Nash Equilibria
in
Discrete Routing Games

Dissertation

von

Manuel Rode

Schriftliche Arbeit zur Erlangung des Grades
eines Doktors der Naturwissenschaften

Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

Paderborn, im November 2004

Acknowledgment

I would like to thank all persons who supported my research during the past three years. First and foremost, let me mention my advisor, Prof. Dr. Burkhard Monien, who consistently gave me guidance in the process of scientific work and set a good example. It was him to encourage me to start this PhD project. He involved me in his way of thinking and the development of new ideas. His working group always provided a convenient research environment to me. I also thank him for bringing to me the opportunity of a graduate fellowship.
Prof. Dr. Marios Mavronicolas from the University of Cyprus got early into the recent research on selfish routing. Mutual visits and discussions contributed to the progress in my research as well. I thank him for giving good advice and for his great job on collaborative publications.
I thank my secondary advisors Prof. Dr. Friedhelm Meyer auf der Heide and Prof. Dr. Leena Suhl.
Besides Prof. Dr. Burkhard Monien and Prof. Dr. Marios Mavronicolas, Dr. Rainer Feldmann, Martin Gairing and Thomas Lücking are part of the "Nash team". I thank them for several helpful discussions and for the fruitful work on joint publications.
I am indebted to Dr. Robert Elsässer, Henning Meyerhenke and Dr. Ulf-Peter Schroeder for reading and commenting on parts of this work.
I thank all my colleagues for several discussions on scientific and non-scientific topics and for technical and administrative support. To name those I did not mention yet: Ulrich Ahlers, Bernard Bauer, Yvonne Bleischwitz, Dr. Torsten Fahle, Sven Grothklags, Georg Kliewer, Dr. Michael Laska, Dr. Ulf Lorenz, Dr. Tomas Plachetka, Marion Rohloff, Stefan Schamberger, Dr. Meinolf Sellmann, Dr. Norbert Sensen, Thomas Thissen, Karsten Tiemann, Andreas Woclaw and Alexander Znamenshchykov.
This work was supported by a fellowship of the International Graduate School *Dynamic Intelligent Systems* at the University of Paderborn. Thanks go to the Graduate School team.

Paderborn,
November 2004

Manuel Rode

Contents

— 1 —

Introduction

1.1 Selfishness

If a number of (rational) persons with diverse objectives act in a common environment, then each of them behaves as to maximize his/her own benefit.

This assumption is both, plausible and observable for a variety of scenarios like communication networks, computer networks, road networks, markets, societies and games.

Independent users of an environment usually have differently directed objectives. Since their actions interfere, they may prevent each other from reaching their goals – unintentionally but carelessly. Typically it is not possible for all participants to entirely achieve their target. But then, what is desirable? One possible answer is to declare the maximization of the average benefit as the social goal. Another common idea is to aim at maximizing the welfare of the users which are worst off, that is, the maximization of the minimal personal benefit among the participants is the social goal. As the users do not care about the social goal, the situation that appears may be socially undesirable. Two questions arise immediately:

- How bad can the situation get?
- What can be done about it?

1.2 Strategic Games

From the viewpoint of game theory we may consider settings like the above as strategic multi-player games. Such a game is described by a set of alternative actions (the *strategies*) and a *payoff function* for each player. A player's payoff may depend on the

collection of all players' choices (the *outcome* of the game). The players independently try to maximize their payoff. There is no cooperation among them. A fundamental concept for non-cooperative games is that of a *Nash equilibrium*. A Nash equilibrium is an outcome of the game such that no player can increase its benefit by unilaterally deviating from its current strategy. It is hence a stable state: It does not change, if the participants are allowed to renew their decisions. The most obtruding questions in the analysis of particular settings are:

- How, if at all, can a Nash equilibrium be computed?
- How far from optimum is the social welfare in a Nash equilibrium state?
- How can the users be forced to attain a socially desirable state, or at least to avoid socially bad states?

The ratio by which the social benefit in a Nash equilibrium lags behind optimum in the worst case due to lack of coordination is called *coordination ratio* or *price of anarchy* [47].

Instead of choosing exactly one strategy deterministically, users may rate their strategies by probability distributions. Such a probability distribution is hence called a *mixed strategy*. Since users behave selfish, we may expect them to choose their probabilities as to maximize their expected individual profit. The social goal is defined by the expectation of the social benefit.

1.3 Routing Games

Selfish routing in networks constitutes a large class of strategic games that draws attention from various domains. Many applications directly involve networks (e.g. communication networks or traffic networks). Moreover, networks can serve to model other kinds of resources. The main focus of this work lies on networks of parallel links. Aside from scenarios that involve this kind of networks, they purchase their interest as a means of modeling settings where multiple users share a set of self-contained resources (e.g. machine scheduling).

In the context of routing networks it is more intuitive to speak of cost instead of benefit. Cost is negated payoff. To maximize payoff, the cost has to be minimized. In networks we can think of cost in terms of latency. In correspondence to the last section, we distinguish between a user's *individual cost*, which is minimized by the user, and *social cost*, which measures how desirable a routing in the network is from a social point of view (the smaller the social cost, the more desirable). The strategies of a network user are its feasible routing paths. In a network of parallel links the strategies correspond to the links. Each user must send a certain amount of flow through the network. Usage of a link involves costs for the users in terms of latency. Associated with each link is a function, the *latency function*, that depends on the set of users occupying the link and gives the latency experienced by them. Thus we make two restrictive but reasonable assumptions:

- The latency on a link depends only on the set of users routing along this link.
- A link causes the same cost to all users occupying it.

The latency experienced by the user of a path is commonly defined as the sum or as the maximum, respectively, of latencies of its links. Of course, both definitions coincide for networks of parallel links.

A major characteristic of a routing model is whether a user's flow is splittable or not. We consider only unsplittable flows here which give rise to combinatorial problems, whereas arbitrarily divisible flows allow the application of continuous mathematics. To distinguish between both we may speak of discrete routing and continuous routing, respectively.

The set of *pure* (i.e., deterministic) routings in a discrete routing game is finite. If users choose mixed strategies, then the outcome of the routing game becomes randomized. We speak of a *mixed routing* or a *mixed assignment* in that case.

1.4 Related Work

The number of publications in the area of strategic games is vast and goes back to the beginning of the last century (and before). In this section we will only cite the work that is most relevant for us. For a general introduction to game theory see [60, 65, 66]. An overview of current issues and applications of game theory is given in [67].

Much of the research in game theory is based in some way on the leading concept of Nash equilibria. The existence of a mixed Nash equilibrium for any strategic game was proven in the seminal paper of Nash in 1951 [61][1]. The work of Nash entailed a series of other publications, many of them concerned with refinements of the equilibrium concept to particular areas.

The research on routing games was originally motivated by the investigation of traffic networks and transportation planning [83, 6, 7, 12, 20]. Here, the behavior of road users interacting in a traffic network is modeled in a selfish routing game. Each user is assumed to choose a path from its start point to its target node that minimizes its experienced latency. As the number of road users is assumed to be very high, the contribution of a single user to the total flow is infinitesimal. Equilibrium states of such flows are called *Wardrop equilibria*. See [3, 40] for a discussion of the relation between Wardrop and Nash equilibria. The social cost is commonly defined as the average latency per unit of flow in this context. We refer to this setting as *Wardrop model*. Recent work on the Wardrop model includes [64, 44, 71, 72, 76, 73, 74, 77, 16, 15, 17]. The infinitesimality of user flows makes this model basically different from the one with indivisible flows which we consider here. Nevertheless, we will apply some of the ideas of Roughgarden [74] to discrete routing.

In [12] Braess describes a counterintuitive phenomenon that can appear in traffic networks: Inserting an additional link into the network can increase the individual cost of

[1] Already in 1838 Cournot used a restricted version of a Nash equilibrium.

every user according to a Nash equilibrium flow. Thus, although an additional link should intuitively enhance the options for the users, the situation gets worse. This phenomenon is known as Braess paradox. It can analogously be observed in other domains like a mechanical network of springs and strings or an electronic circuit of resistors and Zener diodes [14]. A hydraulic and a thermal analogue is known as well. Unfortunately, it is in general very difficult to detect such undesirable situations [71].

In 1999 Koutsoupias and Papadimitriou [47] established a field of research we are concerned with in this thesis. They explore a selfish routing game on parallel links. The links process their traffic at different speeds. Users assign their indivisible flows (deterministically or non-deterministically, resulting in pure or mixed assignments, respectively) to the links. The amount of flow a user has to ship is given by the user's weight. As users behave selfish, they aim at minimizing their experienced latency or their expected latency in case of mixed assignments, respectively. The latency of a link is the sum of user weights that route along this link (the link's *congestion*) over its *speed*. The social cost of a pure assignment is the maximal user cost, i.e., the maximal experienced latency. In case of mixed assignments, the social cost is the expected maximal user cost. We refer to this model as the KP model. It is (for the deterministic case) equivalent to the classical machine scheduling problem with uniform machines. Here, machines correspond to links, jobs to user flows, and makespan to social cost. But whereas the scheduling problem is a global optimization problem (the goal is makespan minimization), the KP model involves only local optimizations (each users' goal is latency minimization). Due to lack of coordination among the users, the routing arising from selfish user decisions (that we expect to settle at a Nash equilibrium) can be very bad (in terms of social cost) compared to a globally optimal one. [47] is the first work to consider this model under this game theoretic light. Koutsoupias and Papadimitriou introduce the coordination ratio as a measure for the loss in social quality if users are uncoordinated. More precisely, they define it as the ratio of the worst social cost of a (mixed) Nash equilibrium over the optimal social cost. They give tight bounds on the coordination ratio for two parallel links and asymptotical bounds for more than two links. Their results have been generalized in different directions.

In [54] Mavronicolas and Spirakis introduce the notion of a *fully mixed Nash equilibrium* for the KP model. In such Nash equilibrium each probability is strictly positive. That is, each user occupies every link with some non-zero probability. They characterize the existence and non-existence of fully mixed equilibria in the KP model. Furthermore, they prove asymptotically tight bounds on the coordination ratio of fully mixed Nash equilibria. They were generalized by Czumaj and Vöcking in [19], who show that the coordination ratio for the KP model with m links is $\Theta(\frac{\log m}{\log\log\log m})$. For either identical links or pure Nash equilibria it becomes $\Theta(\frac{\log m}{\log\log m})$.

Due to Nash it is known that for any strategic game there exists a (mixed) Nash equilibrium. A pure Nash equilibrium does not necessarily exists. In [29] Fotakis et al. show that every instance of the KP model possesses a pure Nash equilibrium and present a greedy algorithm (known as LPT or Graham's algorithm) to compute it efficiently. On the other hand they show that computing the best or worst Nash equilibrium is NP-hard.

They also give rise to the *Fully Mixed Nash Equilibrium Conjecture* here which is explicitly stated in [35]. It says that for any instance of the KP model, the fully mixed Nash equilibrium has the worst social cost among all Nash equilibria, provided that it exists. Fotakis et al. [29] show that the conjecture holds for instances with two users and identical links. For general instances with identical users they show that the social cost of any Nash equilibrium is bounded by that of the generalized fully mixed Nash equilibrium multiplied by a constant. A generalized fully mixed Nash equilibrium is the fully mixed Nash equilibrium induced by the instance resulting from removing the smallest possible number of slowest links such that the fully mixed Nash equilibrium exists. Another result is a randomized FPTAS for the computation of the social cost of a Nash equilibrium on identical links. In [46] Koutsoupias et al. give an asymptotically tight bound on the coordination ratio of approximate Nash equilibria, a relaxation of ordinary Nash equilibria, for identical links. Gairing et al. [35] show the Fully Mixed Nash Equilibrium Conjecture for pure Nash equilibria, that is, the fully mixed Nash equilibrium, if it exits, is worse than any pure Nash equilibrium in terms of social cost. Furthermore, they show that the social cost of a fully mixed Nash equilibrium can be computed by a randomized PTAS, and that it approximates the worst social cost of any Nash equilibrium up to a factor of $(6+\epsilon)$, both for identical links. On the other hand it is NP-hard to approximate the worst social cost within a factor of $(2-\frac{2}{m+1}-\epsilon)$ for any $\epsilon > 0$.

While a Nash equilibrium is a static state that fulfills certain properties, it has to be determined by some additional policy how the dynamic process leading to a Nash equilibrium takes place. Even-Dar et al. [23] and Goldberg [37] investigate the convergence to a pure Nash equilibrium in the KP model. While [23] yields bounds on the worst case number of reassignment steps according to different policies, [37] addresses the expected number of moves until a Nash equilibrium is reached when users are selected by a random policy. An earlier work on this topic is that of Altman et al. [2], where the case of two links is considered, and that of Singh et al. [78], where the dynamics of mixed strategies for bimatrix games with two strategies are explored.

A natural direction of generalization concerns the link latency functions in the KP model. Recall that in the original KP model the latency on link j is $\frac{x_j}{s_j}$, where x_j is the total flow and s_j is the speed of link j. In correspondence to the machine scheduling terminology we speak of related links in this case. Instead, we can allow for more general latency functions. In [50] Libman and Orda explore a routing model with parallel links where the latency is an arbitrary non-increasing function of the residual capacity, where the latter is defined as difference of a links capacity, that is associated with every link, and the congestion of the link. They discuss existence, uniqueness and convergence issues. Czumaj, Krysta and Vöcking state bounds on coordination ratio and bicriteria results for fractional and integral flows on parallel links with monotone cost functions in [18]. In particular they explore $M/M/1$-queue latency functions which fit in the model of residual capacity functions from Libman and Orda.

Gairing et al. address the KP model with restricted links [33]. Here users may only route along a subset of the links. This set of allowed links is given explicitly for each user. For the case of equal link speeds they show how any feasible assignment can be

transformed into a Nash equilibrium without increasing the social cost[2]. In [79] Suri, Tóth and Zhou explore restricted links, too. They consider related and identical links but allow unweighted users, only. Social cost is defined as the sum of user latencies in their model. They bound the coordination ratio for pure Nash equilibria by 2.5 for related links, and by 2.15 in case of identical links, where a lower bound of 2.001 holds.

In [10] Berenbrink et al. consider the KP model with one modification. Here, social cost is defined as the average of individual costs. Several upper and lower bounds on the coordination ratio of pure Nash equilibria and on the ratio of worst and best Nash equilibrium are established.

Milchtaich explores in [56] a model which is equivalent to routing unweighted users on parallel links, but where each user has its own latency function for every link. He shows that for any instance of this model there exists a pure Nash equilibrium and constructs a sequence of greedy steps with polynomial length that leads to a Nash equilibrium. For weighted users and two links a pure Nash equilibrium can only exist in general for up to two users.

Another extension of the KP model is to allow arbitrary network topologies with individual source-sink pairs instead of parallel links only. If we make one step further in this direction, we come to what is called a *congestion game*. Here user strategies can be arbitrary subsets of the set of resources. The cost for using a resource is a function of its congestion. A user's individual cost is the sum of costs of resource the user occupies. A network congestion routing game is hence a special case of a congestion game for which a bijective mapping of the links to the resources exists, such that the users' source-sink paths correspond to their strategies in the congestion game. Rosenthal showed that any unweighted congestion game, i.e., a congestion game with unweighted users, has a pure Nash equilibrium [70]. Libman and Orda gave an example that proves this fact invalid for weighted users [50], even in network congestion games. Rosenthal's proof is based on a potential function. Such a function maps any assignment of users to their strategies on a real number. Any change of a single user's strategy that improves the user's individual cost decreases the value of the potential function. Monderer and Shapley introduced the notion of a potential game which is a game possessing a potential function [58]. They show that potential games have at least one pure Nash equilibrium. See [82, 81, 80] for further characterizations of potential games. Potential functions are a powerful tool for proving the existence of pure Nash equilibria. Recently, Fotakis et al. could show that any weighted network congestion game with linear link latency functions has a pure Nash equilibrium [30]. Their proof is based on a potential function. Furthermore, they show that the coordination ratio for pure Nash equilibria of unsplittable flow in layered networks where the link latency equals the link congestion is $\Theta(\frac{\log m}{\log \log m})$, where m is the number of links. Fabrikant et al. explore the complexity of pure Nash equilibria in congestion games [25]. Existence and uniqueness of pure Nash equilibria for strategic games in general are discussed in [69, 38]. A survey of methods to numerically compute mixed Nash equilibria can be found in [55].

