1 - f^1)\ell_{\min}$; for C^2, $l_3 = C^{\mathrm{SE}} - f^1\ell_{\min}$ and $l_4 = (1 - f^1)\ell_{\max}$. Solid lines denote the minimal upper bound of C^1. Dot dashed lines denote the minimal upper bound of C^2. Thick solid lines denote the upper bound for C^{SSL} which is $\min\{l_1, l_2\} + \min\{l_3, l_4\}$.

and $(1-f^1)\ell_{\max}$ respectively (using Lemma 4.3.2 and 4.3.3). Then the social cost is bounded by $C^{\mathrm{SE}} + (1-f^1)\ell_{\min}$ (using Corollary 2.2.4 for $\ell_{\max} \leq 2\ell_{\min}$). Thus, the price of SSL is, at most,

$$\frac{C_{\mathrm{SE}} + (1-f^1)\ell_{\min}}{C_{\mathrm{SE}}} \leq 1 + \frac{(1-\alpha)\ell_{\min}}{(3/2)\ell_{\min}} \leq 1 + \frac{2(1-\alpha)}{3}.$$

Next, we consider $\alpha < 1/2$. In this case, player 1's cost and player 2's have minimal upper bounds of $C^{\mathrm{SE}} - f^1\ell_{\min}$ and $2f^1\ell_{\min}$ respectively. Consequently, the social cost is bounded by $C^{\mathrm{SE}} + f^1\ell_{\min}$. Hence, the price of SSL in this case is upper bounded by

$$\frac{C_{\mathrm{SE}} + f^1\ell_{\min}}{C_{\mathrm{SE}}} \leq 1 + \frac{\alpha\ell_{\min}}{(3/2)\ell_{\min}} \leq 1 + \frac{2\alpha}{3}.$$

□

Using Lemma 4.4.1 and 4.4.3, we narrow the scope of the values of the optimal cost and player 1's flow needs to be considered. Finally, we prove the main result by conducting a case analysis of the value of C^{SE} and f^1.

Definition 4.4.4. *Let $\ell_{\min}$ and $\ell_{\max}$ respectively denote the minimal and maximal latency in an optimal flow. We say that an optimal flow $\mathbf{f}^{SE}$ has the* diverse latency property *(DLP) if at least 1/4 of the flow gets a latency of at most, $1.16\ell_{\min}$ and at least 1/4 of the flow gets a latency of at least $1.84\ell_{\min}$.*

Definition 4.4.5. *Aloof strategy: (Roughgarden [Rou04]): in a two-player model, player 1 routes f^1, optimising its cost in player 2's absence: compute the optimal flow for a total flow volume of $\mathbf{f}^1$.*

Lemma 4.4.6. *Given a two-player, parallel-link network with linear latency functions. If the DLP is satisfied by an optimal flow, then player 1's cost is at most $1.915f^1\ell_{\min}$.*

Proof. Let $S_{\min}$ and $S_{\max}$ be the sets of links whose latencies in the optimal flow are at most $1.16\ell_{\min}$ and at least $1.84\ell_{\min}$ respectively. We will assume in this proof that player 1 uses the aloof strategy. We exploit the fact that, when the DLP holds, player 1 gets a better performance using the aloof strategy instead of Strategy 1, as described earlier.

First, we consider the optimal flow. Using the DLP assumption and Lemma 2.2.5, the difference between the constant cost $\ell_j(0)$ (i.e. the marginal costs of the links when the flow is zero) of links in $S_{\max}$ and those in $S_{\min}$ is at least twice the difference between the minimal latency in $S_{\max}$ and the maximal latency in $S_{\min}$, i.e. $2(1.84\ell_{\min} - 1.16\ell_{\min}) = 1.36\ell_{\min}$.

Next, we consider the SSL setting. With the aloof strategy, if player 1 uses links in $S_{\max}$, after player 1 has played, the latency difference between $S_{\max}$ and $S_{\min}$ is at least half the difference between the value $\ell_j(0)$ in $S_{\max}$ and that in $S_{\min}$, which is at least $(1.36\ell_{\min})/2 = 0.68\ell_{\min}$. And if player 1 does not use links in $S_{\max}$ then the difference is higher than $0.68\ell_{\min}$. Essentially, this implies that, from player 2's perspective the constant cost in $S_{\max}$ is at least $0.68\ell_{\min}$ more than that in $S_{\min}$.

By Corollary 2.2.4 and Lemma 2.2.5, there is not enough flow in total for the latency in the $S_{\max}$ of links used by either player to be more than $2\ell_{\min}$. Therefore, the latency in $S_{\min}$ after player 2 has played is at most $2\ell_{\min} - (0.68\ell_{\min})/2 = 1.66\ell_{\min}$.

With the DLP assumption, because there is at least 1/4 of the total flow in $S_{\min}$, player 1 is guaranteed to have at least 1/4 of its flow in $S_{\min}$. Hence, player 1's cost is, at most, $(f^1/4)(1.66\ell_{\min}) + (3f^1/4)2\ell_{\min} = 1.915f^1\ell_{\min}$. □

For the case when the DLP is not satisfied, we prove the following upper bound on player 1's cost.

Lemma 4.4.7. *If, in an optimal flow, at most* 1/4 *of the flow gets a latency of at most* $1.16\ell_{\min}$ *or at most* 1/4 *of the flow gets a latency of at least* $1.84\ell_{\min}$*, then player 1's cost is at most* $\max\{C^{SE} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4 - f^1),\ \ell_{\min}/2 + (f^1 - 1/4)(1.84\ell_{\min})\}$.

Proof. In this proof, we assume that player 1 uses Strategy 1 to ensure that the combined flow is an optimal flow. Then, from Lemma 4.3.2, player 1's cost is at most the optimal cost minus player 2's cost which is lowest when all of player 2's flow f^2 achieves a latency of $\ell_{\min}$. However, noting the first alternative of the given assumption, suppose 1/4 of the total flow achieves a latency of $\ell_{\min}$, and the rest gets at least $1.16\ell_{\min}$, then player 1's cost is, at most, $C^{\mathrm{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4 - f^1)$.

Alternatively, player 1's cost is maximised when all of f^1 is on the maximal latency links. However, from the second alternative of the given assumption, only 1/4 of the flow achieves a latency of more than $1.84\ell_{\min}$ and the rest of the flow achieves a latency of at most $1.84\ell_{\min}$. Hence player 1's cost can be at most $(1/4)(2\ell_{\min}) + (f^1 - 1/4)(1.84\ell_{\min}) = \ell_{\min}/2 + (f^1 - 1/4)(1.84\ell_{\min})$. □

Theorem 4.4.8. *In a two-player, n-parallel-link model with non-decreasing linear latency functions, the price of selfish Stackelberg leadership is in the range* $[1.075, 1.322]$.

Proof. In regard to a lower bound, we have obtained a price of SSL of 1.075 in Example 3.3.1.

In regard to an upper bound, we start the proof by first identify the scope of an optimal cost and player 1's flow, to which we should restrict our attention. From Lemma 4.4.3, we note that we only consider player 1's flow in the range $[0.483, 0.517]$

since any scenario in which player 1's flow is outside this range gives the price of SSL of less than 1.322. Similarly, because of Lemma 4.4.1, we only consider the optimal cost in the range $[1.474\ell_{\min}, 1.513\ell_{\min}]$.

We will use the upper bound of player 2's cost from Lemma 4.3.3. For player 1's cost, we have two upper bounds, one for when the DLP holds and the other for when it does not hold.

First, when the DLP is satisfied, we have an upper bound on player 1's cost of $1.915f^1\ell_{\min}$ from Lemma 4.4.6. Combined with player 2's cost, the price of SSL is upper bounded by

$$\begin{aligned}\frac{1.915f^1\ell_{\min} + \min\{C^{\mathrm{SE}} - f^1\ell_{\min},\ 2(1-f^1)\ell_{\min}\}}{C^{\mathrm{SE}}} &\leq \frac{1.915f^1\ell_{\min} + C^{\mathrm{SE}} - f^1\ell_{\min}}{C^{\mathrm{SE}}} \\ &\leq 1 + \frac{0.915f^1\ell_{\min}}{C^{\mathrm{SE}}} \\ &\leq 1 + \frac{0.915(0.517)\ell_{\min}}{1.474\ell_{\min}} \\ &\leq 1.321\end{aligned}$$

Secondly, when the DLP is not satisfied, player 1's cost is upper bounded by $\max\{C^{\mathrm{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4 - f^1), \ell_{\min}/2 + (f^1 - 1/4)(1.84\ell_{\min})\}$. Within this proof, we will use (1) and (2) to denote the expression $C^{\mathrm{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4 - f^1)$ and $\ell_{\min}/2 + (f^1 - 1/4)(1.84\ell_{\min})$ respectively. Thus the upper bound for player 1's cost can be represented by $\max\{(1), (2)\}$. We prove the upper bound by an exhaustive case analysis as follows:

1. Suppose $C^{\mathrm{SE}} \geq (3/2)\ell_{\min}$, we consider f^1.

   - Suppose $f^1 < 1/2$, then $\max\{(1), (2)\} = (1)$. Hence the price of SSL is upper bounded by

$$\begin{aligned}&\frac{(1) + \min\{C^{\mathrm{SE}} - f^1\ell_{\min},\ 2(1-f^1)\ell_{\min}\}}{C^{\mathrm{SE}}} \\ &\qquad\leq \frac{C^{\mathrm{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4 - f^1) + 2(1-f^1)\ell_{\min}}{C^{\mathrm{SE}}} \\ &\qquad\leq 1 + \frac{(0.88 - 0.84f^1)\ell_{\min}}{C^{\mathrm{SE}}} \\ &\qquad\leq 1 + \frac{(0.88 - 0.84(0.483))\ell_{\min}}{(3/2)\ell_{\min}} \\ &\qquad\leq 1.317\end{aligned}$$

- Suppose $f^1 \geq 1/2$, we have player 2's cost $\leq 2(1-f^1)\ell_{\min}$, but player 1's cost is still bounded by $\max\{(1),(2)\}$. The upper bound on the price of SSL can be considered in two cases.
  (a) When $\max\{(1),(2)\} = (1)$ the price is at most $((1)+2(1-f^1)\ell_{\min})/C_{\text{SE}}$. Thus we have

$$\begin{aligned}\frac{(1)+2(1-f^1)\ell_{\min}}{C^{\text{SE}}} &\leq \frac{C^{\text{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4-f^1) + 2(1-f^1)\ell_{\min}}{C^{\text{SE}}} \\ &\leq 1 + \frac{(0.88-0.84f^1)\ell_{\min}}{C^{\text{SE}}} \\ &\leq 1 + \frac{(0.88-0.84(0.5))\ell_{\min}}{(3/2)\ell_{\min}} \\ &\leq 1.307\end{aligned}$$

  (b) When $\max\{(1),(2)\} = (2)$ the price is at most $((2)+2(1-f^1)\ell_{\min})/C_{\text{SE}}$. Thus we have

$$\begin{aligned}\frac{(2)+2(1-f^1)\ell_{\min}}{C^{\text{SE}}} &\leq \frac{\ell_{\min}/2 + (f^1-1/4)(1.84\ell_{\min}) + 2(1-f^1)\ell_{\min}}{C^{\text{SE}}} \\ &\leq \frac{(2.04-0.16f^1)\ell_{\min}}{C^{\text{SE}}} \\ &\leq \frac{(2.04-0.16(0.5))\ell_{\min}}{(3/2)\ell_{\min}} \\ &\leq 1.307\end{aligned}$$

2. Suppose $C^{\text{SE}} < (3/2)\ell_{\min}$, we consider f^1
   - Suppose $f^1 < 1/2$ then player 2's cost is at most $C^{\text{SE}} - f^1\ell_{\min}$. Hence, the upper bound of the price of SSL can be considered in two cases.
     (a) When $\max\{(1),(2)\} = (1)$, the price is at most $((1)+C^{\text{SE}}-f^1\ell_{\min})/C^{\text{SE}}$. We have

$$\begin{aligned}\frac{(1)+C^{\text{SE}}-f^1\ell_{\min}}{C^{\text{SE}}} &\leq \frac{C^{\text{SE}} - \ell_{\min}/4 - 1.16\ell_{\min}(3/4-f^1) + C^{\text{SE}} - f^1\ell_{\min}}{C^{\text{SE}}} \\ &\leq 2 + \frac{(-1.12+0.16f^1)\ell_{\min}}{C^{\text{SE}}} \\ &\leq 2 + \frac{(-1.12+0.16(0.5))\ell_{\min}}{(3/2)\ell_{\min}} \\ &\leq 1.307\end{aligned}$$

(b) When $\max\{(1),(2)\} = (2)$, the price is at most $((2)+C^{\mathrm{SE}}-f^1\ell_{\min})/C^{\mathrm{SE}}$. So we have

$$\begin{aligned}\frac{(2)+C^{\mathrm{SE}}-f^1\ell_{\min}}{C^{\mathrm{SE}}} &\le \frac{\ell_{\min}/2+(f^1-1/4)(1.84\ell_{\min})+C^{\mathrm{SE}}-f^1\ell_{\min}}{C^{\mathrm{SE}}}\\ &\le 1+\frac{(0.04+0.84f^1)\ell_{\min}}{C^{\mathrm{SE}}}\\ &\le 1+\frac{(0.04+0.84(0.5))\ell_{\min}}{1.474\ell_{\min}}\\ &\le 1.313\end{aligned}$$

- Suppose $f^1 \ge 1/2$ then $\max\{(1),(2)\} = (2)$, then the price of SSL is upper bounded by

$$\begin{aligned}&\frac{(2)+\min\{C^{\mathrm{SE}}-f^1\ell_{\min},\ 2(1-f^1)\ell_{\min}\}}{C^{\mathrm{SE}}}\\ &\qquad\le \frac{(2)+C^{\mathrm{SE}}-f^1\ell_{\min}}{C^{\mathrm{SE}}}\\ &\qquad\le \frac{\ell_{\min}2+(f^1-1/4)(1.84\ell_{\min})+C^{\mathrm{SE}}-f^1\ell_{\min}}{C^{\mathrm{SE}}}\\ &\qquad\le 1+\frac{(0.025+0.9f^1)\ell_{\min}}{C^{\mathrm{SE}}}\\ &\qquad\le 1+\frac{(0.04+0.84(0.517))\ell_{\min}}{1.474\ell_{\min}}\\ &\qquad\le 1.322\end{aligned}$$

In conclusion, when the DLP is not satisfied, we have shown that the price of SSL is less than 1.322. □

4.5 Homogeneous Linear Latency Functions

In this section, we consider the model with homogeneous linear latency functions—linear functions of the form $\ell(x) = ax$ where $a > 0$. It was shown by Hayrapetyan et al. [HTW06] that, for parallel-link networks with convex latency functions, every link has the same latency in both a Nash flow and a UE flow, which implies that the social costs of a Nash flow and a UE flow are equal. The main result in this section partially extends the result of Hayrapetyan et al. in a sense that we further show that not only will every link in the Nash flow ends up with the same latency, but also the ratios of the flows on the links are the same for all players. Furthermore, we prove that a SSL flow is identical to a Nash flow in this case, which shows that having Stackelberg leadership,

in this setting, is not beneficial.

The main result in this section is that the price of SSL is 1 which will be proved via the following steps. Essentially, we prove the main result (Theorem 4.5.9) by showing that a Nash flow and a SSL flow in this setting are unique and identical. We define a term *equal shared flow* ($\mathbf{f}^{ES}$) for a flow in which every players split their flow in a uniform way as described below in Definition 4.5.1. We show that a Nash flow and a SSL flow are $\mathbf{f}^{ES}$ in Lemma 4.5.3 and Lemma 4.5.8 respectively. To show that a SSL flow is $\mathbf{f}^{ES}$, we first prove that the leader must split its flow such that every links are increased with an equal latency (Lemma 4.5.7), which then would force the follower to do the same (Lemma 4.5.6).

Definition 4.5.1. *Given a two-player, parallel-link network with homogeneous linear latency functions, a flow is* the equal share flow *if both players allocate their flow such that they increase the latency of every link equally, i.e., for player i, $a_j f_j^i = a_k f_k^i$ for every pair of links j and k. We shall denote the equal share flow of $mathbf f$ with $\mathbf{f}^{ES}$.*

Clearly, in an equal share flow, every links have an equal latency.

Corollary 4.5.2. *For a parallel-link network with homogeneous linear latency functions, all of the latencies of the links in $\mathbf{f}^{ES}$ are equal.*

Lemma 4.5.3. *Let $\mathbf{f}$ be a flow of two players in a parallel-link network with homogeneous linear latency functions. At Nash equilibrium, every link will be used, i.e. $f_j > 0$ for every link j in the network.*

Proof. Recall from Lemma 2.3.4 that, at Nash equilibrium, for all links j and k and for all players i that have $f_j^i > 0$, if the following inequality holds,

$$\ell_j(f_j) + f_j^i \cdot \ell_j'(f_j) \leq \ell_k(f_k) + f_k^i \cdot \ell_k'(f_k).$$

Let us suppose that link k is unused, i.e. $f_k = 0$. The right-hand side of the inequality is $a_k f_k + f_k^1(a_k) = 0$. Then, at least one and possibly both, of the players will want to move some of its flow to k. □

Lemma 4.5.4. *A flow $\mathbf{f}$ of two players in a parallel-link network with homogeneous linear latency functions is at Nash equilibrium if $\mathbf{f}$ is $\mathbf{f}^{ES}$.*

Proof. From Lemma 2.3.4, a flow is at Nash equilibrium for all links j and k and for all players i that have $f_j^i > 0$, if the following inequality holds,

$$\ell_j(f_j) + f_j^1 \cdot \ell_j'(f_j) \leq \ell_k(f_k) + f_k^1 \cdot \ell_k'(f_k).$$