In markets, customers and providers interact, each as to maximize his/her profit. This

[2] We will refer back to this process as *Nashification* later.

is another classical area of application for Nash equilibria. See [21, 8] for recent results.

In [24, 4], games are explored where players have to build up network connections or to establish communication paths, respectively. An obvious motivation for such settings is to model the evolvement of huge networks that are not centrally controlled, like the internet.

Another issue, as a consequence of selfish behavior, is that of *mechanism design*. How can environments be designed as to make users attain only desirable equilibrium states? [62] gives a survey on this topic. In [44, 49, 26, 71, 16, 15, 13] mechanism design for network routing games is discussed under various circumstances. In models that are explored in this context, users are often allowed to have *private information*. [63] and [5] consider the classical scheduling problem (which is basically equivalent to the KP model) under this aspect.

In [72, 48] *Stackelberg strategies* for routing infinitesimal users on parallel links are considered. In a Stackelberg strategy a certain fraction of the flow is controlled centrally in order to induce good Nash equilibrium states for the remaining users. Both publications define social cost as the average cost per unit of flow, i.e., as in the Wardrop model, and give algorithms to compute Stackelberg strategies with guaranteed performance as well as hardness results. In a similar flavor is the work of Korilis et al. [45] who explore Stackelberg strategies for parallel $M/M/1$-queues.

Market equilibria, mechanism design and Stackelberg routing are not a topic of this work. We do not intend to give an exhaustive overview on this area here but only a starting point for the interested reader.

1.5 Contribution

Chapters 3 and 4 are concerned with the original KP model. Recall that an instance of this model defines a set of weighted users and a set of related parallel links. Social cost is defined as the maximal individual latency experienced by any user, here. By Nashification we denote the process of bringing a pure assignment of users to links into a Nash equilibrium without increasing its social cost.

In Chapter 3 we give an efficient Nashification algorithm for the KP model. It uses at most $(m + 1)n$ (not necessarily improving) moves to convert any assignment into Nash equilibrium, where m is the number of links and n is the number of users. The algorithm runs in time $\mathcal{O}(nm)$. In combination with existing approximation algorithms for the well studied classical machine scheduling problem which is equivalent, it provides a method to compute approximate solutions for the NP hard problem of finding the best Nash equilibrium. In particular we get a PTAS for this problem. Prior to this work, only the LPT algorithm was known to compute a Nash equilibrium with guaranteed performance of $\frac{5}{3}$ (see [39, 31, 29]). If the order according to which users perform moves is not determined by some policy (like LPT), then we expect them to perform improving moves in an arbitrary fashion. We show that the worst case number of moves can be superpolynomial in this case, even if always the best possible move is made and even for identical links.

The *Nash Equilibrium Verification Problem* asks for the correct answer to the question

whether a given (pure) assignment is at Nash equilibrium or not. In Chapter 4 we consider this problem for the KP model. We determine its exact algebraic complexity to be $\Theta(n + m \log m)$. The upper bound is obtained from an algorithm that we introduce. A matching lower bound is derived from the theorem of Ben-Or by a chain of Turing reductions that involve some two-dimensional geometrical problems.

In Chapter 5 we turn our attention to two variants of the KP model with capacitated links. First, we explore it for links having the latency characteristics of $M/M/1$-queues. We prove that an existing PTAS for machine scheduling is a polynomial dual approximation scheme for routing identical $M/M/1$-queues. It is known that any assignment can be turned into a Nash equilibrium with a linear number of greedy steps, if their order is chosen appropriately. We show that there exist sequences of greedy steps with superpolynomial length. Second, we address the KP model with an additional weight constraint. That is, we allow users only to route on those links where their weight does not exceed the weight constraint, given by a single value associated with the link. Here, we observe that the LPT algorithm computes a Nash equilibrium. We describe how to modify the PTAS for machine scheduling as to obtain a PTAS for best Nash equilibrium computation in the KP model with weight constraints. At last we show that the Nashification algorithm introduced in Chapter 3 is capable of Nashifying any pure assignment in this model as well.

According to the KP model, the social cost is defined as the maximal individual latency. Hence, the overall quality of a routing is determined by the users that are worst off as opposed to the Wardrop model, where the social cost is computed as average latency per infinitesimal unit of flow. The latter definition incorporates all appearing latencies and weights them by the amount of flow subjected to them. Both concepts are commonly used. In Chapter 6 we apply the latter definition of social cost to the congestion routing model on parallel links with unweighted users. We prove a tight bound of $\frac{4}{3}$ for the coordination ratio of pure Nash equilibria on related links. For arbitrary latency functions and under very mild conditions, we prove a tight bound on the coordination ratio based on the anarchy value introduced in [74]. Furthermore, we give an improved algorithm for the computation of a Nash equilibrium with running time $\mathcal{O}(n + m \log m \log n)$. We show that an optimal assignment can be computed by the same algorithm with a minor modification. This result holds for instances with unweighted users and arbitrary non-decreasing latency functions. Then, we turn our attention to mixed Nash equilibria for this model. First, we show that the fully mixed Nash equilibrium is unique (for non-constant latency functions). It is known, that the fully mixed Nash equilibrium, if it exists, yields the largest social cost for instances with non-decreasing and convex latency functions. We show that this is not true for non-decreasing and non-convex latency functions, even for two identical links. Even the individual cost of a user can be larger according to a pure Nash equilibrium than according to the fully mixed Nash equilibrium. At last we characterize the instances for which the fully mixed Nash equilibrium exists or does not exist. This allows us to extend the Fully Mixed Nash Equilibrium Conjecture to instances for which the fully mixed Nash equilibrium does not exist according to its original definition.

In Chapter 7 we examine routing on unrelated parallel links. Here, a link's latency is the sum of cost values of the users occupying it. These cost values are explicitly given for

every link. Each user has its own vector of cost values. We show that the individual cost of any user, as well as the social cost, according to any pure Nash equilibrium is bounded by the corresponding value of the fully mixed Nash equilibrium, provided it exists and the number of users does not exceed the number of links. Furthermore, for two users, the individual cost according to any (mixed) Nash equilibrium is bounded by that of the fully mixed Nash equilibrium, if the latter exists. If additionally we have only two links, then the social cost of any Nash equilibrium is bounded by that of the fully mixed one, if it exists, that is, the Fully Mixed Nash Equilibrium Conjecture is valid. Finally, we give a negative result, namely that the Fully Mixed Nash Equilibrium Conjecture cannot be valid for more than two users on unrelated links.

The results of Chapters 3, 6 (except for Definition 6.10, Theorem 6.11 and Proposition 6.12) and 7 have been presented on international computer science conferences and are published in the conference proceedings (see the list of publications on page 105).

A previous version of the results of Chapter 3 appeared in [27]. Subsection 6.1.1 appeared in [51]. Parts of Subsection 6.1.2 (Definition 6.6 to Corollary 6.9), Subsection 6.1.3 and Section 6.2 appeared in [34]. Chapter 7 appeared in [52].

— 2 —

Preliminaries

Most of the definitions, denotations and terminologies used throughout this work are introduced in this chapter. Some definitions that are specific to only few particular topics will be stated when they are needed. We summarize all denotations given here in a table in Section2.6.

2.1 Basic Mathematical Definitions

The set of positive natural numbers is denoted $\mathbb{N} = \{1, 2, 3, \dots\}$, while $\mathbb{N}_0 = \{0, 1, 2, \dots\}$ is the set of non-negative natural numbers (including zero). For $i \in \mathbb{N}_0$ we denote as $[i] = \{1, \dots, i\}$ the set of the first i positive natural numbers. The closed interval $[a, b] = \{r \in \mathbb{R} \mid a \leq r \leq b\}$ is the set of reals non-smaller than a and non-larger than b. The corresponding open interval is defined as $(a, b) = \{r \in \mathbb{R} \mid a < r < b\}$. For a set M, 2^M denotes the power set of M. For a function F, $\mathcal{E}_{X \vdash Y}(F(X))$ is the expectation of $F(X)$ with random variable X distributed according to Y. For our purposes X will be an assignment $A : [n] \to [m]$, and Y a probability matrix $\mathbf{P} = (p_{ij}) \in [0, 1]^{n \times m}$. If $\tilde{A} : [n] \to [m]$ is a particular assignment, then the probability for A to attain $\tilde{A}$ is $\prod_{i \in [n]} p_{i\tilde{A}(i)}$. For a matrix $\mathbf{P} \in [0, 1]^{n \times m}$, a vector $\mathbf{p} = (p_1, \dots, p_m), \mathbf{p}^T \in [0, 1]^m$, and a natural number $i \in [n]$ we denote as $\mathbf{P} \mid_i \mathbf{p}$ the matrix resulting from $\mathbf{P}$, if the ith row of $\mathbf{P}$ is replaced by $\mathbf{p}$. For a natural number $j \in [m]$, $\mathbf{P} \mid_i j$ denotes the matrix where all entries of the ith row of $\mathbf{P}$ are set to zero except for the entry in the jth column which is set to 1. For a mapping $A : [n] \to [m]$ and natural numbers $i \in [n], j \in [m]$, $A \mid_i j$ denotes the mapping resulting from A if $A(i)$ is set to j. To avoid the excessive use of brackets, we abide with the following convention. The scope of a sum array ranges to the next addition or subtraction operator. The scope of a product array covers only the operand following

the product symbol. That is, $\sum \sum A \cdot B + C$ is to be read as $\sum (\sum (A \cdot B)) + C$, and $\prod A \cdot B$ as $(\prod A) \cdot B$.

2.2 Routing Models

In this work we consider routing networks of parallel links. The number of links is denoted as m. Non-cooperative users wish to send traffic from a common source to a common destination that are connected by the links. The number of users is denoted as n. We assume that links and users are numbered by $1, \ldots, m$ and $1, \ldots, n$, respectively, and identify the links and users by their indices. Hence we may denote the set of links as $[m]$ and the set of users as $[n]$. In this context $2^{[n]}$ refers to the set of all subsets of the users.

All users that route their traffic along a certain link $j \in [m]$ experience the same cost in terms of latency which is given by the link's latency function l_j.

2.2.1 Generic Model

In the most general model the latency function is of the form $l_j : 2^{[n]} \to \mathbb{R}_{\geq 0}$. That is, for any subset $U \subseteq [n]$ of users, $l_j(U)$ is the latency on link j if exactly the users in U route along link j. An instance of this *generic routing model* is given by $I = (\mathbf{l})$, where $\mathbf{l} = (l_1, \ldots, l_m)$ is the vector of latency functions.

2.2.2 Subclasses

We consider two different subclasses of this general model.

1. In the *unrelated routing model* the latency function on link $j \in [m]$ is defined by $l_j(U) = \sum_{i \in U} c_{ij}$, where $\mathbf{C} = (c_{ij})_{\substack{i \in [n] \\ j \in [m]}}$ is the cost matrix that defines for each user i and link j the contribution of user i to the latency on link j when user i routes its flow along link j. An instance of this model is denoted $I = (\mathbf{C})$.

2. The second subclass is the *congestion routing model*. Here the latency of any link j depends only on the total flow on the link. The users may contribute differently to the flow on the link they occupy which is reflected in their *weights*. The weight of user $i \in [n]$ is denoted w_i. Sometimes w_i is called the *flow* controlled by user i or simply the *traffic* of user i. We combine all user weights into the vector of user weights $\mathbf{w} = (w_1, \ldots, w_n)$. Furthermore, we denote as $W = \sum_{i \in [n]} w_i$ the total weight. If U is the subset of users that route along link j, then the total flow on link j is $\sum_{i \in U} w_i$. The total flow on a link is also called the link's *congestion*. Since in the congestion routing model the latency of any link j only depends on the total flow on that link, we may define the latency function of link j as $l_j : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. That is, if x is the congestion on link j, then the latency of link j is given by $l_j(x)$. An instance of the congestion routing model is denoted as $I = (\mathbf{w}, \mathbf{l})$. Further restricting the set of instances yields subclasses of the congestion routing model. The following subclasses will be of interest in this work.

a) The *congestion routing model with identical users*. Here all users control the same amount of flow. Without loss of generality we may assume this amount to be 1. An instance of this model is given by $I = (n, \mathbf{l})$. As a link's congestion can only attain integral values in this case, the latency functions have the form $l_j : \mathbb{N}_0 \to \mathbb{R}_{\geq 0}$.

b) The *congestion routing model with related links*. Here the latency function of link $j \in [m]$ is given by $l_j(x) = \frac{x}{s_j}$, where $s_j > 0$ is the *speed* of link j. We combine the speeds into the vector of link speeds $\mathbf{s} = (s_1, \ldots, s_m)$ and denote an instance of this model as $I = (\mathbf{w}, \mathbf{s})$. Note that this model can just as well be seen as a subcase of the unrelated routing model by defining the entries of the cost matrix as $c_{ij} = \frac{w_i}{s_j}$ for all $i \in [n], j \in [m]$. This model is also known as KP model.

We will consider two other models in Chapter 5, the *routing model with parallel $M/M/1$-links* and the *KP model with weight constraints*. An instance of the former is denoted as $I = (\mathbf{w}, \mathbf{c})$, an instance of the latter as $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$. They will be defined in the corresponding sections.

2.3 Pure Assignments

For an instance of any of the models, a *routing* is an assignment of users to links, denoted A. That is, A is a mapping of the form $A : [n] \to [m]$ that gives for each user $i \in [n]$ the link $A(i)$ where the user's traffic is routed. For distinction, we call this kind of assignment *pure assignment*, as opposed to mixed assignments that will be introduced in the next section.

Given a particular routing A for an instance I of the congestion routing model (or one of its subcases) with user weight vector $\mathbf{w}$, we can compute the link congestions. For any link $j \in [m]$ the congestion caused by routing A is $x_j(A) = \sum_{\substack{i \in [n] \\ A(i)=j}} w_i$. The *individual cost* or *user cost* of user i according to routing A for instance I, denoted $\lambda_i(A, I)$, is the latency on the link to which user i is assigned by A. Thus, $\lambda_i(A, I) = l_{A(i)}(x_j(A))$ for the congestion routing model. For an instance I of the unrelated routing model the individual cost of user i is $\lambda_i(A, I) = \sum_{\substack{k \in [n] \\ A(k)=A(i)}} c_{ij}$.

For a (partial) pure assignment $A : U \to [m]$, with $U \subseteq [n]$, we define $view(j, A) = \{i \in U \mid A(i) = j\}$ for $j \in [m]$. Note that U is given implicitly by the definition of A.

Sometimes we will need to express the individual cost of a particular user i when all users except for user i itself are assigned according to a given assignment A and user i goes to a given link j. We may denote the individual cost for user i as $\lambda_i(A\,|_i\ j, I)$ in this case. Thus, for an instance I of the congestion routing model $\lambda_i(A\,|_i\, j, I) = l_j(w_i + \sum_{\substack{k \in [n] \setminus \{i\} \\ A(k)=j}} w_k)$, and for an instance I of the unrelated routing model $\lambda_i(A\,|_i\, j, I) = c_{ij} + \sum_{\substack{k \in [n] \setminus \{i\} \\ A(k)=j}} c_{kj}$.

The overall quality of a routing A for an instance I is measured by the *social cost*, denoted $SC(A, I)$, as opposed to the individual cost.

The following three definitions of social cost are common. We use superscripts to explicitly distinguish between them.

1. $SC^{SUM}(I, A) = \sum_{i\in[n]} \lambda_i(A, I)$. In this case social cost is proportional (by factor n) to the average individual cost where all users contribute in equal measure regardless of their weights.

2. $SC^{MAX}(I, A) = \max_{i\in[n]} \lambda_i(A, I)$. This definition focuses on the user for which routing is most costly.

3. $SC^{AVG}(I, A) = \sum_{i\in[n]} w_i\lambda_i(A, I)$. This definition is used for the congestion routing model (it requires user weights). Here social cost is the weighted (by user weights) sum of individual costs. Put in other words, it is proportional (by factor W) to the average cost per unit of flow (the average cost of one flow unit is $\sum_{i\in[n]} w_i\lambda_i(A, I)/W$).

The results of this work involve the last two definitions of social cost. Note that in case of identical users the third definition coincides with the first definition.