Let $\mathbf{f}$ be a flow of two players in a parallel-link network with homogeneous linear latency functions. We show that if $\mathbf{f} = \mathbf{f}^{\mathrm{ES}}$ then $\mathbf{f}$ satisfies the above inequality. And let us suppose $\mathbf{f} = \mathbf{f}^{\mathrm{ES}}$, namely $a_j f_j^i = a_k f_k^i$ for $i = \{1, 2\}$ and for every pair of links j and k in the network. From Lemma 4.5.3, since all the links are used, the above inequality becomes

$$\ell_j(f_j) + f_j^1 \cdot \ell_j'(f_j) = \ell_k(f_k) + f_k^1 \cdot \ell_k'(f_k)$$
$$\ell_j(f_j) + f_j^1 \cdot a_j = \ell_k(f_k) + f_k^1 \cdot a_k$$

which is satisfied by $\mathbf{f}$ from Corollary 4.5.2, and the assumption $a_j f_j^i = a_k f_k^i$. □

Next, we consider a Stackelberg game in this setting. We show that, to optimise its cost, the first player must route its flow such that every link has an equal latency. To achieve that, we first show the following lemma. The Lemma states that, assuming player 2 optimises its cost relative to player 1's strategy, if player 1 transfers a small amount of its flow t from one link to another, then player 2 transfers its flow $t/2$ back in the opposite direction. Then, we show that, in the SSL setting, if player 1 routes its flow such that every link is increased with equal latency, then player 2 will do the same.

Lemma 4.5.5. *Let $\mathbf{f}$ be a flow of two players in a parallel-link network with homogeneous linear latency functions, and $\mathbf{f}^1$ and $\mathbf{f}^2$ be player 1's strategy and player 2's strategy respectively in the SSL setting. If player 1 transfers a small fraction of its flow $t > 0$ from link j to link k, then player 2 reacts by transferring at most $t/2$ of its flow back from link k to link j.*

Proof. Let us suppose that $t > 0$ is the small fraction of player 1's flow on link j that will be transferred, and let $t' > 0$ be the fraction of player 2's flow on link k that will be transferred back to link j. Before the transfer, the flow on link j and link k are f_j and f_k respectively. If the strategy of player 1 is $\mathbf{f}^1 = (\ldots, f_j^1, f_k^1, \ldots)$, then, while holding $\mathbf{f}^1$ fixed we can apply Proposition 2.4.5 to optimise player 2's cost. Let $\tilde{\ell}_j(x) = \ell_j(x + f_j^1)$ for every link j in the network, and we have the following property hold when the cost of player 2 is optimised,

$$\tilde{\ell}_j'(f_j^2) = \tilde{\ell}_k'(f_k^2)$$
$$a_1(f_j^1 + f_j^2) + f_j^2 \cdot a_j = a_k(f_k^1 + f_k^2) + f_k^2 \cdot a_k. \tag{4.2}$$

After player 1 has transferred the flow, player 2 reacts by transferring back t' from link k to link j. Again, optimising player 2's cost while holding player 1' strategy fixed,

we have

$$\begin{aligned} a_j(f_j^1 + f_j^2 - t + t') + (f_j^2 + t')a_j &= a_k(f_k^1 + f_k^2 + t - t') + (f_k^2 - t')a_k \\ a_j(f_j^1 + f_j^2) + a_j(-t + t') + f_j^2 \cdot a_j + t' \cdot a_j &= a_k(f_k^1 + f_k^2) + a_k(t - t') + f_k^2 \cdot a_k - t' \cdot a_k \\ a_j(-t + t') + t' \cdot a_j &= a_k(t - t') - t' \cdot a_k \qquad (4.3) \\ t' &= \frac{t}{2} \end{aligned}$$

where (4.3) is obtained from using (4.2). Therefore, player 2 can transfer back $\min(f_k^2, t/2)$ of its flow on link k, which gives us the stated result. □

Lemma 4.5.6. *Let* $\mathbf{f}$ *be a flow of two players in a parallel-link network with homogeneous linear latency functions. In a Stackelberg game, if player 1 distributes its flow such that every link is increased with equal latency, then player 2 will do the same.*

Proof. a flow of two players in a parallel-link network with homogeneous linear latency functions. Let us suppose that player 1 routes its flow such that every link is increased with equal latency, i.e. $a_j f_j^1 = a_k f_k^1$ for every pair of links j and k in the network. Let $\tilde{\ell}_j(x) = \ell_j(x + f_j^1)$ for every link j in the network. While holding player 1's strategy fixed, we apply Proposition 2.4.5 to optimise player 2's cost. For all links j and k, we have

$$\begin{aligned} \tilde{\ell}_j'(f_j^2) &= \tilde{\ell}_k'(f_k^2) \\ a_j(f_j^1 + f_j^2) + f_j^2 \cdot a_j &= a_k(f_k^1 + f_k^2) + f_k^2 \cdot a_k \\ 2a_j \cdot f_j^2 &= 2a_k \cdot f_k^2. \end{aligned}$$

This gives us the stated result. □

Lemma 4.5.7. *Let* $\mathbf{f}$ *be a flow of two players in a parallel-link network with homogeneous linear latency functions. In the SSL setting, the first player routes its flow such that every link has equal latency.*

Proof. Given a flow $\mathbf{f}$ of two players in a parallel-link network with homogeneous linear latency functions. Let us suppose that player 1 routes its flow such that every link is increased with equal latency, i.e. $a_j f_j^1 = a_k f_k^1$ for every pair of links j and k in the network. Let us suppose that player 1 can still decrease its cost by transferring a small amount of flow $t > 0$ from link 1 to link 2. Let f_1 and f_2 be the flow on link 1 and link 2 respectively before the transfer. By using Lemma 4.5.6, we know that $\ell_1(f_1) = \ell_2(f_2)$.

Before the transfer, the cost of player 1 is

$$f_1^1\ell_1(f_1) + f_2^1\ell_2(f_2) + \sum_{j\neq 1 \text{ or } 2} f_j^1\ell_j(f_j). \tag{4.4}$$

After the flow transfer of player 1, player 2 reacts by transferring some of its flow on link 2 back to link 1, Lemma 4.5.5. Let $t' > 0$ be the flow of player 2 on link 2 that will be transferred back. The cost of player 1 after the transfers is

$$(f_1^1 - t)\ell_1(f_1 - t + t') + (f_2^1 + t)\ell_2(f_2 + t - t') + \sum_{j\neq 1 \text{ or } 2} f_j^1\ell_j(f_j). \tag{4.5}$$

The potential gain, (4.4) - (4.5), must be positive for player 1 to have any incentive to transfer. We have

$$\begin{aligned}
f_1^1\ell_1(f_1) + f_2^1\ell_2(f_2) - (f_1^1 - t)\ell_1(f_1 - t + t') - (f_2^1 + t)\ell_2(f_2 + t - t') &> 0 \\
f_1^1 \cdot a_1 \cdot f_1 + f_2^1 \cdot a_2 \cdot f_2 - (f_1^1 - t)a_1(f_1 - t + t') - (f_2^1 + t)a_2(f_2 + t - t') &> 0 \\
f_1^1 \cdot a_1(t - t') + t \cdot a_1(f_1 - t + t') - f_2^1 \cdot a_2(t - t') - t \cdot a_2(f_2 + t - t') &> 0 \\
t \cdot a_1(f_1 - t + t') - t \cdot a_2(f_2 + t - t') &> 0 \qquad (4.6) \\
t \cdot a_1 \cdot f_1 - t \cdot a_1(t - t') - t \cdot a_2 \cdot f_2 - t \cdot a_2(t - t') &> 0 \\
-t(t - t')(a_1 + a_2) &> 0, \qquad (4.7)
\end{aligned}$$

note that (4.6) is obtained by using the assumption $a_j f_j^1 = a_k f_k^1$ for all links j and k, (4.7) is obtained from the fact that $\ell_1(f_1) = \ell_2(f_2)$.

On the left-hand side of (4.7), $-t(t - t')(a_1 + a_2)$ is always negative because $t > 0$ and t' is at most $t/2$, Lemma 4.5.5. Consequently, player 1 cannot transfer any more flow to decrease its cost. Hence, we have the stated result. □

Lemma 4.5.8. *Let* $\mathbf{f}$ *be a flow of two players in a parallel-link network with homogeneous linear latency functions.* $\mathbf{f}$ *is at SSL equilibrium if* $\mathbf{f}$ *is the equal share flow.*

Proof. By using Lemma 4.5.7 and Lemma 4.5.6, in the SSL setting, both players split its flow such that every link is increased with equal latency, i.e. $\mathbf{f}$ is the equal share flow. □

Theorem 4.5.9. *Let* $\mathbf{f}$ *be a flow of two players in a parallel-link network with homogeneous linear latency functions, so the price of selfish Stackelberg leadership of* $\mathbf{f}$ *is 1.*

Proof. From Lemma 4.5.4 and Lemma 4.5.8, the Nash flow and the SSL flow of $\mathbf{f}$ are identical; hence the social cost is the same, i.e. the price of SSL is 1. □

4.6 Conclusions

We have shown that the upper bound of the price of selfish Stackelberg leadership is a multiplicative constant of 1.322 that is independent of the number of links in a parallel-links network with linear latency functions. It is possible that for parallel links, the worst-case arises for a 2-link network (by analogy to [Rou02]) which we obtain the lower bound of 1.075. In conclusion, we obtain quite a narrow bound of $[1.075, 1.322]$ for the price of SSL.