2.4 Mixed Assignments

Under the light of non-cooperative game theory, a routing A, as introduced in the last section, is a collection of user decisions. The set of links corresponds to the set of allowed choices (or strategies) for any user. As long as the user decisions are deterministic, a state of this "routing game" is completely described by A. The general definition of a strategic game allows for non-deterministic user choices. That is, users may rate their strategies – here the links – by a probability distribution rather than choosing exactly one of them. We combine all those probabilities into a matrix $\mathbf{P} = (p_{ij})$ which we call a *mixed assignment*. That is, the flow of user i is routed via link j with probability p_{ij} for any $i \in [n], j \in [m]$. Thus, $p_{ij} \in [0, 1]$ for all $i \in [n], j \in [m]$ and $\sum_{j\in[m]} p_{ij} = 1$ for all $i \in [n]$.

We define link congestion, link latency, individual cost and social cost according to a mixed assignment $\mathbf{P}$ as the expectation of the respective value for pure assignment A, if A is probability distributed according to $\mathbf{P}$. That is, $x_j(\mathbf{P}) = \mathcal{E}_{A\vdash\mathbf{P}}(x_j(A))$ is the (expected) flow or congestion on link j according to $\mathbf{P}$ for $j \in [m]$, where $A \vdash \mathbf{P}$ means that the expectation is over all assignments A distributed randomly according to $\mathbf{P}$.
In the same way we get $l_j(\mathbf{P}, I) = \mathcal{E}_{A\vdash\mathbf{P}}(l_j(A, I))$ for the link latency on link $j \in [m]$, $\lambda_i(\mathbf{P}) = \mathcal{E}_{A\vdash\mathbf{P}}(\lambda_j(A))$ for the individual cost of user $i \in [n]$ and $SC(\mathbf{P}, I) = \mathcal{E}_{A\vdash\mathbf{P}}(SC(A, I))$ for the social cost. Note that the last statement applies to all three definitions of social cost. That is, $SC^{SUM}(\mathbf{P}, I) = \mathcal{E}_{A\vdash\mathbf{P}}(SC^{SUM}(A, I))$, $SC^{MAX}(\mathbf{P}, I) = \mathcal{E}_{A\vdash\mathbf{P}}(SC^{MAX}(A, I))$ and $SC^{AVG}(\mathbf{P}, I) = \mathcal{E}_{A\vdash\mathbf{P}}(SC^{AVG}(A, I))$.

Similarly to the deterministic case, $\lambda_i(\mathbf{P}\,|_i\,\mathbf{p}) = \mathcal{E}_{A\vdash(\mathbf{P}\,|_i\,\mathbf{p})}(\lambda_j(A))$ is the (expected) individual cost of user i according to a mixed assignment that is given by $(\mathbf{P}\,|_i\,\mathbf{p})$, the matrix where the ith row of $\mathbf{P}$ is replaced by $\mathbf{p} = (p_1, \ldots, p_m)$, a probability distribution over m links, that is, $p_j \in [0, 1]$ for all $j \in [m]$ and $\sum_{j\in[m]} p_j = 1$. Furthermore,

$\lambda_i(\mathbf{P} \mid_i j) = \mathcal{E}_{A \vdash (\mathbf{P} \mid_i j)}(\lambda_j(A))$ is the (expected) individual cost of user i according to $\mathbf{P}$, but with user i going to link j for sure, i.e., $\mathbf{P} \mid_i j$ is the matrix resulting from $\mathbf{P}$ if the entries of the ith row are all set to zero except for the jth entry which is set to one.

For any mixed assignment $\mathbf{P}$ we define the *support* of user $i \in [n]$ by $support(i, \mathbf{P}) = \{j \in [m] \mid p_{ij} > 0\}$ (the set of links chosen by user i with positive probability) and the *view* of link $j \in [m]$ by $view(j, \mathbf{P}) = \{i \in [n] \mid p_{ij} > 0\}$ (the set of users that choose link j with positive probability).

We call a mixed assignment $\mathbf{P}$ *fully mixed*, if it describes a routing where each user chooses every link with positive probability. That is, $support(i, \mathbf{P}) = [m]$ for all users $i \in [n]$ in a fully mixed assignment.

Note that a pure assignment A is just a special case of a mixed assignment $\mathbf{P}$, where for all $i \in [n], j \in [m]$, $p_{ij} = 1$ if $A(i) = j$, and $p_{ij} = 0$ if $A(i) \neq j$. Hence, every pure assignment can be described by a probability matrix with entries one and zero. But it will be more convenient to characterize a pure assignment by a mapping of users to links.

A fully mixed assignment and a pure assignment are hence two oppositional extreme formations of a mixed assignment.

2.5 Optimum, Nash equilibrium and Coordination Ratio

For an instance I of a routing model, $OPT(I) = \min_{A:[n]\to[m]} SC(A, I)$ is the *optimal social cost*. We use $OPT^{MAX}(I)$ or $OPT^{AVG}(I)$ to explicitly refer to one of the definitions of social cost, if the reference is not clear from the context. Note that the minimization is over all pure assignments, since there always exists a pure assignment with minimal social cost among all (mixed) assignments. The next definition formalizes this property.

In general, an assignment is at *Nash equilibrium*, if no user can decrease its individual cost by unilaterally changing its strategy.

Definition 2.1. *A mixed assignment $\mathbf{P}$ for an instance I of a routing model is at Nash equilibrium, if and only if for all users $i \in [n]$ and all probability distributions $\mathbf{p}$, that is $\mathbf{p} \in [0,1]^m$ and $\sum_{j\in[m]} p_j = 1$,*

$$\lambda_i(\mathbf{P}, I) \leq \lambda_i(\mathbf{P} \mid_i \mathbf{p}, I)\,.$$

Sometimes we simply say that $\mathbf{P}$ is a Nash equilibrium to express that $\mathbf{P}$ is an assignment at Nash equilibrium. Note that Definition 2.1 applies to fully mixed assignments and pure assignments as well, since both are particular mixed assignments. To explicitly refer to a pure assignment A that is at Nash equilibrium, we call A a pure Nash equilibrium. Similar, we call $\mathbf{F}$ a fully mixed Nash equilibrium, to express that $\mathbf{F}$ is a fully mixed assignment at Nash equilibrium.

We denote as $\mathcal{N}(I) = \{A : [n] \to [m] \mid A \text{ is at Nash equilibrium}\}$ the set of pure Nash equilibrium assignments for instance I, and as $\mathcal{M}(I) = \{\mathbf{P} \in [0,1]^{n\times m} \mid \forall i \in [n] : \sum_{j\in[m]} p_{ij} = 1, \text{ and } \mathbf{P} \text{ is at Nash equilibrium}\}$ the set of (mixed) Nash equilibria for instance I.

The *coordination ratio* (also called *price of anarchy*) gives the loss in quality (in terms of social cost) of a routing due to lack of coordination. For an instance I of a routing model, we define the coordination ratio $CR(I)$ of I as $CR(I) = \sup_{\mathbf{P} \in \mathcal{M}(I)} \frac{SC(\mathbf{P},I)}{OPT(I)}$. The coordination ratio of a routing model is the supremum of the coordination ratios of all its instances, that is, if S is the set of all instances of the routing model under consideration, then $\sup_{I \in S} CR(I)$ is the coordination ratio of the routing model. Many a time we restrict our considerations only to pure Nash equilibria. In that case coordination ratio refers to the *pure coordination ratio* $PCR(I) = \max_{A \in \mathcal{N}(I)} \frac{SC(A,I)}{OPT(I)}$ when applied to a single instance. The pure coordination ratio of a routing model is again the supremum of $PCR(I)$ among all instances I of the model.

2.6 Summary of Notations

The following list summarizes all our denotations introduced so far.

Denotation	Definition	Description
m	$\in \mathbb{N}$	number of links
n	$\in \mathbb{N}$	number of users
$[m]$	$= \{1, \ldots, m\}$	set of links
$[n]$	$= \{1, \ldots, n\}$	set of users
$2^{[n]}$	$= \{U \subseteq [n]\}$	set of all user subsets
w_i	$\in \mathbb{R}_{\geq 0}$	weight or traffic or flow of user i
$\mathbf{w}$	$= (w_1, \ldots, w_m)$	vector of user flows
W	$= \sum_{i \in [n]} w_i$	total weight or total flow
l_j		latency function of link j in the
	a) : $2^{[n]} \to \mathbb{R}_{\geq 0}$	a) general routing model
	b) : $\mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$	b) congestion routing model
	c) : $\mathbb{N}_{\geq 0} \to \mathbb{R}_{\geq 0}$	c) congestion routing model with identical users
s_j	$\in \mathbb{R}_{>0}$	speed of link j (for related links)
$\mathbf{s}$	$= (s_1, \ldots, s_m)$	vector of link speeds
$\mathbf{l}$	$= (l_1, \ldots, l_m)$	vector of link latency functions
$l_j(x)$	$= \frac{x}{s_j}$	latency function j (for related links)
c_{ij}	$\in \mathbb{R}_{\geq 0}$	cost contribution of user i on link j
$\mathbf{C}$	$\in \mathbb{R}_{\geq 0}^{n \times m}$	cost matrix (unrelated routing model)
$l_j(U)$	$= \sum_{i \in U} c_{ij}$	latency function j (for unrelated links)
I		instance of the
	a) $= (\mathbf{l})$	a) generic routing model
	b) $= (\mathbf{w}, \mathbf{l})$	b) congestion routing model
	c) $= (n, \mathbf{l})$	c) congestion routing model with identical users
	d) $= (\mathbf{w}, \mathbf{s})$	d) congestion routing model with related links
	e) $= (\mathbf{C})$	e) unrelated routing model
	f) $= (\mathbf{w}, \mathbf{c})$	f) routing model with $M/M/1$-links
	g) $= (\mathbf{w}, \mathbf{s}, \mathbf{u})$	g) KP model with weight constraints

Denotation	Definition	Description
p_{ij}	$\in [0,1]$	probability for user i to go on link j
$\mathbf{P}$	$= (p_{ij})_{i,j} \in [0,1]^{n \times m}$	mixed assignment
A	$: [n] \to [m]$	pure assignment
$x_j(A,I)$	$= \sum_{A(i)=j} w_i$	congestion on link j according to A
$x_j(\mathbf{P},I)$	$= \mathcal{E}_{A \vdash \mathbf{P}}(x_j(A,I))$	(expected) congestion on link j according to $\mathbf{P}$
$\lambda_i(A,I)$		individual cost of user i according to A
$\lambda_i(\mathbf{P},I)$	$= \mathcal{E}_{A \vdash P}(\lambda_j(A,I))$	(expected) individual cost of user i
$\lambda_i(A \mid_i j, I)$		individual cost of user i, if user i goes to link j and all other users are assigned according to A
$\lambda_i(\mathbf{P} \mid_i \mathbf{p}, I)$	$= \mathcal{E}_{A \vdash (\mathbf{P} \mid \mathbf{p})}(\lambda_j(A,I))$	(expected) individual cost of user i, if the probability distribution of user i is $\mathbf{p}$ and that of all other users is given by $\mathbf{P}$
$\lambda_i(\mathbf{P} \mid_i j, I)$		(expected) individual cost of user i, if user i goes to link j (for sure) and the probability distributions of all other users are given by $\mathbf{P}$
$support(i,\mathbf{P})$	$= \{j \in [m] \mid p_{ij} > 0\}$	supported links of user i
$view(j,\mathbf{P})$	$= \{i \in [n] \mid p_{ij} > 0\}$	view of link j
$view(j,A)$	$= \{i \in [n] \mid A(i) = j\}$	set of users that choose link j
$SC^{SUM}(A,I)$	$= \sum_i \lambda_i(A,I)$	social cost of A (1st definition)
$SC^{MAX}(A,I)$	$= \max_i \lambda_i(A,I)$	social cost of A (2nd definition)
$SC^{AVG}(A,I)$	$= \sum_i w_i \lambda_i(A,I)$	social cost of A (3rd definition)
$SC^{XXX}(\mathbf{P},I)$	$= \mathcal{E}_{A \vdash \mathbf{P}}(SC^{XXX}(A,I)$	social cost of (mixed) assignment $\mathbf{P}$, where $XXX \in \{SUM, MAX, AVG\}$
$OPT(I)$	$= \min_{A:[n]\to[m]} SC(A,I)$	optimal social cost according to respective definition of social cost
$\mathcal{N}(I)$		set of pure Nash equilibria for I
$\mathcal{M}(I)$		set of mixed Nash equilibria for I
$CR(I)$	$= \max_{\mathbf{P} \in \mathcal{M}(I)} \frac{SC(\mathbf{P},I)}{OPT(I)}$	coordination ratio of instance I
$PCR(I)$	$= \max_{A \in \mathcal{N}(I)} \frac{SC(A,I)}{OPT(I)}$	pure coordination ratio of instance I

2.7 Relations and Expressions

Many of the measures defined above can be expressed in several ways. It is often convenient to start from some particular expression in oder to simplify the calculations. We will now have a closer look to the different routing models and concretize the respective definitions.

2.7.1 The Generic Routing Model

2.7.1.1 Individual Cost

We start with a simple but useful lemma on the individual cost in the generic routing model.

Lemma 2.2. *Let $\mathbf{P} \in \mathbb{R}^{n \times m}$ be a mixed assignment for an instance $I = (\mathbf{l})$ of the generic routing model. Then, the individual cost of user $i \in [n]$ according to $\mathbf{P}$ is*

$$\lambda_i(\mathbf{P}, I) = \sum_{j \in [m]} p_{ij} \lambda_i(\mathbf{P} \mid_i j, I) \,.$$

Proof. It is

$$\begin{aligned}
\lambda_i(\mathbf{P}, I) &= \mathcal{E}_{A \vdash \mathbf{P}}(\lambda_i(A, I)) \\
&= \sum_{A:[n]\to[m]} \prod_{k \in [n]} p_{kA(k)} \lambda_i(A, I) \\
&= \sum_{j \in [m]} \sum_{A:[n]\setminus\{i\}\to[m]} p_{ij} \prod_{k \in [n]\setminus\{i\}} p_{kA(k)} l_j(view(j, A) \cup \{i\}) \\
&= \sum_{j \in [m]} p_{ij} \sum_{A:[n]\setminus\{i\}\to[m]} \prod_{k \in [n]\setminus\{i\}} p_{kA(k)} \lambda_i(A \mid_i j, I) \\
&= \sum_{j \in [m]} p_{ij} \mathcal{E}_{A \vdash \mathbf{P} \mid_i j}(\lambda_i(A, I)) \\
&= \sum_{j \in [m]} p_{ij} \lambda_i(\mathbf{P} \mid_i j, I) \,.
\end{aligned}$$

□

The following lemma gives an alternative way to express a user's individual cost according to a mixed assignment. We will use it in Chapter 6 for instances of the congestion routing model.

Lemma 2.3. *Let $\mathbf{P}$ be a mixed assignment for an instance $I = (\mathbf{l})$ of the generic routing model. Then the expected individual cost for user $i \in [n]$ when going to link $j \in [m]$ is*

$$\lambda_i(\mathbf{P} \mid_i j) = \sum_{U \subseteq [n]\setminus\{i\}} \prod_{k \in U} p_{kj} \prod_{k \in [n]\setminus(U \cup \{i\})} (1 - p_{kj}) \cdot l_j(U \cup \{i\}) \,.$$

Proof. We first prove the following.
Claim: For any $V \subseteq [n]\setminus\{i\}$

$$\begin{aligned}
\lambda_i(\mathbf{P} \mid_i j) = \sum_{A:\bar{V}\to[m]} \Big[\prod_{k \in \bar{V}} p_{kA(k)} \sum_{U \subseteq V} \Big(\prod_{k \in U} p_{kj} \prod_{k \in V\setminus U)} (1 - p_{kj}) \\
l_j(\{k \in \bar{V} \mid A(k) = j\} \cup U \cup \{i\}) \Big) \Big] \,,
\end{aligned}$$

where $\bar{V} = [n]\backslash\{i\}\backslash V$ is the inverse set of V.
Proof: We have

$$\lambda_i(\mathbf{P}\,|_i\,j) = \mathcal{E}_{A\vdash\mathbf{P}\,|_i\,j}(\lambda_i(A,I)) = \sum_{A:[n]\backslash\{i\}\to[m]} \prod_{k\in[n]\backslash\{i\}} p_{kA(k)} l_j(\{k\in[n]\backslash\{i\} \mid A(k)=j\}\cup\{i\})$$

which yields the claim for $V = \emptyset$. Now assume the claim is true for some $V \subsetneq [n]\backslash\{i\}$. Let $r \in \bar{V}$ be arbitrary. Then,

$$\begin{aligned}
\lambda_i(\mathbf{P}\,|_i\,j) = &\sum_{A:\bar{V}\to[m]} \Big[\prod_{k\in\bar{V}} p_{kA(k)} \sum_{U\subseteq V}\Big(\prod_{k\in U} p_{kj} \prod_{k\in V\backslash U}(1-p_{kj}) \\
&\quad l_j(\{k\in\bar{V} \mid A(k)=j\}\cup U\cup\{i\})\Big)\Big] \\
= p_{rj} &\sum_{A:\bar{V}\backslash\{r\}\to[m]} \Big[\prod_{k\in\bar{V}\backslash\{r\}} p_{kA(k)} \sum_{U\subseteq V}\Big(\prod_{k\in U} p_{kj} \prod_{k\in V\backslash U}(1-p_{kj}) \\
&\quad l_j(\{k\in\bar{V}\backslash\{r\} \mid A(k)=j\}\cup U\cup\{i,r\})\Big)\Big] \\
&+ \sum_{t\in[m]\backslash\{j\}} \Big[p_{rt} \sum_{A:\bar{V}\backslash\{r\}\to[m]} \prod_{k\in\bar{V}\backslash\{r\}} p_{kA(k)} \sum_{U\subseteq V}\Big(\prod_{k\in U} p_{kj} \\
&\quad \prod_{k\in V\backslash U}(1-p_{kj}) l_j(\{k\in\bar{V}\backslash\{r\} \mid A(k)=j\}\cup U\cup\{i\})\Big)\Big] \\
= &\sum_{A:\bar{V}\backslash\{r\}\to[m]} \Big[\prod_{k\in\bar{V}\backslash\{r\}} p_{kA(k)} \sum_{U\subseteq V}\Big(\prod_{k\in U} p_{kj} \prod_{k\in V\backslash U}(1-p_{kj}) \\
&\quad [p_{rj} l_j(\{k\in\bar{V}\backslash\{r\} \mid A(k)=j\}\cup U\cup\{i,r\}) \\
&\quad + (1-p_{rj}) l_j(\{k\in\bar{V}\backslash\{r\} \mid A(k)=j\}\cup U\cup\{i\})]\Big)\Big] \\
= &\sum_{A:\bar{V}\backslash\{r\}\to[m]} \Big[\prod_{k\in\bar{V}\backslash\{r\}} p_{kA(k)} \sum_{U\subseteq(V\cup\{r\})} \Big(\prod_{k\in U} p_{kj} \\
&\quad \prod_{k\in(V\cup\{r\})\backslash U}(1-p_{kj}) l_j(\{k\in\bar{V}\backslash\{r\} \mid A(k)=j\}\cup U\cup\{i\})\Big)\Big].
\end{aligned}$$

Hence the claim holds for $V' = V\cup\{r\}$. By induction the claim follows for any $V \subseteq [n]\backslash\{i\}$. This completes the proof of the claim.
Now use the claim with $V = [n]\backslash\{i\}$. This yields the lemma. □

2.7.1.2 Nash Equilibrium Condition

In a Nash equilibrium $\mathbf{P}$, each user i chooses a probability distribution (the ith row of $\mathbf{P}$) which minimizes its expected individual cost. This yields the following lemma.

Lemma 2.4. *A mixed assignment* $\mathbf{P}$ *for an instance* $I = (\mathrm{l})$ *of the generic routing model is at Nash equilibrium, if and only if for each user* i *and for all links* $j \in support(i)$

$$\lambda_i(\mathbf{P} \mid_i j, I) = \min_{k \in [m]} \lambda_i(\mathbf{P} \mid_i k, I)\,. \tag{2.1}$$

Proof. Definition 2.1 yields that $\mathbf{P}$ is at Nash equilibrium, exactly if for each user i and all probability distributions $\mathbf{p} = (p_1, \ldots, p_m)$ for user i it holds $\lambda_i(\mathbf{P}, I) \leq \lambda_i(\mathbf{P} \mid_i \mathbf{p}, I)$. Hence, in any Nash equilibrium $\mathbf{P}$, the ith row of $\mathbf{P}$ must be such that $\lambda_i(\mathbf{P}, I)$ is minimal. From Lemma 2.2 we get that

$$\lambda_i(\mathbf{P}, I) = \sum_{j \in [m]} p_{ij} \lambda_i(\mathbf{P} \mid_i j, I)\,.$$

In order to minimize $\lambda_i(\mathbf{P}, I)$, only those entries p_{ij} of the ith row of $\mathbf{P}$ can be positive, for which $\lambda_i(\mathbf{P} \mid_i j, I)$ is minimal. The Nash equilibrium condition is therefore equivalent to

$$p_{ij} > 0 \quad \Rightarrow \quad \lambda_i(\mathbf{P}|j, I) = \min_{k \in [m]} \lambda_i(\mathbf{P}|k, I)$$

for all users $i \in [n]$ and links $j \in [m]$, which yields the claim. □

2.7.1.3 Social Cost

The social cost of a pure assignment A for an instance I of the generic routing model, according to the definition of SC^{SUM} and SC^{MAX}, can be alternatively computed as follows.

$$SC^{SUM}(I, A) = \sum_{j \in [m]} l_j(view(j, A))|view(j, A)| \quad \text{and}$$

$$SC^{MAX}(I, A) = \max_{\substack{j \in [m] \\ view(j) \neq \emptyset}} l_j(view(j, A))\,.$$

2.7.2 The Unrelated Routing Model

For an instance of the unrelated routing model we can refine the definition of individual cost.

Lemma 2.5. *Let* $\mathbf{P} \in \mathbb{R}^{n \times m}$ *be a mixed assignment for an instance* $I = (\mathbf{C})$ *of the unrelated routing model. Then, the individual cost of user* $i \in [n]$ *according to* $\mathbf{P}$ *is*

$$\lambda_i(\mathbf{P}, I) = \sum_{j \in [m]} p_{ij} \left(c_{ij} + \sum_{\substack{k \in [n] \\ k \neq i}} p_{kj} c_{kj} \right)\,.$$

Proof. First note that for any subset $U \subseteq [n]$ of users

$$\sum_{A: U \to [m]} \prod_{k \in U} p_{kA(k)} = 1\,. \tag{2.2}$$

It is

$$\begin{aligned}
\lambda_i(\mathbf{P}\mid_i j, I) &= \mathcal{E}_{A\vdash \mathbf{P}\mid_i j}(\lambda_i(A, I)) \\
&= \sum_{A:[n]\setminus\{i\}\to[m]} \prod_{k\in[n]\setminus\{i\}} p_{kA(k)} \left(c_{ij} + \sum_{\substack{k\in[n]\setminus\{i\}\\ A(k)=j}} c_{kj} \right) \\
&= c_{ij} \sum_{A:[n]\setminus\{i\}\to[m]} \prod_{k\in[n]\setminus\{i\}} p_{kA(k)} \\
&\quad + \sum_{A:[n]\setminus\{i\}\to[m]} \prod_{k\in[n]\setminus\{i\}} p_{kA(k)} \sum_{\substack{k\in[n]\setminus\{i\}\\ A(k)=j}} c_{kj} \\
&= c_{ij} + \sum_{A:[n]\setminus\{i\}\to[m]} \prod_{k\in[n]\setminus\{i\}} p_{kA(k)} \sum_{\substack{k\in[n]\setminus\{i\}\\ A(k)=j}} c_{kj}\,,
\end{aligned}$$

Claim: For any subset of users $U \subseteq [n]\setminus\{i\}$ with inverse set $\bar{U} = [n]\setminus\{i\}\setminus U$

$$\lambda_i(\mathbf{P}\mid_i j, I) = c_{ij} + \sum_{k\in U} p_{kj} c_{kj} + \sum_{A:\bar{U}\to[m]} \prod_{k\in\bar{U}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\\ A(k)=j}} c_{kj}\,.$$

Proof: For $U = \emptyset$ the claim is obvious. Now fix $U \subseteq [n]\setminus\{i\}$ and assume that the claim holds for this U. We will prove that it holds for $V = U \cup \{r\} \subseteq [n]\setminus\{i\}$ in place of U for any $r \in \bar{U}$. The claim then follows by induction. So, let $r \in \bar{U}$ be arbitrary, and set $V = U \cup \{r\}$. Let $\bar{V} = [n]\setminus\{i\}\setminus V$ be the inverse set of V. We have

$$\begin{aligned}
&\sum_{A:\bar{U}\to[m]} \prod_{k\in\bar{U}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\\ A(k)=j}} c_{kj} \\
&= \sum_{t\in[m]} \sum_{A:\bar{U}\setminus\{r\}\to[m]} p_{rt} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \left(\sum_{\substack{k\in\bar{U}\setminus\{r\}\\ A(k)=j}} c_{kj} + \begin{cases} c_{rj} & t = j \\ 0 & t \neq j \end{cases} \right) \\
&= \sum_{A:\bar{U}\setminus\{r\}\to[m]} p_{rj} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \left(\sum_{\substack{k\in\bar{U}\setminus\{r\}\\ A(k)=j}} c_{kj} + c_{rj} \right) \\
&\quad + \sum_{t\in[m]\setminus\{j\}} \sum_{A:\bar{U}\setminus\{r\}\to[m]} p_{rt} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\setminus\{r\}\\ A(k)=j}} c_{kj}
\end{aligned}$$

$$
\begin{aligned}
&= p_{rj}c_{rj} \sum_{A:\bar{U}\setminus\{r\}\to[m]} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \\
&\quad + p_{rj} \sum_{A:\bar{U}\setminus\{r\}\to[m]} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\setminus\{r\}\\A(k)=j}} c_{kj} \\
&\quad + \sum_{t\in[m]\setminus\{j\}} p_{rt} \sum_{A:\bar{U}\setminus\{r\}\to[m]} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\setminus\{r\}\\A(k)=j}} c_{kj} \\
&= p_{rj}c_{rj} + \sum_{A:\bar{U}\setminus\{r\}\to[m]} \prod_{k\in\bar{U}\setminus\{r\}} p_{kA(k)} \sum_{\substack{k\in\bar{U}\setminus\{r\}\\A(k)=j}} c_{kj} \,.
\end{aligned}
$$

Note that the last equality is due to (2.2) and

$$\sum_{t\in[m]\setminus\{j\}} p_{rt} = 1 - p_{rj} \,.$$

Hence,

$$\lambda_i(\mathbf{P}\,|_i\,, I) = c_{ij} + \sum_{k\in V} p_{kj}c_{kj} + \sum_{A:\bar{V}\to[m]} \prod_{k\in\bar{V}} p_{kA(k)} \sum_{\substack{k\in\bar{V}\\A(k)=j}} c_{kj}$$

which completes the proof of the claim.
Applying the claim with $U = [n]\setminus\{i\}$ yields

$$\lambda_i(\mathbf{P}\,|_i\, j, I) = c_{ij} + \sum_{k\in[n]\setminus\{i\}} p_{kj}c_{kj} \,.$$

Together with Lemma 2.2 this proves the lemma. □

2.7.3 The Congestion Routing Model

2.7.3.1 Social Cost

We will make use of two different definitions of social cost for the congestion routing model. In Chapters 3, 4, 7 and 5 it will be defined as SC^{MAX}, whereas in Chapter 6 we will define social cost as SC^{AVG}. There is a great difference in handling the social cost according to both definitions. In the first case, the social cost of a mixed assignment is the expectation of the maximum of some expression. The expectation involves a summation that must not be exchanged with the computation of the maximum. This makes the first definition quite intractable. The congestion routing model with related links together with this definition of social cost is also known as KP model, since it was first examined by Koutsoupias and Papadimitriou [47] in the context of non-cooperative routing. The second definition allows to significantly simplify the calculation of social cost, as the following lemma shows.

Lemma 2.6. *Let* $\mathbf{P}$ *be a mixed assignment for an instance* $I = (\mathbf{w}, \mathbf{l})$ *of the congestion routing model. Then,*

$$SC^{AVG}(\mathbf{P}, I) = \sum_{i \in [n]} w_i \lambda_i(\mathbf{P}, I)\,.$$

Proof.

$$\begin{aligned} SC^{AVG}(\mathbf{P}, I) &= \mathcal{E}_{A \vdash \mathbf{P}}(SC^{AVG}(A, I)) = \mathcal{E}_{A \vdash \mathbf{P}}(\sum_{i \in [n]} w_i \lambda_i(A, I)) \\ &= \sum_{i \in [n]} w_i \mathcal{E}_{A \vdash \mathbf{P}}(\lambda_i(A, I)) = \sum_{i \in [n]} w_i \lambda_i(\mathbf{P}, I)\,. \end{aligned}$$

□

For instances I of the congestion routing model the social cost of a pure assignment A, according to the definition of SC^{AVG}, can be calculated by summing up over all links instead of summing up the weighted user costs. That is,

$$SC^{AVG}(I, A) = \sum_{j \in [m]} x_j(A) l_j(x_j(A))\,.$$

We will use this relation in Chapter 6.

2.7.3.2 Related Links

In case of related links, the definitions of individual cost and social cost simplify as follows.

The individual cost of user $i \in [n]$ according to a pure assignment A for an instance $I = (\mathbf{w}, \mathbf{s})$ is

$$\lambda_i(A, I) = \frac{\sum_{\substack{k \in [n] \\ A(k) = A(i)}} w_k}{s_{A(i)}} = \frac{x_{A(i)}}{s_{A(i)}}\,.$$

As related links are a special case of unrelated links, we can use Lemma 2.5 with $c_{ij} = \frac{w_i}{s_j}$ for $i \in [n], j \in [m]$ to get that

$$\lambda_i(\mathbf{P}, I) = \sum_{j \in [m]} p_{ij} \frac{w_i + \sum_{\substack{k \in [n] \\ k \neq i}} p_{kj} w_i}{s_j}$$

for a mixed assignment $\mathbf{P}$. The social cost of $\mathbf{P}$, according to the definition of SC^{AVG} and using Lemma 2.6 is hence

$$SC^{AVG}(\mathbf{P}, I) = \sum_{i \in [n]} \sum_{j \in [m]} p_{ij} \frac{w_i^2 + w_i \sum_{\substack{k \in [n] \\ k \neq i}} p_{kj} w_i}{s_j}\,.$$

2.8 Remark

Throughout this work we will sometimes leave out arguments and super- or subscripts when this increases readability and if it does not lead to ambiguity. For instance, $\lambda_i(A)$ is short for $\lambda_i(A, I)$, and $SC(A)$ can mean $SC^{AVG}(A, I)$ or $SC^{MAX}(A, I)$ which will be clear from the context.

— 3 —

Nashification

3.1 Introduction

In this chapter we consider the problem of converting a given pure assignment for an instance of the KP model into a Nash equilibrium without increasing the social cost. We denote this kind of transformation as *Nashification*.

We will give a polynomial time algorithm for Nashification in Section 3.4. In Section 3.5 we show that it may take a superpolynomial number of steps to transform an assignment into a Nash equilibrium by successively performing best moves on the users, if the order of moves is chosen arbitrarily.

3.2 Definitions and Terminology

The results in this chapter concern pure assignments in the original KP model. That is, we consider weighted users on related links, and the social cost is defined as the maximal user cost or equivalently as the maximal link latency.

Let A be a pure assignment for an instance of the KP model. An algorithm that, given A and the instance as input, computes an assignment A' with $SC^{MAX}(A') \leq SC^{MAX}(A)$ and such that A' is at Nash equilibrium is said to *Nashify* A.

In a *selfish step* or *selfish move* one user is reassigned to a new link such that the latency of that user decreases. A *greedy step* is the special case of a selfish step where a user is reassigned to one of its best links. That is, a greedy step leads to the lowest individual cost for that user among all selfish steps. A user is said to be *satisfied* if no selfish step is possible for this user. An assignment is at Nash equilibrium, if and only if all users are satisfied.

3.3 Motivation and Context

Provided that the n user flows and the m link speeds are given in sorted order our Nashification algorithm `Nashify` in Figure 3.1 computes a Nash equilibrium in time $\mathcal{O}(nm)$ when started on an arbitrary pure assignment. Thus, we provide another algoritm besides the LPT algorithm (sometimes called Graham's algorithm) [39, 29] to compute a pure Nash equilibrium for any instance of the KP model.

On the other hand, and more important, our algorithm can be used to compute a Nash equilibrium of any desired quality, by starting it on a correspondingly good initial assignment. Such an initial assignment can be computed using an approximation algorithm for job scheduling on related machines which is equivalent to our model.