In the case where latency functions are restricted to homogeneous linear, we show that a Nash flow and a SSL flow are not only identical but also unique. As a result, we have proved that the price of SSL in this setting is one.

Perhaps the main question to ask is whether there is a more dramatic cost (perhaps depending on the size of the network) in the setting of more general networks.

One alternative line of work is investigating the price of SSL in a model of one selfish splittable leader and the rest of the players, each with a negligible fraction of the flow (a Wardrop flow). We believe that there is no price of SSL in this setting since the SSL solution is essentially the same as Nash equilibrium.

Chapter 5

Bounds for the Convergence Rate of a Randomised Local Search in a Load-balancing Game with Variable-capacity Resources

Up until now, we have been investigating the problem of the quality of a Nash equilibrium, setting aside the issue of the computation of a Nash equilibrium. In this chapter, we study a simple algorithm for finding Nash equilibria called "Randomised Local Search" (RLS) that can simulate a network of selfish players. In particular, our main result shows upper bounds for the convergence rate of RLS in a simple load-balancing game with variable-capacity resources.

5.1 Overview

We consider a load-balancing game which essentially is a parallel-link network with homogeneous linear latency functions that we studied in the previous chapters. In particular, we study the problem of constructing Nash equilibrium by a simple algorithm called Randomised Local Search (RLS) that (as we will show) can be realised by a simple distributed network of selfish users that have no central control and only interact via the effect they have on the latency functions.

RLS can be informally described as follows. Initially, each player is assigned a link, and then, at each step, a player and a link are selected uniformly at random. The selected player moves its flow to that resource if its resulting cost is lower. That move is called "self-improving", as recent works in this field showed that an algorithm

that repeatedly makes this kind of moves will eventually reach Nash equilibrium. Our contribution here is to investigate the question of how long does RLS takes to reach Nash equilibrium in a special case where player flows are an integer?

In Section 5.2, we start by introducing the load-balancing model and note some of its similarities and differences to the parallel-link network models studied in the previous chapters. In Section 5.3, we present a distributed version of RLS that has been shown previously by Goldberg [Gol04]. In Section 5.4, we show the first result of this chapter. We prove that there is an upper bound on the expected time for RLS to reach Nash equilibrium. We exhibit a bound on the convergence rate that is polynomial in terms of the number of players, the number of links and the maximum player flow in the network. In Section 5.5, through a more careful analysis, we place a better upper bound on the convergence rate in a sense that the number of players has been decreased to quadratic power. This can be useful because in a distributed setting, the number of player in the expected number of attempts becomes linear. We close the chapter with the conclusions in Section 5.6.

5.2 The Load-balancing Model

We continue to use most of the definitions and notations defined earlier in Section 2.1. The main difference between the model in this chapter and those in earlier chapters is that, here, we focus on a pure strategy; namely every player selects exactly one link to route its flow. In other words, for player $i \in M$, a strategy S^i of player i is a link j where $j \in E$. Another difference is that there are m players in the network in this chapter, while we usually assume the case of two players in the previous chapters. Furthermore, we will assume the player flows to be a positive integer.

Recall that a flow $\mathbf{f}$ denotes a vector of strategies, one for each player. We redefine a flow $\mathbf{f}$ as follows. If we are given an initial flow $\mathbf{f}$, let $\lambda_0(\mathbf{f}) = \mathbf{f}$, and $\lambda_{h+1}(\mathbf{f})$ is obtained from $\lambda_h(\mathbf{f})$ at step h by performing a single transfer of one player flow from one link to another. The total quantity of a flow $\mathbf{f}$ is a positive integer number f, that is not necessarily one unit, as previously assumed. As regards the strategies of the players, $S^i(\lambda_h)$ denotes a strategy of player i in a flow λ_h at step h.

Following from typical load-balancing games, we assume that each link $j \in E$ has an associated capacity $c_j \in \mathbb{Z}^+$. Without a loss of generality, we will assume that the minimal capacity is 1 and an integer $c_{\max}$ denotes the maximal capacity. Let $f_j(\lambda_h)$ denote the fraction of the total flow assigned to link $j \in E$ in λ_h. Each player $i \in M$ controls a positive integer flow $f^i \in \mathbb{Z}^+$. Similar to the capacity of the links, we will assume that the minimum flow of player is 1 and an integer $f_{\max}$ denotes the maximum flow of the player. The latency of using link j at λ_h is defined to be the total of flow

loaded onto j at λ_h divided by the capacity of j , i.e. $\ell_j(\lambda_h) = f_j(\lambda_h)/c_j$. This is similar to the homogeneous linear latencies in which $1/c_j$ is the coefficient. The cost of a player at a flow λ_h, in this chapter, is now defined as the cost of link $\ell_j(\lambda_h)$, where j is the link used by that player. That is, every players that use link j receives the common cost equal to the latency of the link that they share.

Note that, often we drop the notation $\mathbf{f}$ and λ_h if they are obvious from the content or do not need to be denoted.

Definition 5.2.1. *The cost experienced by a player that routes its flow on link $j \in E$ with respect to a flow λ_h is the latency of link j; that is, the sum of all player flows that are routed on link j, divided by the capacity of j. For every player i*

$$C^i(\lambda_h) = \frac{f_j(\lambda_h)}{c_j}, \tag{5.1}$$

provided that $S^i(\lambda_h) = j$.

We borrow the potential function from [EDKM07] that will map a flow onto a real value. It was shown in Lemma 4.1 in that paper that, when a player makes a self-improving move, the potential in the following definition decreases.

Definition 5.2.2. *[EDKM07] The potential function of a flow λ_h is defined as the real-value function P:*

$$P(\lambda_h) = \sum_{j \in E} \frac{(f_j(\lambda_h))^2}{c_j} + \sum_{i \in M} \frac{(f^i)^2}{c_{S^i(\lambda_h)}} \tag{5.2}$$

To set a minimal value of the potential function to 0, we subtract the function with the minimal possible value P^* of potential function for any flow.

Definition 5.2.3. *For a flow λ_h, the potential function of λ_h is*

$$P(\lambda_h) = \sum_{j \in E} \frac{(f_j(\lambda_h))^2}{c_j} + \sum_{i \in M} \frac{(f^i)^2}{c_{S^i(\lambda_h)}} - P^* \tag{5.3}$$

where P^* is the smallest value for any flow λ_h that can be taken by $\sum_{j \in E}(f_j(\lambda_h))^2/c_j + \sum_{i \in M}(f^i)^2/c_{S^i(\lambda_h)}$.

We consider the Randomised Local Search algorithm presented in Figure 5.1. Basically, the algorithm at any one time makes an *attempt* by randomly choosing a single player and a single link, an attempt is successful when the chosen player reduce the potential value, defined in Definition 5.2.3, of the resulting flow by reallocating its flow to the selected link. We call a successful attempt a *move*. We define the h-th move

Given an initial flow $\mathbf{f}$,

Repeatedly do the following:

1. Choose a player $i \in M$ from a uniform distribution.
2. Choose a link $j \in E$ from a uniform distribution.
3. Assign link j to the strategy of player i if and only if the potential value of the resulting flow is lower.

Figure 5.1: Randomised Local Search algorithm

to be a move of one player's flow from λ_h to λ_{h+1}. We use Definition 2.3.3 for Nash equilibrium which implies that the model is said to reach pure Nash equilibrium if there is no more move available. We study the number of attempts that RLS takes to reach Nash equilibrium in this load-balancing game setting.

5.3 A Distributed version of Randomised Local Search

Given an initial flow $\mathbf{f}$, let j be the link player i uses.

Repeatedly do the following:

1. Generate a delay δ from an exponential distribution $P(\delta) = \exp(-\delta)$; wait δ time units.
2. Generate a link k from a uniform distribution.
3. Evaluate the cost of using k instead of j; if it is lower, replace j with k.

Figure 5.2: A Distributed version of Randomised Local Search (Goldberg [Gol04]); used by each player.

In this section, we present the results proved by Goldberg in [Gol04] to show that RLS can be realised as a distributed network, i.e. RLS simulates the network of selfish users without central authority to regulate the network. The distributed model is as follows. Each player in the network independently makes a sequence of attempts, using its own copy of the distributed algorithm depicted in Figure 5.2. We assume that the model runs in "continuous time"—that is, only one player is allowed to reallocate its

flow at each specific time. Then, this process generates a sequence of events in which a single player updates its strategy (changing the link to the route) which might change the cost that other players experience. We assume that the costs are updated without delay.

Remark 5.3.1. Because all of the players are using the distributed algorithm, they behave similarly. Note that the flow of players does not effect on their behaviours. Players only interact through the effects they have on the latency of the links. If a player selects the minimal-latency link rather than a random link (as the standard "best response"), then the process of selecting a link would become a $\Theta(m)$ operations (recall that there are m links in a model). Moreover, as mentioned above, we consider a continuous time model to avoid the issue of a link's latency being changed (by a different player) in between being tested and being selected.