In [32] Friesen and Langston give such an algorithm with approximation ratio at most 1.4 which runs in time $\mathcal{O}(nm)$. The approximation ratio of the LPT algorithm is at most 1.67 but at least 1.52 [31]. In [42] Hochbaum and Shmoys give a polynomial time approximation scheme (PTAS) for the scheduling problem. Hence, combining their algorithm with ours, we get a PTAS for the problem of computing the best Nash equilibrium. As computing the best Nash equilibrium is known to be NP-hard in the strong sense [29], there cannot exist a fully polynomial time approximation scheme (FPTAS) for this problem, unless $P = NP$.

3.4 A Fast Nashification Algorithm

Figure 3.1 shows our Nashification algorithm. A crucial observation for proving the correctness of the algorithm is stated by the following lemma.

Lemma 3.1. *Let A be any pure assignment. Assume some user i performs a greedy move from its current link $j = A(i)$ to some link k with $s_j \leq s_k$, yielding a new assignment A' with $A'(i) = k$. Then, $\lambda_r(A) = \min_{t\in[m]} \lambda_r(A\,|_r\, t)$ implies $\lambda_r(A') = \min_{t\in[m]} \lambda_r(A'\,|_r\, t)$ for any $r \in [n]$ with $w_r \geq w_i$.*

Proof. Let the assumptions of the lemma hold and consider any user r with $w_r \geq w_i$ and $\lambda_r(A) = \min_{t\in[m]} \lambda_r(A\,|_r\, t)$. Let user r be located on link $s = A(r)$. Assume by way of contradiction, that there exists some link t with $\lambda_r(A') > \lambda_r(A'\,|_r\, t)$. Only the traffic on link j and k changes (it is decreased on link j and increased on link k) due to the move of user i. This implies $s = k$ or $t = j$.
Assume first, that $s \neq k$ (and therefore $t = j$). We have

$$\frac{x_s(A)}{s_s} = \lambda_r(A) \leq \lambda_r(A\,|_r\, k) = \frac{x_k(A) + w_r}{s_k}\,.$$

User i improves by moving to link k, thus

$$\frac{x_k(A) + w_i}{s_k} = \lambda_i(A') < \lambda_i(A) = \frac{x_j(A)}{s_j}\,,$$

and we get

$$\begin{aligned}\lambda_r(A' \mid_r t) &= \frac{x_j(A) - w_i + w_r}{s_j} > \frac{x_k(A) + w_i}{s_k} + \frac{w_r - w_i}{s_j} \\ &\geq \frac{x_s(A)}{s_s} - \frac{w_r}{s_k} + \frac{w_i}{s_k} + \frac{w_r - w_i}{s_j} \\ &= \frac{x_s(A)}{s_s} + (w_r - w_i)(\frac{1}{s_j} - \frac{1}{s_k}) \geq \frac{x_s(A)}{s_s} = \frac{x_s(A')}{s_s} = \lambda_r(A')\,,\end{aligned}$$

a contradiction.
Now assume $s = k$. First note that

$$\lambda_r(A' \mid_r j) = \frac{x_j(A) - w_i + w_r}{s_j} \geq \frac{x_j(A)}{s_j} = \lambda_i(A) > \lambda_i(A') = \frac{x_k(A) + w_i}{s_k} = \lambda_r(A')$$

implies $t \neq j$.
Since user i performs a greedy step, we have

$$\frac{x_t(A) + w_i}{s_t} \geq \frac{x_k(A) + w_i}{s_k}\,,$$

and hence

$$\lambda_r(A' \mid_r t) = \frac{x_t(A) + w_r}{s_t} \geq \frac{x_t(A) + w_i}{s_t} \geq \frac{x_k(A) + w_i}{s_k} = \lambda_r(A')\,,$$

a contradiction. □

Put in other words, Lemma 3.1 claims that a greedy move of some user from its current link to a non-slower link cannot cause any non-smaller user to become unsatisfied. For the special case of identical links, every move is a move to a non-slower link. This implies that any greedy move of some user does not unsatisfy any user with larger or equal traffic. Thus, by successively moving each user to its best link in order of non-increasing traffic sizes, we end up in a Nash Equilibrium. As greedy steps do not increase the social cost, this algorithm Nashifies any pure assignment on identical links in n steps [35, 23].

With algorithm `Nashify` in Figure 3.1 this idea is generalized to non-identical links. The input to the algorithm consists of the user flows, the link speeds and a pure assignment A. User flows and link speeds are supposed to be sorted non-increasingly.

The algorithm works in two phases. At every time, $A(i)$ denotes the link user i is currently assigned to. Whenever user i is moved, $A(i)$ has to be updated which is not explicitly mentioned in the pseudocode.

The main idea is to accumulate users with small traffic on slow links in a special way in the first phase (without increasing the social cost) and to perform greedy steps for unsatisfied users in the second phase.

We may assume that the overall flow on each link is computed as a first step (in time $\mathcal{O}(n + m)$) and updated whenever it changes (in constant time). Then, setting M to the

```
    Nashify(I, A)
    Input:    instance I of the routing model
                 with sorted users and links
              initial assignment A
    Output:   equilibrium assignment A with
                 non-increased social cost
        // phase 1:
01      M := SC^MAX(A);
02      S := {n};
03      mini(A(n)) := n;
04      repeat
            // begin of sweep
05          i := n;
06          j := m;
07          while i ∈ S do
08              if j > A(i) and user i fits on link j
                       without exceeding latency M then
09                  put user i on link j;
10                  mini(j) := i;
11                  i := i − 1;
12              else if j > A(i) then
13                  j := j − 1;
14              else i := mini(A(i)) − 1;
            // end of sweep
15          if i ≥ 1 then
16              put user i on link with smallest index
                  without exceeding latency M;
17              if A(i) ≤ A(i + 1) then
18                  S := S ∪ {i};
19                  mini(A(i)) := i;
20              else i := 0;
21      until i < 1
        // phase 2:
22      while S ≠ ∅ do
23          i := min(S);
24          make greedy move for user i;
25          S := S\{i};
```

Figure 3.1: Algorithm Nashify computes a Nash equilibrium for any instance $I = (\mathbf{w}, \mathbf{s})$ of the KP model and assignment A without increasing the social cost. Users and links must be sorted so that user flows and link speeds are ordered non-increasingly, i.e., $w_1 \geq \cdots \geq w_n$ and $s_1 \geq \cdots \geq s_m$.

social cost of the initial assignment which is the first step of phase 1 (line `01`) takes time $\mathcal{O}(m)$.

The set S is used during the first phase to collect all those users that have already been considered and for which certain conditions (*(ii)* and *(iii)* of Lemma 3.2) have been established. These conditions will be explained later. For the time being, just let us remark that they are fulfilled trivially for $S = \{n\}$ (line `02`).

The array $mini()$ is used during the first phase to store for each link the smallest index of a user located on that link and contained in S. If no such user exists, then the entry of $mini()$ for that link is undefined. That is, for $j \in [m]$

$$mini(j) = \begin{cases} \min(S \cap view(j)) & if\ S \cap view(j) \neq \emptyset \\ undefined & otherwise \end{cases} . \tag{3.1}$$

We denote a complete execution of lines `05-14` (until the while-loop is finished) a *sweep*. During a sweep each user in S is put on the link with highest possible index it fits on without increasing the latency on that link (and hence the social cost) beyond M (the initial social cost). The users are taken up in reverse order of there indices, always starting with user n.

After a sweep, i points to the user with highest index among all users which are not contained in S. If $i = 0$ at this point, then all users are in S which causes the algorithm to leave the first phase (line `21`). Otherwise the current user is reassigned to the link with smallest index it fits on without increasing the social cost beyond M (line `16`). This can possibly mean that the user is not moved at all. If it is now located on a link with non-higher index than user $i+1$, then it is added to S and phase 1 goes on with the next sweep (line `17-19`). Otherwise the first phase is quitted by setting i to zero (line `20`).

So, after each sweep, either a new user is added to S or the first phase is terminated. For the latter case we will prove (Lemma 3.2) that all unsatisfied users are in S and that they are arranged in a special way.

In the second phase we successively perform greedy steps for all unsatisfied users in S in order of increasing indices, i.e., in order of non-increasing traffic sizes. Because of the special conditions that have been established by phase 1, and by Lemma 3.1, these greedy moves do not cause other users with larger traffic to become unsatisfied.

To prove correctness we will proceed in two steps. First we prove that phase 1 establishes four postconditions (Lemma 3.2). Then we show that the same conditions remain valid throughout the execution of the second phase (Lemma 3.3). It is then a simple matter to conclude that the assignment is at Nash equilibrium at the end of phase 2 and that the social cost has not increased (Theorem 3.4).

Lemma 3.2. *After phase 1 the following holds:*

(i) *All unsatisfied users are in S.*

(ii) $S = \{n, (n-1), \ldots, (n+1-|S|)\}$, *that is, S contains the $|S|$ users with highest indices.*

(iii) $i, i+1 \in S \Rightarrow A(i) \leq A(i+1)$.

(iv) Every user $i \in S$ can only improve (if at all) by moving to a link with smaller index.

Proof. Throughout the first phase $mini()$ is kept valid (except for the transitions from line `02` to `03`, from line `09` to `10` and from line `18` to `19`). Let us state this as the fifth property:

(v) Array $mini()$ is valid according to (3.1) throughout phase 1.

Actually, all we need is that *(v)* is valid at line `14`, the only time when $mini()$ is read. We first prove properties *(ii)*, *(iii)* and *(v)*, which are strongly interrelated, to be invariants that hold at any time throughout phase 1. Obviously, all three hold before the first run of the repeat-until-loop. Now consider the first time when *(ii)*, *(iii)* or *(v)* becomes invalid.

Assume that *(ii)* is the first property to become invalid. Note that a user which is included in S, will never be removed from S during the first phase. Hence *(ii)* can only become invalid when a new user is added to S after a sweep in line `18`. After a sweep, i (if it is not zero) points to the user with the largest index that is not contained in S so far. This is because a sweep is finished if $i \notin S$ and i is either decremented by one (line `11`) during an iteration of the sweep or, as *(v)* is valid and $mini(A(i))$ is (particularly) a user contained in S, set (in line `14`) to an index which is at most by one lower than that of some user in S. Thus, S is always a consecutive set of the users with the $|S|$ largest indices as stated by *(ii)*.

Now assume that *(iii)* becomes invalid first. There are two points in the algorithm where *(iii)* could possibly become invalid. The first is in line `09`, where a user $i \in S$ is moved during a sweep. The second is in line `18`, where a new user is added to S.

If a user $i \in S$ is moved during a sweep in line `09` to a link j, then this could invalidate *(iii)* in two ways: First, if $i > 1$, then $j < A(i-1)$ would cause *(iii)* to be invalid after the reassignment. But as $j > A(i)$ (because of line `08`) this would imply $A(i) < A(i-1)$, and hence *(iii)* would have been invalid before, contradicting our assumption. Second, if $i < n$, then $j > A(i+1)$ would invalidate *(iii)*. But consider the last run of the while-loop where the value of i was greater than it is now. Either it was $i+1$ and has been decremented to i in line `11`. But then user $i+1$ has been assigned to link j in line `09`, and as j is never increased during a sweep, we must now have $j \leq A(i+1)$. Or i was decreased by line `14` to its current value. Let i' be the former value of i (before the decreasement). Then $mini(A(i')) = i+1$ (because of line `14`). In particular, this implies $A(i+1) = A(i')$ due to the definition of $mini()$. As line `14` has been executed, we know that the if-condition in line `12` was false, and hence, $j \leq A(i') = A(i+1)$ at that point of time. As j is never increased during a sweep, $j \leq A(i+1)$ is still true.

Line `18` is only executed if this does not invalidate *(iii)* which is explicitly inquired in line `17`. Note that $i-1 \notin S$ because of *(ii)* and the fact that $i \notin S$ (before the execution of line `18`).

At last assume that *(v)* becomes invalid first. This could only happen due to the execution of line `09` or line `18`. If line `09` is executed, then user i has the smallest index among

all users on link j contained in S, because *(iii)* implies (by induction) that $A(r) \leq A(i)$ for all $r \in S$ with $r < i$, and, due to line `08`, $A(i) < j$. Thus $mini()$ is properly updated in line `10`. When a new user is added to S in line `18`, then it must have the smallest index among all users in S (due to *(ii)*), hence the updating of $mini()$ in line `19` is correct.

To show *(iv)*, we first prove that *(iv)* holds directly after any sweep and then argue that the assignment is not changed (in line `16`) after the last sweep. (Actually we will prove a much stronger claim than *(iv)*, namely that any user $i \in S$ cannot be moved to any link $j > A(i)$ without exceeding the initial social cost M, that is, $x_j(A) + w_i > Ms_j$ for all $j \in [m]$ with $j > A(i)$.)
Assume by way of contradiction that after some sweep there is some user r which fits on some link $s > A(r)$ without exceeding latency M. Without loss of generality, let this be the first sweep for which such user exists, and let r be the largest index of such user. Hence, we know that user $r + 1$ does not fit on any link $j > A(r + 1)$. As $w_r \geq w_{r+1}$, user r does not fit on any link $j > A(r + 1)$. If $A(r) = A(r + 1)$, then we are done.
Otherwise, let i' be the last value of variable i greater than r. When i is decreased from i' to r in line `11` or `14`, we know that user i' does not fit on any link $j' > A(i')$. This implies, as $w_r \geq w_{i'}$, that user r cannot fit on any link $j' > A(i')$, too, and therefore $s \leq A(i')$. If user r is located on the same link as user i', i.e., $A(r) = A(i')$, then we are done. Otherwise, $A(r) < A(i')$ (due to *(iii)*). During the following iterations of the while-loop, j is successively set to $A(i'), A(i')-1, \ldots, A(r)+1$. As $s \in \{A(i'), A(i')-1, \ldots, A(r)+1\}$, user r would have been moved to link s by line `08` in the iteration of the while-loop with $j = s$ if it would fit on link s, a contradiction.

It remains to show that after the last sweep the assignment is not changed. This could only happen if user i is moved to some link $j > A(i+1)$ in line `16`, because if it is moved to some link $j \leq A(i + 1)$, then the repeat-until-loop proceeds with a new sweep. But if user i is moved to some link $j > A(i + 1)$, then, as $w_i \geq w_{i+1}$, user $i + 1$ had fit on link j before user i is moved. This contradicts the above proof for *(iv)* to hold after any sweep.

At last we prove *(i)*. The repeat-until-loop in phase 1 can only be terminated if i becomes zero. This can happen in two different ways. Either i becomes zero at the end of the last sweep, or it is set to zero in line `20`. In the first case all users are in S, which implies *(i)*. In the second case we know that $A(i) > A(i + 1)$ (because of line `17`), that user i does not fit on any link $j < A(i)$ (because of line `16`) and that user $i + 1$ cannot be moved to any link $j > A(i + 1)$ (due to *(iv)*). Due to *(ii)*, each user $r \notin S$ controls at least the same amount of traffic as users i and $i + 1$, i.e., $w_r \geq w_i \geq w_{i+1}$. Hence, none of the users $r \notin S$ can fit on any link on which w_i does not fit or on which w_{i+1} does not fit, that is, on any link in $\{j \in [m] \mid j < A(i)\} \cup \{j \in [m] \mid j > A(i + 1)\} = [m]$. □

Our next step is to show that properties *(i)*,*(ii)*,*(iii)* and *(iv)*, stated in Lemma 3.2 as postconditions of phase 1, remain true during (and after) phase 2 as well.

Lemma 3.3. *After any run of the while-loop in phase 2 the four properties from Lemma 3.2 remain valid, provided that they had been valid before.*

Proof. Consider any run of the while-loop and assume the conditions of Lemma 3.2 to hold beforehand. Let i be the user with smallest index in S, and suppose it is moved from link $j = A(i)$ to its best link k. Because of *(ii)*, we have $i = n + 1 - |S|$. *(iv)* implies $k \leq A(i)$ and therefore $s_k \geq s_j$.
Now let $r \notin S$ be any user on some link $s = A(r)$. Due to Lemma 3.1 and since $w_r \geq w_i$, r cannot be unsatisfied after i has been moved. Thus, *(i)* still holds after moving i. As i is removed from S (in line `25`) and $i = n+1-|S|$, *(ii)* and *(iii)* still hold after the iteration. *(iii)* and the fact that i was moved to a link $j \leq A(i)$ imply that *(iv)* remains true after the iteration. □

Having Lemmas 3.2 and 3.3 available, we can now conclude that algorithm `Nashify` is correct and bound its running time.

Theorem 3.4. *Given any pure assignment, algorithm* `Nashify` *computes a Nash equilibrium with non-increased social cost, performing at most* $(m + 1)n$ *moves. Provided that user flows and link speeds are sorted, its running time is* $\mathcal{O}(nm)$.

Proof.

Correctness
After execution of phase 1 properties *(i)-(iv)* from Lemma 3.2 hold. Applying Lemma 3.3 inductively shows that properties *(i)-(iv)* remain valid beyond the end of the algorithm. Combining property *(i)* with the fact that set S is empty at the end of the algorithm (due to line `22`) proves that the assignment is at Nash equilibrium when the algorithm terminates.

As the latency of any link is never increased beyond the initial value of the social cost throughout the algorithm, the social cost of the computed Nash equilibrium will be no more than that of the initial assignment.

Running time
We now analyze the running time of the first phase. Lines `01`,`02` and `03` are executed once. Lines `04`,`05`,`06`,`15` and `21` are executed once for each iteration of the repeat-until-loop. Denote the number of iterations of the repeat-until-loop as r. Lines `16`,`17`,`18` and `19` are executed at most r times. Line `20` is executed once. Denote the overall number of iterations of the while-loop in lines `07-14` as w. Then, lines `08-14` are executed at most w times. The while-statement in line `08` itself is executed $w + 1$ times.

We assume that the flow on each link is computed in advance by adding all user flows assigned to the respective link, which takes time $\mathcal{O}(m + n)$. Whenever the assignment is changed during the algorithm, the link flows have to be updated which takes only constant time. With these precomputations it takes time $\mathcal{O}(m)$ to execute lines `01` and `16`, and all other lines of phase 1 take constant time. Hence, the running time of phase 1 is $\mathcal{O}(rm+w)$. The repeat-until-loop is iterated a most n times, because in each iteration, except for the last maybe, one user (more precisely its index) is added to S. Once in S a user is never removed from S again during the first phase. Starting with $S = \{n\}$, at most $n - 1$ users can be added to S. Thus, $r \leq n$. During one iteration of the while-loop (lines `07-14`) either lines `09-11` or line `13` or line `14` are executed. By lines `09-11` a user from S

located on some link $A(i)$ is moved to some link with higher index $j > A(i)$. This user is never moved back again (to a link with lower index) during the first phase. As there are m links, this can happen at most $m-1$ times for each user. Hence, there are at most $n(m-1)$ iterations of the while-loop that execute lines `09-11` throughout the first phase. In line `13` j is decremented if $j-1 > A(i)$. This can happen at most $m-2$ times during one iteration of the repeat-until-loop, because j is started at m, is never increased during the while-loop and $A(i) \geq 1$. It follows that there are at most $r(m-2)$ runs of the while-loop that execute line `13` altogether. In line `14`, i is set to a user which is either not in S or which is located on a link with smaller index than the user to which i is currently pointing. In the first case the while-loop is not continued which happens once per iteration of the repeat-until-loop. The second case can appear at most $m-1$ times during one iteration of the repeat-until-loop. Hence, line `14` is executed at most rm times throughout phase 1. The amortized number w of iterations of the while-loop is therefore at most $n(m-1) + r(m-2) + rm = \mathcal{O}(nm)$. Thus, the running time of phase 1 is $\mathcal{O}(nm)$.

The while-loop in phase 2 is iterated at most $|S| \leq n$ times. During each iteration one user has to be put on its best link, which takes time $\mathcal{O}(m)$. Hence, the running time of phase 2 is $\mathcal{O}(nm)$.

Number of Moves

Now we bound the total number of reassignment steps. In phase 1 each user is shifted at most once to a link with smaller index (in line `16`). Afterwards it is contained in S and hence line `16` can never be executed again for that user. Once in S, a user can be shifted only to a link with higher index than its current link (in line `09`). This can happen at most $m-1$ times. In phase 2 each of the users in S is reassigned at most once. Thus, the total number of moves is bounded by $n(m+1)$. □

Apart from the routing model with related links as considered here, the algorithm can also cope with slightly more general models. All we need is that users and links can be ordered in a certain way so that the properties stated by the following proposition hold.

Proposition 3.5. *Let $I = (\mathbf{1})$ be an instance of the generic routing model, and let A be any pure assignment for I. If the following three properties hold, then algorithm* `Nashify`*, started on input (I, A), Nashifies A.*

(i) Let A' be any assignment that is attained during the execution of `Nashify` *on input (I, A). Let j be any link, and set $U = view(j, A') = \{r \in [n] \mid A'(r) = j\}$. Then, $\forall i, r \in [n] \backslash U, r < i : l_j(U \cup \{r\}) \geq l_j(U \cup \{i\})$.*

(ii) Let A' and A'' be any two consecutive assignments that are attained during the execution of algorithm `Nashify` *on input (I, A). Since, A' and A'' differ only in the assignment of one user, say user i, we have $A'(i) = j$, $A''(i) = k \neq j$ and $A'(r) = A''(r)$ for all $r \in [n] \backslash \{i\}$. Set $U = view(j, A') \backslash \{i\} = \{r \in [n] \backslash \{i\} \mid A'(r) = j\}$ and $V = view(k, A') = \{r \in [n] \mid A'(r) = k\}$. If $j > k$, then $\forall r \in [n] \backslash U, r < i$: $l_j(U \cup \{r\}) - l_k(V \cup \{r\}) \geq l_j(U \cup \{i\}) - l_k(V \cup \{i\})$.*

(iii) Let A' be any assignment that is attained during the execution of `Nashify` *on input (I, A). Let j be any link, and set $U = view(j, A') = \{r \in [n] \mid A'(r) = j\}$. Then, $\forall i \in [n] : l_j(U \cup \{i\}) \geq l_j(U)$.*

Proof. In other words, property *(i)* claims that adding a user to some link causes the same or higher latency than adding any user with larger index to the link instead. This ensures that the first phase of the algorithm works correctly, because all we need for the first phase (see proof of Lemma 3.2) is that if some user fits on some link without increasing the social cost, then any user with larger index does as well.

Properties *(ii)* and *(iii)* are used for proving Lemma 3.6 which is a generalized version of Lemma 3.1 and assures that the second phase works properly (by the same arguments as in the proof of Lemma 3.3). Put in words, *(ii)* states that moving a user from some link to any link with smaller index decreases its experienced latency by at least the same amount as for a user with larger index being moved between the same links. Property *(iii)* simply states that the latency cannot decrease if additional flow is routed through a link. □

Lemma 3.6. *Let properties (i), (ii) and (iii) of Proposition 3.5 hold for an execution of* `Nashify` *on input (I, A), where $I = (\mathbb{1})$ is an instance of the generic routing model and A is any pure assignment for I. Consider any step of the second phase of the algorithm. Let A' and A'' be the assignments before and after this step, respectively. Assume some user i is greedy-moved from its current link $j = A'(i)$ to some link k with $k < j$, yielding an assignment A'' with $A''(i) = k$. Then, $\lambda_r(A) = \min_{t\in[m]} \lambda_r(A\,|_r\, t)$ implies $\lambda_r(A'') = \min_{t\in[m]} \lambda_r(A''\,|_r\, t)$ for any $r \in [n]$ with $r < i$.*

Proof. Let the assumptions of the lemma hold and consider any user r with $r < i$ and $\lambda_r(A') = \min_{t\in[m]} \lambda_r(A'\,|_r\, t)$. Let user r be located on link $s = A'(r)$. Assume by way of contradiction, that there exists some link t with

$$\lambda_r(A'') > \lambda_r(A''\,|_r\, t)\,. \tag{3.2}$$

Since only the traffic on link j and k changes (it is decreased on link j and increased on link k) due to the move of user i, and because of property *(iii)* (which excludes $t = k$), we have $s = k$ or $t = j$.
Assume first, that $s \neq k$ (and therefore $t = j$). Let us write $view(p)$ for the view of link p according to assignment A', i.e., $view(p) = view(p, A')$ for any link $p \in [m]$. Then we have

$$l_s(view(s)) = \lambda_r(A') \leq \lambda_r(A'\,|_r\, k) = l_k(view(k) \cup \{r\})\,. \tag{3.3}$$

User i improves by moving to link k, thus

$$l_k(view(k) \cup \{i\}) = \lambda_i(A'') < \lambda_i(A') = l_j(view(j))\,, \tag{3.4}$$

and we get

$$\begin{aligned}\lambda_r(A'' |_r t) &= l_j(view(j)\backslash\{i\} \cup \{r\}) \\ &\overset{(3.4)}{>} l_j(view(j)\backslash\{i\} \cup \{r\}) + l_k(view(k) \cup \{i\}) - l_j(view(j)) \\ &= l_j(V \cup \{r\}) + l_k(U \cup \{i\}) - l_j(V \cup \{i\}) \\ &\quad \text{with } U = view(k) \text{ and } V = view(j)\backslash\{i\} \\ &\overset{(ii)}{\geq} l_k(U \cup \{r\}) \\ &= l_k(view(k) \cup \{r\}) \overset{(3.3)}{\geq} l_s(view(s)) = \lambda_r(A'')\,,\end{aligned}$$

contradicting (3.2).
Now assume $s = k$. First note that

$$\begin{aligned}\lambda_r(A'' |_r j) &= l_j(view(j)\backslash\{i\} \cup \{r\}) = l_j(U \cup \{r\}) \text{ with } U = view(j)\backslash\{i\} \\ &\overset{(i)}{\geq} l_j(U \cup \{i\}) = l_j(view(j)) = \lambda_i(A') > \lambda_i(A'') = \lambda_r(A'')\end{aligned}$$

implies $t \neq j$.
Since user i has performed a greedy step, we have

$$l_t(view(t) \cup \{i\}) \geq l_k(view(k) \cup \{i\})\,, \tag{3.5}$$

and hence

$$\lambda_r(A'' |_r t) = l_t(view(t) \cup \{r\}) \overset{(i)}{\geq} l_t(view(t) \cup \{i\}) \overset{(3.5)}{\geq} l_k(view(k) \cup \{i\}) = \lambda_r(A'')\,,$$

a contradiction to (3.2) . □

If the properties of Proposition 3.5 hold for *all* assignments, then, in particular, they hold for those assignments that are attained during the execution of `Nashify`. In this way we can simplify the properties, as stated by the following corollary. Nevertheless, the exact properties of Proposition 3.5 will be useful in Chapter 5.

Corollary 3.7. *Let $I = (\mathbf{1})$ be an instance of the generic routing model, where users and links are numbered such that*

(i)' $\forall j \in [m], i \in [n-1], U \subseteq [n]\backslash\{i, i+1\} : l_j(U \cup \{i\}) \geq l_j(U \cup \{i+1\})$ *and*

(ii)' $\forall j \in [m-1], i \in [n-1], U, V \subseteq [n]\backslash\{i, i+1\}$:
$l_{j+1}(U \cup \{i\}) - l_j(V \cup \{i\}) \geq l_{j+1}(U \cup \{i+1\}) - l_j(V \cup \{i+1\})$ *and*

(iii)' $\forall j \in [m], i \in [n], U \subseteq [n]\backslash\{i\} : l_j(U \cup \{i\}) \geq l_j(U)$.

Then algorithm `Nashify` Nashifies any pure assignment.

Proof. We claim that properties *(i)'*, *(ii)'* and *(iii)'* imply properties *(i)*, *(ii)* and *(iii)* of Proposition 3.5. Property *(i)* follows by induction on i from property *(i)'*. *(ii)* follows by induction on i and j from *(ii)'*. *(iii)* and *(iii)'* are essentially the same. Note that Proposition 3.5 requires the properties only to hold for the assignments that are actually attained by algorithm `Nashify`, whereas *(i)'*, *(ii)'* and *(iii)'* yield more general versions that hold for all assignments. □

For the KP model the latency on a link j is defined as $l_j(U) = \sum_{i \in U} \frac{w_i}{s_j}$, where $U = view(j)$. So, any instance of this model possesses the properties stated in Corollary 3.7. E.g., $l_j(U) = f(\sum_{i \in U} \frac{w_i}{s_j})$ with any non-decreasing and convex function f is another appropriate definition obeying the properties of Corollary 3.7 and therefore also those of Proposition 3.5.

We are not aware of any other efficient algorithm for approximating best Nash equilibria in the KP model up to a given quality. Hence, our approach is the first to yield a Nash equilibrium of arbitrary constant approximation ratio in polynomial time. The work which bears the closest similarity to ours, is that of Hurkens and Vredeveld [43]. They explore so called *jump optimal* assignments. An assignment is said to be *jump optimal*, if no user on a link with maximal latency can improve by a selfish step. They give an algorithm for computing jump optimal assignments in at most $n(m-1)$ steps. As the Nash equilibrium property particularly implies jump optimality, our algorithm computes a jump optimal assignment in about the same number of steps. However, our algorithm achieves asymptotically better worst case running time.

3.5 Convergence to a Nash Equilibrium

In the previous section we addressed the question of how to compute a Nash equilibrium assignment of guaranteed quality in the KP model by a central algorithm. In this section we will explore the process of finding a Nash equilibrium if the users are being left to their own decisions.

During this process we allow only one user at a time to change its link. This user is chosen randomly. All users are provided with complete information, that is, they know the speed and the current flow of each link. We assume that the user which is allowed to move goes to its best link, i.e., that it makes a greedy step.

First of all this process always comes to a Nash equilibrium after a finite number of moves. The validity of this claim does not require the assumption of greedy steps. That is, the number of general selfish steps is finite as well. This can be proven by exploring the vector of non-increasingly ordered link latencies [23]. With every selfish step its lexicographic order decreases. The number of different vectors is bounded by the number of different pure assignments, which is m^n. Hence, there cannot be more than m^n selfish steps. This argument has also been used in [29] to prove the existence of a pure Nash equilibrium in the KP model.

We now show that the process of finding a Nash equilibrium can take a superpolynomial number of steps if the users are unguided, even on identical links, and even if we allow greedy steps only.

We give an instance with n users for which there exists a sequence of greedy steps of length at least $2^{\frac{3}{2}\sqrt{n}-6}$.

Definition 3.8. *Define instance $I(m,a)$ of the KP model with m identical links, $a, m \in \mathbb{N}_{\geq 2}$, as follows:*
There are $m-1$ classes of users $U_1, \ldots, U_{m-1}$. The number of users in class $i \in [m-1]$ is $u_i = a+1+(m-i)a$. Each user r of class $m-1$ has traffic size $w_r = 1$, and each user r in class $i \in [m-2]$ has traffic size $w_r = 1 + \sum_{j=i+1}^{m-1} \sum_{k \in U_j} w_k$.

For instance $I = I(m,a)$ let $A(I)$ be the assignment where all users go to link 1. We consider the process where users successively perform greedy steps until a Nash equilibrium is reached and hence no more selfish steps are possible. Often many users can improve by switching their link. In this case we have to choose which user is allowed to move next. We do so by always choosing a user of smallest traffic size and refer to this as Smallest-Traffic-First (STF) selection rule. Ties are broken arbitrarily. We show that the STF selection rule, and hence random selection in the worst case, leads to a superpolynomial number of steps.

Theorem 3.9. *Consider instance $I(m,a)$ of the KP model for $a, m \in \mathbb{N}_{\geq 2}$. Starting with initial assignment $A(I(m,a))$ and letting the users successively perform greedy steps according to the STF selection rule it takes at least $a^{m-1} \geq a^{\sqrt{\frac{2}{a}n}-2}$ steps to reach a Nash equilibrium.*

Proof. Let $t(m,a)$ denote the number of steps for instance $I = I(m,a)$ until a Nash equilibrium is reached starting from initial assignment $A(I)$ where all users go to link 1 and selecting users according to the STF rule.

For $m = 2$ we initially have $u_1 = 2a+1$ users of size 1 on link 1 while link 2 is empty. Hence, a of the users will move from link 1 to link 2 which takes a steps. This proves $t(2,a) = a$.