In the following theorem borrowed from Theorem 1.1 by Goldberg's [Gol04], we show that the distributed algorithm is a distributed version of the RLS algorithm.

Theorem 5.3.2. *[Gol04] If we are given a network of n parallel links and m players, then*

- *Let $\mathbf{f}$ be an initial flow, and $\mathbf{f}^{NE}$ be the Nash flow with respect to $\mathbf{f}$. If $\mathbf{f}^{NE}$ is reached with the probability $p(\mathbf{f}^{NE})$ by the distributed algorithm then $p(\mathbf{f}^{NE})$ is also the probability that the RLS algorithm will find $\mathbf{f}^{NE}$.*
- *Let t be the expected number of attempts taken for the RLS algorithm to find Nash equilibrium. Then, the expected time taken for the distributed algorithm to reach Nash equilibrium is t/m.*

Proof. First, we prove the first part of the theorem. In the distributed algorithm, the significance of the exponential distribution used in the distributed algorithm is that it is "forgetful" in the sense that if the exponential distribution $P(t) = \exp(-t)$ then, for any positive t_0:

$$P(t \mid t \geq t_0) = \exp(t - t_0) \qquad for\ t \geq t_0.$$

As a result, at any point in time, if all of the players use the distributed algorithm then the next player to make an attempted move will be selected uniformly at random, regardless of previous events.

For $\lambda_0(\mathbf{f})$ and $\lambda_1(\mathbf{f})$, let $p(\lambda_0, \lambda_1)$ be the probability of using the distributed algorithm on λ_0 such that, after a move, λ_0 is replaced by λ_1. Similarly, let $p'(\lambda_0, \lambda_1)$ be the probability of using the RLS algorithm on λ_1 such that after a move, λ_0 is replaced by λ_1.

Given any initial flow λ_0, if we use the distributed algorithm, then the first player is selected from the uniform distribution, and then that player selects a link from a uniform distribution. This is similar to RLS; consequently, for any λ_0 and λ_1, we have $p(\lambda_0, \lambda_1) = p'(\lambda_0, \lambda_1)$.

For flow λ_h, let $D_\alpha(\lambda_h)$ and $D'_\alpha(\lambda_h)$ be the probability distribution over flows after α attempts by the distributed algorithm and the RLS algorithm respectively. Hence for a positive number $k < h$, we have $D_{\alpha+1}(\lambda_h) = \sum_{\lambda_k} D_\alpha(\lambda_k)p(\lambda_k, \lambda_h)$, and $D'_{\alpha+1}(\lambda_h) = \sum_{\lambda_k} D'_\alpha(\lambda_k)p'(\lambda_k, \lambda_h)$. By induction, $D_\alpha(\lambda_h) = D'_\alpha(\lambda_h)$, we proved the first statement of the theorem.

Regarding the second statement of the theorem, the expected value of the exponential random variable δ in the distributed algorithm is 1. Hence, during a time interval of length t, the expected number of attempts moves made by a player using the distributed algorithm is t. Thus, the expected number of attempts made by m players is $m \cdot t$. This gives the ratio of m between its expected time and the expected number of steps in the RLS algorithm. □

5.4 A Quick Upper Bound for the Convergence Time

In this section, we show the first upper bound on the expected number of attempts that RLS uses to construct Nash equilibrium in a load-balancing game. The outline of the proof is as follows. First, we show in Lemma 5.4.1 that the potential value is lower if a player makes a move such that its individual cost is lower. Then, in Lemma 5.4.2, we show the upper bound of the potential of an initial flow. After that we prove a lower bound on the loss of potential caused by a single move. Combining the upper bound of the initial potential and the lower bound of the loss of potential, we prove the bound on the number of attempts for Nash convergence in Theorem 5.4.4.

As pointed out by Even-dar et al. [EDKM07], the following lemma shows the fact that the game is a "weighted potential game", as we described earlier in the Previous Works section.

Lemma 5.4.1. *[EDKM07] For a flow λ_h, if, at the h-th move, player i reallocates its flow f^i from link j to k, such that the cost of i decreases then the corresponding potential decreases, by a positive value of $2f^i\left(\ell_j(\lambda_h) - \ell_k(\lambda_{h+1})\right)$.*

Proof. Suppose that, at the h-th move, the given flow is changed from λ_h to λ_{h+1} as a result of player i rerouting f^i from j to k such that the cost of i is lower. From

Definition 5.2.3, the loss of potential from λ_h to λ_{h+1} is:

$$\begin{aligned}
P(\lambda_h) - P(\lambda_{h+1}) &= \sum_{j\in E} \frac{(f_j(\lambda_h))^2}{c_j} + \sum_{i\in M} \frac{(f^i)^2}{c_{S^i(\lambda_h)}} + P^*(\lambda_h) - \sum_{j\in E} \frac{(f_j(\lambda_{h+1}))^2}{c_j} \\
&\quad - \sum_{i\in M} \frac{(f^i)^2}{c_{S^i(\lambda_{h+1})}} - P^*(\lambda_{h+1}) \\
&= \frac{(f_j(\lambda_h))^2}{c_j} + \frac{(f_k(\lambda_h))^2}{c_k} + \frac{(f^i)^2}{c_j} - \frac{(f_j(\lambda_{h+1}))^2}{c_j} \\
&\quad - \frac{(f_k(\lambda_{h+1}))^2}{c_k} - \frac{(f^i)^2}{c_k} \qquad (5.4) \\
&= \frac{(f_j(\lambda_h))^2 - (f_j(\lambda_h) - f^i)^2 + (f^i)^2}{c_j} + \\
&\quad \frac{(f_k(\lambda_{h+1}) - f^i)^2 - (f_k(\lambda_{h+1}))^2 - (f^i)^2}{c_k} \qquad (5.5) \\
&= \frac{2f^i \cdot f_j(\lambda_h)}{c_j} - \frac{2f^i \cdot f_k(\lambda_{h+1})}{c_k} \\
&= 2f^i \Big(\frac{f_j(\lambda_h)}{c_j} - \frac{f_k(\lambda_{h+1})}{c_k} \Big) \\
&= 2f^i \left(\ell_j(\lambda_h) - \ell_k(\lambda_{h+1}) \right)
\end{aligned}$$

where (5.4) comes from the fact that only the loads of link j and k, and $f_j(\lambda_h)$ change due to the move, and $c_{S^i(\lambda_h)} = j$ and $c_{S^i(\lambda_{h+1})} = k$; (5.5) follows from the fact that $f_j(\lambda_{h+1}) = f_j(\lambda_h) - f^i$ and $f_k(\lambda_{h+1}) = f_k(\lambda_h) + f^i$.

Note that for player i to benefit from the transfer the latency difference between links j and k must be more than the latency gain that player i would add to link k. That is

$$\begin{aligned}
\frac{f_j(\lambda_h)}{c_j} - \frac{f_k(\lambda_h)}{c_k} &> \frac{f^i}{c_k} \\
\frac{f_j(\lambda_h)}{c_j} - \frac{f_k(\lambda_h)}{c_k} - \frac{f^i}{c_k} &> 0 \\
\ell_j(\lambda_h) - \ell_k(\lambda_{h+1}) &> 0
\end{aligned}$$

which shows that the corresponding loss of potential is positive. □

Lemma 5.4.2. *For a parallel-link network model of m players and n links, assume that each player i possesses a positive flow f^i in the range $\{1, \ldots, f_{\max}\}$ and each link has a positive capacity c_j in the range $\{1, \ldots, c_{\max}\}$. The potential of the model is bounded by $O((mf_{\max})^2)$.*

Proof. For a parallel-link network, the potential is maximised when every player controls

a flow of the $f_{\max}$ unit, and uses the same link whose capacity is minimal, i.e. $c_j = 1$. The potential of the network, from Definition 5.2.3, of that flow is

$$\begin{aligned} P &= \sum_{j \in E} \frac{(f_j(\lambda_h))^2}{c_j} + \sum_{i \in M} \frac{(f^i)^2}{c_{S^i(\lambda_h)}} - P^* \\ &\leq (m f_{\max})^2 + m(f_{\max})^2 - P^* \\ &\leq 2m^2 (f_{\max})^2, \end{aligned}$$

which gives us the stated result. □

Lemma 5.4.3. *For a parallel-link network model, assume that the link capacities and player flows are in the range* $\{1, \ldots, c_{\max}\}$ *and* $\{1, \ldots, f_{\max}\}$ *respectively. A move lowers the potential by at least* $2/(c_{\max})^2$.