Let $m \geq 3$, and let the claim be valid for $m-1$. That is $t(m-1,a) \geq a^{m-2}$. Consider instance $I = I(m,a)$ with initial assignment $A(I)$. There are $am+1$ users in class U_1. Initially, their flows are assigned to link 1 (as for all other users). As each of them is larger than the total flow of all users in $U_2, \ldots, U_{m-1}$, in a Nash equilibrium they must be distributed among the m links such that one link carries $a+1$ of them and all other links have exactly a. This implies that $a(m-1)$ of them will move away from link 1 during the process. Let t_2 be the point in time when the first $m-1$ users of class U_1 have been moved away from link 1. Note that they must go to different links. Let t_1 be the time just before the last of these users is moved. At time t_1 no user from $U_2, \ldots, U_{m-1}$ can improve, because otherwise the user of class U_1 would not be allowed to move since we always choose the smallest traffic first. Hence, at time t_1 we have $am+1-(m-2)$ users of class U_1 on link 1, we have $m-2$ other links which carry one user from class U_1 and no other users, and we have one link which contains all other users. The latter

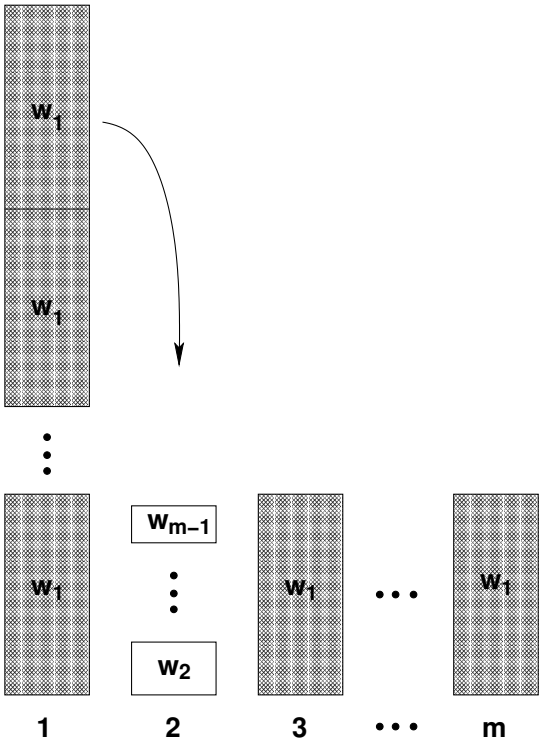

Figure 3.2: Link 1 contains $(a-1)m+3$ users of class U_1. Links $3,\ldots,m$ contain one user of class U_1 each. Link 2 contains all remaining users of classes $U_2,\ldots,U_{m-1}$. w_i is the traffic size of any user from class i.

has minimal latency among all links, and without loss of generality we may assume that it is link 2 (we may renumber the links appropriately). Thus the next user from link 1 will go to link 2. Figure 3.2 illustrates this step. After that, links $2, \ldots, m$ induce an instance $I' = I(m-1, a)$ of $m-1$ links with initial assignment $A' = A(I')$ and we can apply the induction hypothesis and get that it takes $t(m-1, a) \geq a^{m-2}$ steps until a Nash equilibrium is reached for those links. Note that each link of I' contains one additional traffic (from class U_1). This raises the latency on each link by the same amount but has no influence on the moves that are carried out for I'. When all steps for I' are done we can repeat our argumentation. After the next $m-1$ users have moved from link 1 we again have an initial assignment for I' where all users from $U_2, \ldots, U_{m-1}$ are assigned to the first link of I' (which is link 2 for instance I). Hence, again $t(m-1, a)$ moves are necessary until the next user from link 1 is allowed to move.

Initially there are $am + 1$ users of class U_1 on link 1. Hence it happens a times that $m-1$ of them are moved from link 1 yielding an initial assignment A' for instance I'. Each time it takes $t(m-1, a)$ steps to Nashify the sub instance I'. This is $at(m-1, a) \geq a^{m-1}$ steps altogether, which proves that $t(m, a) \geq a^{m-1}$.

The total number of users for instance $I(m, a)$ is

$$n = \sum_{i \in [m-1]} u_i = \sum_{i \in [m-1]} [a + 1 + (m-i)a] = (m-1)(a + 1 + a\frac{m}{2}) < \frac{a}{2}(m+1)^2 ,$$

which implies $\sqrt{\frac{2}{a}n} - 1 \leq m$. Thus we can bound the total number of moves from below by $t(m, a) \geq a^{m-1} \geq a^{\sqrt{\frac{2}{a}n}-2}$. □

The term $a^{\sqrt{\frac{2}{a}}}$, with $a \in \mathbb{N}_{\geq 2}$, is maximized for $a = 7$. Hence for instance $I(m, 7)$ we obtain at least $7^{\sqrt{\frac{2}{7}n}-2} \geq 2.829^{\sqrt{n}-4}$ steps. So, on one hand, the worst case number of greedy steps for unguided users on identical links is at least $2.829^{\sqrt{n}-4} > 2^{\frac{3}{2}(\sqrt{n}-4)}$. On the other hand, it is shown in [27] that the number of greedy steps is bounded by $2^n - 1$ for identical links.

Theorem 3.10. *([27]) For any instance with n users on identical links, the length of any sequence of greedy steps is at most $2^n - 1$.*

Similarly, but independently, Even-Dar et al. [23] show that the number of greedy steps for n users with $k \leq n$ different flow sizes on identical links is bounded from above by $\mathcal{O}((\frac{n}{k} + 1)^k)$.

— 4 —

The Nash Equilibrium Verification Problem

4.1 Introduction

In this chapter we deal with the question of how to decide whether a given pure assignment for the KP model is at Nash equilibrium or not. We will refer to this as the *Nash Equilibrium Verification* problem. We reveal an interesting equivalence to a geometric problem in two dimensional space. On this basis we can then give a fast algorithm for Nash Equilibrium Verification in Section 4.2. The algorithm runs in time $\mathcal{O}(n+m\log m)$. Section 4.3 describes how the algorithm can be extended to verify whether an assignment is at approximate Nash equilibrium.

In Section 4.4 we reduce the *Open Halfspace Emptiness* problem to the Nash Equilibrium Verification problem. With help of Ben-Or's theorem [9] the former can be shown to take at least $\mathcal{O}(m\log m)$ steps in the worst case of any computation that is describable by an algebraic computation tree. This yields a lower bound of $\mathcal{O}(n + m\log m)$ for the Nash Equilibrium Verification problem matching the upper bound of our algorithm.

4.2 A Nash Equilibrium Verification Algorithm

In this section we present an algorithm `IsNash(`I`,` A`)` (Figure 4.2), which outputs $true$, if the pure assignment A is at Nash equilibrium for the instance $I = (\mathbf{w}, bfs)$ of the KP Model with link speeds $\mathbf{s} = (s_1, \ldots, s_m)$ and user flows $\mathbf{w} = (w_1, \ldots, w_n)$, and $false$ otherwise.

The simplest approach to test whether A is at Nash equilibrium, is to check for each user $i \in [n]$, whether it can improve by changing to another link k, i.e., whether

$$\frac{x_k + w_i}{s_k} < \frac{x_{A(i)}}{s_{A(i)}} = \lambda_i(A) \tag{4.1}$$

for some $k \in [m]\backslash\{A(i)\}$. If such user exists, A is not a Nash equilibrium. To verify that no such user exists, we have to check $m - 1$ links for each of n users. This takes time $\Theta(mn)$ in the worst case.

To get better, we make use of the fact that it suffices to verify for each link the user with the smallest traffic. This is due to the following lemma.

Lemma 4.1. *If there is some user i assigned to link j which can improve by moving to some link k, then any user r on link j with $w_r \leq w_i$ can improve by moving to link k.*

Proof. Assume user i is assigned to link j and can improve by moving to link k. Then

$$w_r \leq w_i \quad \Rightarrow \quad \frac{x_k + w_r}{s_k} \leq \frac{x_k + w_i}{s_k} < \frac{x_j}{s_j}\,,$$

which proves the claim. □

Hence, if we initially compute for each link j its congestion x_j and the smallest user flow assigned to link j, which takes time $\mathcal{O}(n + m)$, then we only have to check (4.1) for m users. This takes time $\mathcal{O}(n + m^2)$ altogether.

For the special case of identical link, we only have to check whether a user can improve by changing to the least loaded link, which yields an $\mathcal{O}(n + m)$-time algorithm.

For the special case of identical users (say of identical traffic size w) but related links, we just have to check for each link whether a user can improve by changing to the link j for which $\frac{x_j+w}{s_j}$ is minimal. This can be done in time $\mathcal{O}(m)$ if the assignment is given by the link flows $x_1, \ldots, x_m$. If the assignment is given by A and the flow values must be computed by the algorithm itself, then it takes n steps to go through all users. Hence, the algorithm takes time $\mathcal{O}(n + m)$ in this case.

For the same reason we will never get rid of the additive $\mathcal{O}(n)$ term in the upper bound on the complexity of the general problem. But we will now see how to bring down the running time below $\mathcal{O}(n + m^2)$ in the general case. First note that we only have to check the smallest user on each link (due to Lemma 4.1). Henceforth, let $\bar{w}_j$ be the smallest traffic on link j, and let $l_j = \frac{x_j}{s_j}$ be the latency on link j. All users on link j are satisfied, if and only if there exists no other link with speed s and flow x, such that

$$\begin{aligned} & \frac{x + \bar{w}_j}{s} < l_j \\ \Leftrightarrow \quad & x < l_j s - \bar{w}_j\,. \end{aligned} \tag{4.2}$$

Inequality (4.2) describes a closed halfplane in the two-dimensional sx-coordinate plane with boundary line $x = l_j s - \bar{w}_j$. If we draw for each link j the corresponding line into

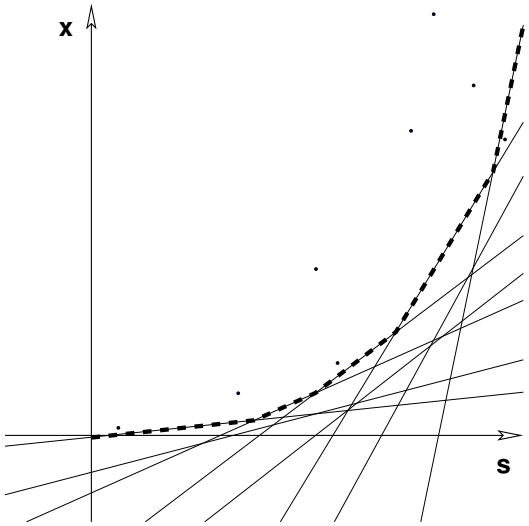

Figure 4.1: Set of lines and points in the plane corresponding to an assignment in the KP model. The dashed course marks the boundary of the intersection of the upper halfplanes of all lines. In a Nash equilibrium all points must lie on or above this course.

the plane, we get a passel of lines like in Figure 4.1. Furthermore, for each link k, (s_k, x_k) can be interpreted as a point in the same plane.

Then a link k with speed s_k and flow x_k fulfilling (4.2) for some $j \in [m]$ and $x = x_k$, $s = s_k$, corresponds to a point (s_k, x_k) lying below line $x = l_j s - \bar{w}_j$. Hence, A is at Nash equilibrium, exactly if none of the points $(s_k, x_k), k = 1, \ldots, m$, lies below any of the lines $x = l_j s - \bar{w}_j, j = 1, \ldots, m$. That is, for A to be a Nash equilibrium, all points must lie in the intersection of the closed upper halfplanes determined by the lines.

It takes time $\mathcal{O}(n+m)$ to compute $x_j, \bar{w}_j$ and l_j for each $j \in [m]$. The intersection of m halfplanes is a convex polygonal region which can be computed in time $\mathcal{O}(m \log m)$ (see e.g. [68]). Checking whether a single point is included in a convex polygonal region with m corners can be done in time $\mathcal{O}(\log m)$ with preprocessing time $\mathcal{O}(m)$ [68]. For a set of m points, as the preprocessing must be carried out only once, the question whether all points are contained in a convex polygonal region can be answered after time $\mathcal{O}(m \log m)$. Thus we have the following $\mathcal{O}(n + m \log m)$ time algorithm for the Nash Equilibrium Verification problem:

1. compute $x_j, \bar{w}_j, l_j$ for all $j \in [m]$
2. compute the intersection Q of the m halfplanes determined by $\{(s,x) \in \mathbb{R}^2 \mid x \geq l_j s - \bar{w}_j\}, j \in [m]$
3. check whether all m points $(s_j, x_j), j \in [m]$, lie in Q

Figure 4.2 states the pseudocode of a more dedicated algorithm. All we need in order to decide whether a point lies in the closed upper halfspace of all lines (see Figure 4.1),

IsNash(I, A)

Input: instance $I = (\mathbf{w}, \mathbf{s})$ of the KP model, assignment A

Output: *true* if A is Nash equilibrium, *false* otherwise

1. for all $j \in [m]$ set $x_j := 0$; and $\bar{w}_j := \infty$;
   for $i = 1, \ldots, n$
   $x_{A(i)} := x_{A(i)} + w_i$;
   $\bar{w}_{A(i)} := \min\{\bar{w}_{A(i)}, w_i\}$;
   for $j = 1, \ldots, m$
   $l_j := x_j / s_j$;

2. sort $((x_j, \bar{w}_j, l_j))_{j \in [m]}$ non-decreasingly according to l_j;
   $a_1 := 0$; $b_1 := -\bar{w}_1$; $i_1 := 1$; $p := 1$;
   for $j = 2, \ldots, m$
   while $l_j a_p - \bar{w}_j \geq l_{i_p} a_p - \bar{w}_{i_p}$ **and** $p > 0$
   $p := p - 1$;
   if $p = 0$ **then**
   $a_1 := 0$; $b_1 := -\bar{w}_j$; $i_1 := j$; $p := 1$;
   else if $l_j > l_{i_p}$ **then**
   $q := \frac{\bar{w}_j - \bar{w}_{i_p}}{l_j - l_{i_p}}$;
   $p := p + 1$;
   $a_p := q$; $b_p := l_j q - \bar{w}_j$; $i_p := j$;

3. sort $((x_j, s_j))_{j \in [p]}$ non-decreasingly according to s_j;
   $k := 1$;
   for $j = 1, \ldots, m$
   while $k < p$ **and** $a_{k+1} \leq s_j$
   $k := k + 1$;
   if $x_j < l_{i_k} s_j - \bar{w}_{i_k}$ **then** return *false*;
   return *true*;

Figure 4.2: Algorithm IsNash(I, A) verifies whether a pure assignment A for an instance I of the congestion routing model with related links (KP model) is at Nash equilibrium or not.

is the course of the lower border of the intersection of all halfplanes. The algorithm computes the corners $(a_0, b_0), \ldots, (a_r, b_r)$ of this course. After the precomputations in step 1, in step 2 we first sort the links according to non-decreasing latencies l_j, i.e., according to the slopes of the lines. Then, we successively add lines according to that order in the for-loop. Variable p stores the number of corner points the course produced so far consists of. $i_1, i_2, \ldots$ are the corresponding line indices, i.e., (a_k, b_k) is the intersection point of line i_{k-1} with line i_k for $k \in [p]\backslash\{1\}$. Assume, (a_p, b_p) is the last corner so far. When a new line is added, we have to check whether its value at position a_p is lower than b_p or not (this is done by the while-condition). In the first case we get a new corner (a_{p+1}, b_{p+1}) (the intersection point of the new line and the last line) which is constructed in the else-statement (the inner if-statement accounts for the special case where the new line and the last line have the same slope). In the second case, as any point below the last line also lies below the new line or some other line processed so far, we may delete the last line and the corresponding corner and repeat the step with $(a_{i-1}, b_{i-1}), (a_{i-2}, b_{i-2}), \ldots$ until the first case applies or all corners have been deleted.

Once we have computed all corners $(a_0, b_0), \ldots, (a_r, b_r)$, we have to check for the points $(s_1, x_1), \ldots, (s_m, x_m)$ whether they lie below the course given by the corners. This is done in the third step. First we sort the points non-decreasingly according to their abscissa s_j. Then we take up all points one after the other. For each point (s_j, x_j) we advance in the list of corners (starting with corner point $k = 1$) until we find the corresponding line segment, i.e., until $a_k < s_j \leq a_{k+1}$ or $k = p$. If (s_j, x_j) lies below the line segment, then A is not at Nash equilibrium. If all points lie above or on their corresponding line segment, A is at Nash equilibrium.

Running Time

The running time for the precomputation step is $\mathcal{O}(n+m)$. The second step is dominated by the sorting. The rest of this step takes time $\mathcal{O}(m)$, because each corner is added to and deleted from the list at most once. Similarly, the running time of $\mathcal{O}(m \log m)$ for the third phase is due to the sorting, whereas the remainder of this phase takes only linear time. Thus, altogether the algorithm runs in time $\mathcal{O}(n + m \log m)$.