Proof. From Lemma 5.4.1, after a move, the potential decreases by

$$\begin{aligned} 2f^i \left(\ell_j(\lambda_h) - \ell_k(\lambda_{h+1})\right) &= 2f^i \left(\frac{f_j(\lambda_h)}{c_j} - \frac{f_k(\lambda_{h+1})}{c_k} \right) \\ &= 2f^i \left(\frac{c_k f_j(\lambda_h) - c_j f_k(\lambda_{h+1})}{c_j c_k} \right) \end{aligned}$$

Since all player flows and link capacities are integers and the loss of potential must be positive, the expression $c_k f_j(\lambda_h) - c_j f_k(\lambda_{h+1})$ is an integer greater than or equal to one. Therefore, we have the above equation satisfying

$$2f^i \left(\frac{c_k f_j(\lambda_h) - c_j f_k(\lambda_{h+1})}{c_j c_k} \right) \geq \frac{2}{c_j c_k} \geq \frac{2}{(c_{\max})^2}.$$

□

Theorem 5.4.4. *For a parallel-link network of* n *links, whose capacities are in the range* $\{1, \ldots, c_{\max}\}$*, and* m *players whose flow are in the range* $\{1, \ldots, f_{\max}\}$*, the expected number of attempts that RLS takes to reach Nash equilibrium is at most* $O(m^3 n (f_{\max})^2 (c_{\max})^2)$.

Proof. From Lemma 5.4.2 the initial potential is bounded by $m^2(f_{\max})^2$, and, from Lemma 5.4.3, the loss of potential after a move is at least $2/(c_{\max})^2$. Hence, the number of moves required for Nash convergence is at most $m^2(f_{\max})^2(c_{\max})^2/2$.

Given m players and n links, the probability of RLS selecting an attempt that is a move is at least $1/mn$. Therefore, the expected number of attempts for Nash convergence by RLS is at most $m^3 n (f_{\max})^2 (c_{\max})^2/2$ which give us the stated result. □

5.5 A Better Upper Bound

In this section, through a more careful analysis, we provide a better upper bound for the number of attempts that RLS uses to construct Nash equilibrium. We still use the potential function from Definition 5.2.3.

Lemma 5.5.1. *For a parallel-link network with m players, each with a flow in the range $\{1, \ldots, f_{\max}\}$, and n links each with a capacity in the range $\{1, \ldots, c_{\max}\}$, if $P(\lambda_h) > 2\tau mnf_{\max} + m(f_{\max})^2$ where $\tau \in \mathbb{R}^+$ then there is at least one pair of links j and k, such that $\ell_j(\lambda_h) - \ell_k(\lambda_h) > \tau$.*

Proof. Given a flow λ_h, let us suppose that j' is a maximal latency link and k' is a minimal latency link. Suppose for a contradiction proof, that $P(\lambda_h) > 2\tau mnf_{\max} + m(f_{\max})^2$, but $\ell_{j'}(\lambda_h) - \ell_{k'}(\lambda_h) < \tau$. From Definition 5.2.3, the potential of λ_h is

$$\begin{aligned} P(\lambda_h) &= \sum_{j\in E} \frac{(f_j(\lambda_h))^2}{c_j} + \sum_{i\in M} \frac{(f^i)^2}{c_{S^i(\lambda_h)}} - P^* \\ &\leq \sum_{j\in E} \frac{(f_j)^2}{c_j} + \sum_{i\in M} \frac{(f^i)^2}{c_{S^i}} - \widetilde{P}, \end{aligned}$$

where we substitute a smaller real value $\widetilde{P}$, for P^*. Let $\widetilde{P} = \sum_{j\in E} (\widetilde{f}_j)^2/c_j + \sum_{i\in M} (f^i)^2/c_{\max}$ where for $j \in E$, $\widetilde{f}_j$ is chosen such that the latency of link each j with respect to $\widetilde{f}_j$ is equal to the latency of link k', i.e. the minimal latency. So we have

$$\begin{aligned} P(\lambda_h) &\leq \sum_{j\in E} \frac{(f_j)^2}{c_j} + \sum_{i\in M} \frac{(f^i)^2}{c_{S^i}} - \sum_{j\in E} \frac{(\widetilde{f}_j)^2}{c_j} - \sum_{i\in M} \frac{(f^i)^2}{c_{\max}} \\ &\leq \sum_{j\in E} \left(\frac{(f_j)^2}{c_j} - \frac{(\widetilde{f}_j)^2}{c_j} \right) + \sum_{i\in M} \left(\frac{(f^i)^2}{c_{S^i}} - \frac{(f^i)^2}{c_{\max}} \right) \\ &\leq \sum_{j\in E} \left(c_j(\ell_j)^2 - c_j(\widetilde{\ell}_j)^2 \right) + \sum_{i\in M} (f^i)^2, \end{aligned}$$

which comes from the fact that, for each link j, $\ell_j = f_j/c_j$. Then we have

$$\begin{aligned} P(\lambda_h) &\leq \sum_{j\in E} \left(c_j(\ell_j - \widetilde{\ell}_j)(\ell_j + \widetilde{\ell}_j) \right) + m(f_{\max})^2 \\ &\leq \sum (\ell_j - \widetilde{\ell}_j)(f_j + \widetilde{f}_j) + m(f_{\max})^2. \end{aligned}$$

Since a latency difference between a pair of links is at most τ, and a link can have

at most $mf_{\max}$ of flow, we have

$$\begin{aligned} P(\lambda_h) &\leq \sum_{j \in E} (\tau 2mf_{\max}) + m(f_{\max})^2 \\ &\leq 2\tau mnf_{\max} + m(f_{\max})^2. \end{aligned}$$

This contradicts the earlier assumption, and hence we prove the stated result. □

Observation 5.5.2. *For a parallel-link network of m players and n links, assume that each player controls a flow in the range $\{1, \ldots, f_{\max}\}$ and that each link has a capacity in the range $\{1, \ldots, c_{\max}\}$. If there exist a pair of links j and k such that $\ell_j - \ell_k > f_{\max}$, then every players on j can make a move to k.*

Proof. For $i \in M; S^i = j$, player i wants to make a move to link k if the move results in a reduction in its cost. That is $\ell_j(\lambda_h) > \ell_k(\lambda_h) + f^i/c_k$ which implies $\ell_j(\lambda_h) - \ell_k(\lambda_h) > f_{\max}$. □

Lemma 5.5.3. *For a parallel-link network of n links and m players, assume that the link capacities are in the range $\{1, \ldots, c_{\max}\}$ and player flows are in the range $\{1, \ldots, f_{\max}\}$. If a flow λ_h satisfies $P(\lambda_h) \geq 4mn(f_{\max})^2 + m(f_{\max})^2$, then the probability that an attempt by RLS on λ_h is a move is at least $1/(2nc_{\max}f_{\max})$.*

Proof. From Lemma 5.5.1, if $P(\lambda_h) \geq 4mn(f_{\max})^2 + m(f_{\max})^2$ then there exist links j' and k' such that $\ell_{j'}(\lambda_h) - \ell_{k'}(\lambda_h) > 2f_{\max}$. Let us suppose that j' is a maximal latency link and that k' is a minimal latency link. There are two cases to be considered; the first case is when at least half of all the links have a latency of more than $\ell_{j'} - f_{\max}$ and the second case is when fewer than half of the links have a latency of more than $\ell_{j'} - f_{\max}$.

Case 1 : There are at least $n/2$ links whose latencies are more than $\ell_{j'} - f_{\max}$ which, by Observation 5.5.2, means that there exists a move that every player can make on those links. The assumption also implies that at least half of the total cost of the players is on those links (if we pick n/2 links with the highest latency). That is, at least $(1/2)\sum_{i \in M} f^i/c_{S^i}$ of the total cost is on those links. Since $(1/2)\sum_{i \in M} f^i/c_{S^i} \geq 1/2\sum_{i \in M} f^i/c_{\max} \geq (1/2)(m/c_{\max})$, there are at least $m/(2c_{\max}f_{\max})$ players whose flows can be moved on those links. Hence, an attempt by RLS has a probability of $1/2c_{\max}f_{\max}$ of selecting a player whose flow can be moved, and a probability of $1/n$ of selecting link k' to which that player may move, so the probability of an attempt by RLS being a move is at least $1/2nc_{\max}f_{\max}$.

Case 2 : There are less than $n/2$ links whose latencies are more than $\ell_{j'} - f_{\max}$. Since j' is a maximal latency link, the latency on link j' is at least $(\sum_{i \in M} f^i / c_{S^i})/n \geq (\sum_{i \in M} f^i / c_{\max})/n \geq m/nc_{\max}$. Hence, there are at least $m/nc_{\max}f_{\max}$ players on link j' whose flows can be moved to links that have a latency of less than $\ell_{j'} - f_{\max}$ (by Observation 5.5.2). We have the probability of $1/nc_{\max}f_{\max}$ of selecting a player on link j' whose flow can be moved, and the probability of $1/2$ of selecting a link to which those players may move its flow. Hence, in this case, RLS has the probability of at least $1/(2nc_{\max}f_{\max})$ that an attempt will be a move.

Combining the results from Case 1 and Case 2, we proved the stated result. □

We are now prepared to prove the main theorem.

Theorem 5.5.4. *For a parallel-link network with m players and n links, assume that the player flows are in the range $\{1, \ldots, f_{\max}\}$ and that the link capacities are in the range $\{1, \ldots, c_{\max}\}$. The expected number of attempts taken by RLS to construct Nash equilibrium is at most $O\left(n(mc_{\max}f_{\max})^2(n + c_{\max}f_{\max})\right)$.*

Proof. Suppose we are given an initial flow $\mathbf{f}$. Let $\mathcal{S} = (\lambda_0, \lambda_1, \lambda_2, ..., \lambda_N)$ be a sequence of flows in which $\lambda_0 = \mathbf{f}$ and λ_{h+1}, for $h \geq 1$, is obtained by RLS making an attempt on λ_h. N is chosen such that λ_N is the first occurrence in $\mathcal{S}$ of Nash equilibrium. We are interested in the expected value of N, i.e. $E(N) = E(|\mathcal{S}|)$. We break $\mathcal{S}$ down into the concatenation of two subsequences, $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2$. $\mathcal{S}_1$ is a sequence of flows λ_h, such that $P(\lambda_h) > 4mn(f_{\max})^2 + m(f_{\max})^2$ and $\mathcal{S}_2$ is a sequence of flows λ_h, such that $4mn(f_{\max})^2 + m(f_{\max})^2 \geq P(\lambda_h)$. We obtain bounds on the expected lengths of these subsequences that add up to the bound on the expected length of $\mathcal{S}$.

1. Bounding the expected length of $\mathcal{S}_1$.

   Recall that the initial potential is bounded by $O((mf_{\max})^2)$.

   In this case, we have $P(\lambda_h) > 4mn(f_{\max})^2 + m(f_{\max})^2$ which, by using Lemma 5.5.3, we find that the probability of an attempt by RLS on λ_h being a move is at least $1/(2nc_{\max}f_{\max})$. By Lemma 5.4.3, the lower bound on the loss of potential by a move is at least $2/(c_{\max})^2$. So the expected loss of potential by an attempt is at least $1/nf_{\max}(c_{\max})^3$. Hence, the expected number of attempts of RLS to reduce the potential from $(mf_{\max})^2$ to $4mn(f_{\max})^2 + m(f_{\max})^2$ is at most

$$\frac{(mf_{\max})^2}{1/nf_{\max}(c_{\max})^3} \leq nm^2(c_{\max}f_{\max})^3.$$

2. Bounding the expected length of $\mathcal{S}_2$.

   We have $P(\lambda_h) \leq 4mn(f_{\max})^2 + m(f_{\max})^2$. In general, the probability that an attempt results in a move is at least $1/mn$ for any flow that is not at Nash equilibrium. From Lemma 5.4.3, any move lowers the potential by at least $2/c_{\max}^2$, so the expected loss of potential due to an attempt is at least $2/mnc_{\max}^2$. Hence the expected number of attempts during $\mathcal{S}_2$ before a Nash equilibrium is reached is at most

$$\frac{4mn(f_{\max})^2 + m(f_{\max})^2}{2/mnc_{\max}^2} \leq \frac{4(mnc_{max}f_{\max})^2 + m^2n(c_{max}f_{\max})^2}{2}.$$

Combining the results of the expected length of each of $\mathcal{S}_1$ and $\mathcal{S}_2$ we obtain a bound of $O(n(mc_{\max}f_{\max})^2(n + c_{\max}f_{\max}))$. Hence, an upper bound on the length of $\mathcal{S}$ is $O(n(mc_{\max}f_{\max})^2(n + c_{\max}f_{\max}))$. □

5.6 Conclusions

We have shown that RLS can be realised by a simple distributed network of players that act selfishly, have no central control and only interact via the effect they have on the latency functions. As Goldberg [Gol04] pointed out, that this property is important for an algorithm in the study of the "coordination ratio" for this particular situation (for example [BGGM06, CV02, FKK+02, KP99, GLM+05, MS07, RT02, CKV02, FGL+03]), which is typically discussed as being a price that is paid for not having any central control over the players.

We exhibit two upper bounds on the expected number of attempts that RLS needs to reach Nash equilibrium. Both bounds are polynomial in term of the number of players, the number of links, the maximum flow of players and the maximum capacity of the links. However, one of the bounds shows a slightly better bound in term of number of players, i.e. m^3 to m^2, even though it comes with prices in term of number of links, the maximum flow of players and the maximum capacity of the links, i.e. n to n^2, $f_{\max}^2$ to $f_{\max}^3$ and $f_{\max}^2$ to $f_{\max}^3$.

A few obvious questions have been raised by the work in this chapter. Firstly, what is the convergence rate bounds of RLS for unrestricted flow and capacity? Goldberg [Gol04] showed an upper bound for unrestricted flow, but with identical links. He proved the expected number of attempts of RLS to be polynomial in terms of the number of players and the number of links. Secondly, what is the lower bound of RLS? Goldberg [Gol04] showed that the lower bound for unrestricted flow in a network with identical links is $\Theta(m^2)$.

Another open problem that is in the same direction as this thesis is what is the convergence rate of RLS for splittable flow. RLS would be modified as follows. Initially, each player split its flow randomly among the available links. At each step, a source link and a player that has a flow on that link, and a destination link are randomly selected. The player moves its flow as much as it can to reduce its cost from the source link to its destination link.

Chapter 6

Conclusions and Discussion

In this chapter, we will summarise the results presented in this thesis. We also examine the significance of these results and try to put them into the context of related works. We will then answer the main questions stated in the introduction—namely, how much is the additional social cost due to selfish leader? And does the randomised local search converge quickly to Nash equilibrium? Finally, we suggest some open problems.

6.1 Summary of the Results

In Chapter 2, we demonstrated Pigou's example—a simple network of two nodes, two parallel links—that is used to demonstrate the additional social cost that could arise when there exists a selfish Stackelberg leader even with the simplest network. Not only that, but we also demonstrated the other network set-ups that the other related works have studied, such as a set-up where a benign leader with a UE follower, a malicious leader with an atomic follower.

In Chapter 3, we provided several more examples that demonstrate the lower bounds on the price of selfish Stackelberg leadership. We consider both symmetric networks and an asymmetric network. For symmetric networks, we provided examples of cases in which the latency functions are linear, quadratic, cubic and quartic in a two-parallel link network. For an asymmetric network, we considered a network of three links, one that each player privately uses, and one shared between the players. All the links have linear latency functions. In summary, we showed the price of SSL of 1.075, 1.091, 1.135 and 1.161 for the networks with linear, quadratic, cubic and quartic latency functions respectively. We obtained the price of SSL of 1.074 for the asymmetric network.

In Chapter 4, we presented our main results. For a parallel-link network with affine linear latency functions, we proved a constant upper bound for the price of SSL that is independent of the number of links in the network. In particular, we first obtained

a quick upper bound for the price of SSL of 2. Then, through a sightly more careful analysis, we obtained a better upper bound of 4/3, and, with a deeper analysis, we were able to show an upper bound for the price of less than 4/3. At the end of the chapter, we investigate the network with homogeneous linear latency functions. We proved that the price of SSL in this case is one and showed a unique Nash and SSL equilibria.

In Chapter 5, we considered the problem of selfish routing from another perspective. We focused more on the convergence time of an algorithm that construct Nash equilibria, rather than the quality of Nash equilibria. A simple algorithm called the Randomised Local Search is investigated since it can simulate selfish behaviour in unregulated congestion networks. We consider a more general setting than [Gol04] who studied a network of identical links. We showed the distributed version of the algorithm. The main results in this chapter are that the upper bounds of the number of attempts the RLS needs to achieve Nash convergence which is polynomial for both the number of players and the number of links. We first shoed the upper bound for $O(m^3 n(f_{\max})^2(c_{\max})^2)$. Then, through a more careful analysis, we showed another upper bound of $O\left(n(mc_{\max}f_{\max})^2(n + c_{\max}f_{\max})\right)$.

6.2 Discussion

We have answered some of the questions in the problem statement. We have bounds on the price of SSL for the two players case, which usually is a foundation on analysing any game, on a parallel-link network, which is one of the most studied in network routing problems. In particular, we have obtained a narrow bound on the price of SSL for linear latency functions. These results should be a stepping-stone towards a more detailed analysis of Stackelberg games.

Roughgarden showed in [RT02] that the price of anarchy for a network with infinitely many players is unbounded. We only study the price of SSL for two players, and it seems that the price of SSL has a constant bound even for a network with a general latency function. Moreover, we suspect that the price of SSL is 1, if there are infinitely many followers as Nash flow and SSL flow are identical. Perhaps the main question to ask is whether there is a more dramatic cost (perhaps depending on the size of the network) for the setting up of more general networks. We have obtained a lower bound for the price of SSL of a network with latency functions up to quartic. It would be interesting to see the price of SSL for more general latency functions.

Another interesting question worth investigation is whether a social cost of SSL flow can be less than Nash flow. It seems obvious for the two players case since a leader will at improve its cost in SSL flow, or at least will not do worse than Nash flow, on the other hand a follower receives a minimal cost in SSL flow if the leader plays with the

same strategy as in the Nash flow. This is unclear when there are more players.

One extension to this thesis is to consider more than one follower, or in an extreme case of infinite number of followers. That is, a leader chooses a strategy before the rest of the players simultaneously compete to optimise their cost. As Pigou's example suggests, it may be better for a leader to have an atomic follower than a UE follower. Thus, we can imagine that the price of SSL decreases as the number of followers increases. We should note that in the case where every players simultaneously routing its flow, the more players does not necessary mean the higher social cost in the Nash equilibrium as it was shown by Catoni and Pallottino [CP91].

With regard to the convergence rate of the randomised local search in load-balancing games, Nash equilibrium can be reached rapidly. One obvious extension along the line of the problems studied in this thesis is would be to consider the splittable flow. That is, can a variant of RLS reach Nash equilibrium quickly in this setting?

Another open problem would be to consider a unrestricted flow model. In the case of integer flow, we have a nice drop in potential of $2/c_{\max}^2$. For the network with identical links, but general latency functions, Goldberg showed an upper bound that is polynomial in terms of the number of players and the number of links.

Bibliography

[ABJS00] E. Altman, T. Başar, T. Jiménez, and N. Shimkin. Competitive routing in networks with polynomial cost. In *Proceedings of the 2000 IEEE Computer and Communications Societies Conference on Computer Communications*, 2000.

[ACY06] J. Aspnes, K. L. Chang, and A. Yampolskiy. Inoculation strategies for victims of viruses and the sum-of-squares partition problem. *Journal of Computer and System Sciences*, 72(6):1077–1093, 2006.

[BFG+07] P. Berenbrink, T. Friedetzky, L. A. Goldberg, P. W. Goldberg, Z. Hu, and R. Martin. Distributed selfish load balancing. *SIAM Journal on Computing*, 37(4):1163–1181, 2007.

[BGGM06] P. Berenbrink, L. A. Goldberg, P. W. Goldberg, and R. Martin. Utilitarian resource assignment. *Journal of Discrete Algorithms*, 4(4):567–587, 2006.

[BKP07] M. Babaioff, R. Kleinberg, and C. H. Papadimitriou. Congestion games with malicious players. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 103–112, New York, NY, USA, 2007. ACM.

[BMW55] M. Beckmann, C. B. McGuire, and C. B. Winsten. Studies in the economics of transportation. Technical report, U.S. AIR FORCE, 1955.

[CCM06] R. Cominetti, J. R. Correa, and Nicolás E. Stier Moses. Network games with atomic players. In *Annual International Colloquium on Automata, Languages and Programming*, pages 525–536, 2006.

[CKV02] A. Czumaj, P. Krysta, and B. Vöcking. Selfish traffic allocation for server farms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 287–296, 2002.

[CP91] S. Catoni and S. Pallottino. Traffic equilibrium paradoxes. *Transportation Science*, 24(3):240–244, 1991.

[CS07] J. R. Correa and N. E. Stier Moses. Stackelberg routing in atomic network games. Columbia Working Paper No. DRO-2007-03, February 2007.

[CSM05] J. R. Correa, A. S. Schulz, and N. E. Stier Moses. On the inefficiency of equilibria in congestion games. In *Proceedings of the 11th Integer Programming and Combinatorial Optimization Conference*, pages 167–181, 2005.

[CV02] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 413–420, 2002.

[DJW02] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276(1–2):51–81, 2002.

[EDKM07] E. Even-Dar, A. Kesselman, and Y. Mansour. Convergence time to nash equilibrium in load balancing. *ACM Transactions on Algorithms*, 3(2):32, 2007.

[ES90] A. A. Economides and J. A. Silvester. Priority load sharing: An approach using stackelberg games. In *Proceedings of the 28th Annual Allerton Conference on Communications, Control, and Computing*, pages 674–683, 1990.

[FGL+03] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proceedings Automata of the 30th International Colloquium, Languages and Programming*, 2003.

[FKK+02] D. Fotakis, S. C. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. G. Spirakis. The structure and complexity of nash equilibria for a selfish routing game. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, pages 123–134, 2002.

[Gib92] R. Gibbons. *Game Theory for Applied Economists*. Printon University Press, 1992.

[GLM+05] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. G. Spirakis. Structure and complexity of extreme nash equilibria. *Theoretical Computer Science*, 343:133–157, 2005.

[Gol04] P. W. Goldberg. Bounds for the convergence rate of randomized local search in a multiplayer load-balancing game. In *Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing*, pages 131–140, 2004.

[GP07] P. W. Goldberg and P. Polpinit. The price of selfish stackelberg leadership in a network game, 2007. arXiv.org:0711.1242.

[GW03] O. Giel and I. Wegener. Evolutionary algorithms and the maximum matching problem. In *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 415–426, 2003.

[Har85] P. T. Harker. Multiple equilibrium behaviors on networks. *Transportation Science*, 22(1):39–46, 1985.

[HM85] A. Haurie and P. Marcott. On the relationship between nash-cournot and wardrop equilibria. *Networks*, 15(3):295–308, 1985.

[HTW06] A. Hayrapetyan, É. Tardos, and T. Wexler. The effect of collusion in congestion games. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 89–98, 2006.

[KK07] G. Karakostas and S. G. Kolliopoulos. Stackelberg strategies for selfish routing in general multicommodity networks. *Algorithmica*, 2719:514–526, 2007.

[KLO97a] Y. A. Korilis, A. A. Lazar, and A. Orda. Achieving network optima using stackelberg routing strategies. *IEEE/ACM Transactions on Networking*, 5(1):161–173, 1997.

[KLO97b] Y. A. Korilis, A. A. Lazar, and A. Orda. Capacity allocation under noncooperative routing. *IEEE Transactions on Automatic Control*, 42(3):309–325, 1997.

[KM02] V. S. Anil Kumar and M. V. Marathe. Improved results for stackelberg scheduling strategies. In *Proceedings of the 29th Annual International Colloquium on Automata, Languages and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pages 776–787, 2002.

[KP99] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563 of *Lecture Notes in Computer Science*, pages 404–413, 1999.

[KPS05] A. C. Kaporis, E. Politopoulou, and P. G. Spirakis. The price of optimum in stackelberg games. *Electronic Colloquium on Computational Complexity*, 12(056), 2005.

[KS06] A.C. Kaporis and P.G. Spirakis. The price of optimum in stackelberg games on arbitrary single commodity networks and latency functions. In *Proceedings of the 18th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 19–28, 2006.

[MS96] D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

[MS01] M. Mavronicolas and P. G. Spirakis. The price of selfish routing. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 510–519, 2001.

[MS07] M. Mavronicolas and P. G. Spirakis. The price of selfish routing. *Algorithmica*, 48(1):91–126, 2007.

[MSW06] T. Moscibroda, S. Schmid, and R. Wattenhofer. When selfish meets evil: byzantine players in a virus inoculation game. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 35–44, New York, NY, USA, 2006. ACM.

[MT03] J. E. Marsden and A. J. Tromba. *Vector Calculus*. W.H. Freeman and Company, fifth edition, 2003.

[Nas51] J. F. Nash. Non-cooperative games. *Annals of Mathematics*, 2(54):286–295, 1951.

[NW99] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag New York, 1999.

[NW07] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 378(1):32–40, 2007.

[ORS93] A. Orda, R. Rom, and N. Shimkin. Competitive routing in multiuser communication networks. *IEEE/ACM Transactions on Networking*, 1(5):510–521, 1993.

[Osb94] M. J. Osborne. *A Course in Game Theory*. MIT Press, 1994.

[Pap01] C. H. Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 749–753, 2001.

[Pig20] A. C. Pigou. *The Economics of Welfare*. Macmillan, 1920.

[RN03] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2nd edition, 2003.

[Ros65] J. B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33(3):520–534, 1965.

[Ros73a] R. W. Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973.

[Ros73b] R. W. Rosenthal. The network equilibrium problem in integers. *Networks*, 3(1):53–59, 1973.

[Rou02] T. Roughgarden. The price of anarchy is independent of the network topology. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 428–437, 2002.

[Rou04] T. Roughgarden. Stackelberg scheduling strategies. *SIAM Journal on Computing*, 33(2):332–350, 2004.

[Rou05a] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. MIT Press, 2005.

[Rou05b] T. Roughgarden. Selfish routing with atomic players. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1184–1185, 2005.

[RT02] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236–259, 2002.

[Swa07] C. Swamy. The effectiveness of stackelberg strategies and tolls for network congestion games. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1133–1142, 2007.

[vM44] J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, 1st edition, 1944.

[von34] H. von Stackelberg. *Marktform und Gleichgewicht*. Springer, 1934.

[Weg03] I. Wegener. Towards a theory of randomized search heuristics. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science*, volume 2747 of *Lecture Notes in Computer Science*, pages 125–141, 2003.

[WW05] I. Wegener and C. Witt. On the optimization of monotone polynomials by simple randomized search heuristics. *Combinatorics, Probability & Computing*, 14(1-2):225–247, 2005.

[YZM07] H. Yang, X. Zhang, and Q. Meng. Stackelberg games and multiple equilibrium behaviors on networks. *Transportation Research Part B: Methodological*, 41(8):841–861, 2007.