Theorem 4.2. *The Nash Equilibrium Verification Problem can be decided in time $\mathcal{O}(n + m \log m)$ for any instance of the KP model with n users and m links..*

4.3 Verification of Approximate Nash Equilibria

There are two established ways to define an approximate Nash equilibrium.

Definition 4.3. *Let A be an assignment for an instance of the KP model with n users and m links. Then A is at ϵ-Nash equilibrium for $\epsilon > 0$, if $\forall i \in [n], j \in [m] : l_i(A \,|\, j) + \epsilon \geq l_i(A)$.*

Definition 4.4. *Let A be an assignment for an instance of the KP model with n users and m links. Then A is at $(1+\epsilon)$-approximate Nash equilibrium for $\epsilon > 0$, if $\forall i \in [n], j \in [m] : (1+\epsilon) l_i(A \,|\, j) \geq l_i(A)$.*

According to Definition 4.3 an assignment is at ϵ-approximate Nash equilibrium if no user can decrease its cost by more than ϵ by changing to another link. According to Definition 4.4 an assignment is at $(1+\epsilon)$-approximate Nash equilibrium if no user can decrease its cost by a factor of $\frac{1}{1+\epsilon}$ or below. Both kinds of approximate Nash equilibria give way to exact Nash equilibria if ϵ is zero.

Our algorithm for the Nash Equilibrium Verification problem can be used to check whether an assignment is at approximate Nash equilibrium as well, if slight modifications are being undertaken.

First note that Lemma 4.1 is transferable to the approximate case. That is, if a user, assigned to some link, can decrease its experienced latency by at least ϵ, or, respectively, by at least a factor of $\frac{1}{1+\epsilon}$, when switching to another link, then the same holds for any user with smaller or equal flow that is currently assigned to the same link. Hence, as in the original algorithm, we have to check the Nash equilibrium condition only once per link, namely for the smallest flow.

According to Definition 4.3 the ϵ-approximate Nash equilibrium condition for link j is not fulfilled if there exists another link with speed s and flow x so that

$$\begin{aligned} & \frac{x+\bar{w}_j}{s} + \epsilon < l_j \\ \Leftrightarrow \quad & x < (l_j - \epsilon)s - \bar{w}_j \,. \end{aligned} \tag{4.3}$$

Again we can interpret this inequality geometrically. This yields for each link a line and a point in the sx-plane. The only difference to the verification problem for exact Nash equilibria is that the line for link j has slope $(l_j - \epsilon)$ now instead of l_j. Similarly, on the basis of Definition 4.4, the slope is $\frac{1}{1+\epsilon} l_j$. With these modifications we can adopt our argumentation from the last section. So, we only have to change the last line in the first step of the algorithm in Figure 4.2. We compute l_j as $l_j := \frac{x_j}{s_j} - \epsilon$ or $l_j := \frac{x_j}{(1+\epsilon)s_j}$ to obtain an algorithm for the ϵ-approximate Nash Equilibrium Verification Problem or for the $(1+\epsilon)$-approximate Nash Equilibrium Verification Problem, respectively. Note that in the first case the slopes of some lines might be negative. Those lines can never cause the instance to be not at Nash equilibrium, but processing them does not derange the algorithm nor does it affect its worst case running time.

Theorem 4.5. *The ϵ-approximate Nash Equilibrium Verification Problem and the $(1+\epsilon)$-approximate Nash Equilibrium Verification Problem can be decided in time $\mathcal{O}(n + m \log m)$ for any instance of the KP model with n users and m links..*

4.4 A Lower Bound for the Nash Equilibrium Verification Problem

In this section we prove a lower bound of $\Omega(n + m \log m)$ for the problem to decide whether a given assignment of n users to m links in the KP model is at Nash equilibrium or not. This lower bound holds for any algorithm whose computation can be described by an *algebraic computation tree* [9].

Algebraic Computation Trees
An algebraic computation tree T is a binary tree with three kinds of nodes: *computation nodes*, *branch nodes* and *output nodes*. Computation nodes have exactly one child node, branch nodes have two child nodes and the output nodes are exactly the leaves of the tree. An input for the algebraic computation tree is given by a vector $\mathbf{x} \in \mathbb{R}^n$ of real numbers. A function is associated with the tree that assigns

- to each computation node v an instruction $f_v = f_u \circ f_w$, $f_v = c \circ f_u$ or $f_v = \sqrt{f_u}$ which assigns the value f_v to v where $\circ$ is one of the operations $+, -, \cdot, /$ and $f_u, f_w \in \mathbb{R}$ are input values or values associated with computation nodes u, w that are ancestors of v and $c \in \mathbb{R}$ is a constant,
- to each branch node v a test instruction $f_u > 0$, $f_u \geq 0$ or $f_u = 0$, where $f_u \in \mathbb{R}$ is an input value or the value associated with an ancestor u of v,
- to each output node Yes or No

An algebraic computation tree assigns either Yes or No to each of its inputs $\mathbf{x} \in \mathbb{R}^n$ by traversing a path in the tree. At each computation node the associated operation is performed. At each branch node the path branches either to the left or right child according to the associated test. At a leaf node either Yes or No is reported. An algebraic computation tree T is said to decide the membership for a set $W \subseteq \mathbb{R}^n$, if W is the set of Yes-instances of T.

The following theorem of Ben-Or is the basis for our lower bound proof by counting the number of connected components in the subspace of Yes-instances.

Theorem 4.6. *([9]) Let $W \subseteq \mathbb{R}^n$ be any set, and let T be an algebraic computation tree that solves the membership problem for W. If N is the number of disjoint connected components of W and h is the height of T, then $2^h 3^{n+h} \geq N$.*

Algorithms that can be described by algebraic computation trees correspond to programs that can be carried out by a *realRAM* (see [68]), which is a widely used model of computation, particularly in the area of computational geometry. A realRAM is similar to an ordinary RAM [1] but can operate on real numbers (i.e., with infinite precision) in constant time.

To achieve a lower bound on the Nash Equilibrium Verification Problem we will bound the complexity of another algebraic problem from below and then reduce it in a chain of reductions to the Nash Equilibrium Verification Problem. The proof involves the following five problems:

P_1 Given $n \in \mathbb{N}$ and $3n$ numbers $a_1, \ldots, a_n, b_1, \ldots, b_n, c_1, \ldots, c_n \in \mathbb{R}$, is for all $i, j \in [n] : (c_i \notin (a_j, b_j) \wedge a_j < b_j)$?

P_2 Given $n \in \mathbb{N}$ closed halfplanes and n points in the plane, do all points lie in the intersection of all halfplanes?

P_3 Given $n \in \mathbb{N}$ lines in the xy-plane with positive slopes intersecting the negative y-axis and n points in the open first quadrant (i.e., with positive coordinates), does no point lie below any line?

P_4 Given $n \in \mathbb{N}$ lines in the xy-plane with positive slopes $a_1, \ldots, a_n \in \mathbb{R}_{>0}$ and negative y-axis intercepts $b_1, \ldots, b_n \in \mathbb{R}_{<0}$ (i.e., $(0, b_i)$ is the intersection point of line i with the y-axis), does none of the points $(\frac{-b_1}{a_1}, -b_1), \ldots, (\frac{-b_n}{a_n}, -b_n) \in \mathbb{R}^2_{>0}$ lie below any line?

P_5 Given $n \in \mathbb{N}$ links with speeds $s_1, \ldots, s_n \in \mathbb{R}_{>0}$, congestions $x_1, \ldots, x_n \in \mathbb{R}_{>0}$ and smallest flows $\bar{w}_1, \ldots, \bar{w}_n \in \mathbb{R}_{>0}$, is this situation at Nash equilibrium?

Note that, different from our former convention, n denotes the number of links in the description of problem P_5. We will prove that

$$P_1 \leq^{\mathcal{O}(n)}_{const-T} P_2 \leq^{\mathcal{O}(n)}_{const-T} P_3 \leq^{\mathcal{O}(n)}_{const-T} P_4 \leq^{\mathcal{O}(n)}_{const-T} P_5\,,$$

where $A \leq^{\mathcal{O}(n)}_{const-T} B$ means that problem A is Turing-reducible to problem B via some deterministic linear time function that makes at most a constant number of queries to B. We will give then a lower bound for the computational complexity of P_1 in the algebraic computation tree model, which therefore applies to P_5, which is essentially the Nash Equilibrium Verification Problem, as well.
Let us start with the last step of the reduction chain.

Lemma 4.7. $P_4 \leq^{\mathcal{O}(n)}_{const-T} P_5$.

Proof. Let $a_1, \ldots, a_n \in \mathbb{R}_{>0}, b_1, \ldots, b_n \in \mathbb{R}_{<0}$ be an input to Problem P_4. Then set the input of the P_5 problem to an instance of the KP model with n links of speeds $s_1, \ldots, s_n$, congestions $x_1, \ldots, x_n$ and smallest traffic sizes $\bar{w}_1, \ldots, \bar{w}_n$. We obtain the link speeds by $s_i = \frac{-b_i}{a_i}$ and the congestions and smallest flows by $x_i = \bar{w}_i = -b_i$ for all $i \in [n]$. This corresponds to an assignment of n users with weights $w_1 = -b_1, \ldots, w_n = -b_n$, where user i is assigned to link i solely for all $i \in [n]$. Then, for any $i, j \in [n]$,

$$-b_i < \frac{-b_i}{a_i} a_j + b_j \Leftrightarrow \frac{w_i + w_j}{s_i} < \frac{w_j}{s_j}\,.$$

Note that $-b_i$ is positive. The left inequality formalizes the fact that the ith point (which has coordinates $(\frac{-b_i}{a_i}, -b_i)$) lies below the jth line (with slope a_j and intersection with the y-axis at b_j) referring to the input of P_4. The second inequality stands for the fact that, referring to the P_5-input, the only user on link j can improve by switching to link i, and hence the situation is not at Nash equilibrium. Due to the above equivalence, the P_4 instance is positive, exactly if the P_5 instance we obtained from it is positive. □

Remark 4.8. *Note that in the proof of Lemma 4.7 problem P_4 is transformed to a special case of P_5 where each link carries exactly one user's flow. One can formulate a more general version P_4' of P_4 such that there is a bijective transformation between the instances of P_4' and P_5.*

P_4' *Given $n \in \mathbb{N}$ lines with positive slopes $a_1, \ldots, a_n \in \mathbb{R}_{>0}$ and y-axis intercepts $b_1, \ldots, b_n \in \mathbb{R}_{<0}$ and n abscissas $z_1, \ldots, z_n \in \mathbb{R}_{>0}$ with $z_i = \frac{-b_i}{a_i}$ or $z_i \geq 2\frac{-b_i}{a_i}$ for all $i \in [n]$, does none of the points $(z_1, a_1 z_1), \ldots, (z_n, a_n z_n)$ lie below any line?*

The constraints on $z_1, \ldots, z_n$ are due to the fact that in the P_5 instance the smallest traffic size of a link can either be equal to the link's congestion (if there is at most one user on that link) or must be lower than or equal to half the congestion (if there are at least two users on that link).

Showing $P_3 \leq^{\mathcal{O}(n)}_{const-T} P_4$ is the main step of the reduction chain. We first give a description of the transformation and the main idea. Then a formal proof follows.

Let the input to P_3 be given by $a_1, \ldots, a_n \in \mathbb{R}_{>0}$, the slopes of the n lines, $b_1, \ldots, b_n \in \mathbb{R}_{<0}$, their y-axis intercepts (i.e., $(0, b_i)$ is the intersection point of line i with the y-axis) and $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^2_{>0}$, the n points.

Problem P_3 is similar to P_4 in that both problems question whether any of n points lies below any of n lines. But for problem P_3 both, lines and points, are part of the input. The lines may have arbitrary positive slope and negative y-axis intercept. The points may have arbitrary positive coordinates. Hence we have $4n$ degrees of freedom. For P_4 only the lines are part of the input, yielding only $2n$ degrees of freedom. Thus, in order to cope with this non-conformity, we map the instance of P_3 to an instance of P_4 with $2n$ lines instead of only n lines. Lines $1, \ldots, n$ will be the same as the n lines of the P_3-instance. Lines $n+1, \ldots, 2n$ will be determined so that their corresponding points match the points of the P_3-instance. Denote the slopes of the lines of the P_4 instance as $\hat{a}_1, \ldots, \hat{a}_{2n}$ and their y-axis intercepts by $\hat{b}_1, \ldots, \hat{b}_{2n}$. Hence we set

$$\hat{a}_i = \begin{cases} a_i & \text{for } i \in [n] \\ y_{i-n}/x_{i-n} & \text{for } i \in [2n]\backslash[n] \end{cases} \text{ and } \hat{b}_i = \begin{cases} b_i & \text{for } i \in [n] \\ -y_{i-n} & \text{for } i \in [2n]\backslash[n] \end{cases}. \tag{4.4}$$

Let $(\hat{x}_1, \hat{y}_1), \ldots, (\hat{x}_{2n}, \hat{y}_{2n})$ be the points yielded by the lines we just defined, i.e., $\hat{x}_i = -\hat{b}_i/\hat{a}_i$ and $\hat{y}_i = -\hat{b}_i$ for $i \in [2n]$. Note that lines $1, \ldots, n$ of the P_4 instance and points $(\hat{x}_{n+1}, \hat{y}_{n+1}), \ldots, (\hat{x}_{2n}, \hat{y}_{2n})$ correspond to the lines and points of the P_3 instance. Let us call these lines and points *deliberate lines* and *deliberate points*, respectively.

If one of the points lies below one of the lines for the P_3 instance, then the same is true for the corresponding deliberate line and point of the P_4 instance. But, unfortunately, one of the points $(\hat{x}_1, \hat{y}_1), \ldots, (\hat{x}_n, \hat{y}_n)$ or one of the lines $n+1, \ldots, 2n$ of the P_4 instance could also cause the instance to be negative. Let us call these points and lines *artefactual points* and *artefactual lines*, respectively. Thus, we cannot guarantee that the P_4 instance is positive if the P_3 instance is positive.

To overcome this problem, we shift all lines and points of the original P_3 instance by an appropriate number $\delta \in \mathbb{R}$ of units in the direction of the positive x-axis. Denote the modified input to P_3 by $a'_1, \ldots, a'_n \in \mathbb{R}_{>0}$ (the slopes of the lines), $b'_1, \ldots, b'_n \in \mathbb{R}_{<0}$ (their y-axis intercepts) and $(x'_1, y'_1), \ldots, (x'_n, y'_n) \in \mathbb{R}^2_{>0}$ (the points). Shifting a line does not change its slope but only its intersection point with the y-axis. Shifting a point only increases its x-coordinate. That is, we have $a'_i = a_i, b'_i = b_i - \delta a_i, x'_i = x_i + \delta$ and $y'_i = y_i$ for all $i \in [n]$. The P_4-instance is now defined according to the shifted P_3-instance. That is,

$$\hat{a}_i = \begin{cases} a'_i & \text{for } i \in [n] \\ y'_{i-n}/x'_{i-n} & \text{for } i \in [2n]\backslash[n] \end{cases} \quad \text{and} \quad \hat{b}_i = \begin{cases} b'_i & \text{for } i \in [n] \\ -y'_{i-n} & \text{for } i \in [2n]\backslash[n] \end{cases} . \tag{4.5}$$

Shifting all lines and points of the P_3 instance by the same amount does not affect the question whether the instance is positive or not since it does not change the relative position of lines and points. But, as we will see, it changes the artefactual lines and points of the corresponding P_4 instance that we get by the transformation as described above. If the amount of movement is sufficiently large, then the artefactual lines and points cannot cause the instance to be negative.

Let us assume the P_4 instance that we obtained by transforming the modified (after shifting) P_3 instance is negative. Then there exist $i, j \in [2n]$ such that point i lies below line j, i.e.,

$$\begin{aligned} & \hat{y}_i < \hat{a}_j \hat{x}_i + \hat{b}_j \\ \Leftrightarrow \qquad & -\hat{b}_i < \frac{-\hat{a}_j \hat{b}_i}{\hat{a}_i} + \hat{b}_j \, . \end{aligned} \tag{4.6}$$

Subject to i and j we can distinguish four cases.

$$\begin{cases} \text{artefactual point } i \text{ lies below deliberate line } j & \text{if } i, j \in [n] \\ \text{artefactual point } i \text{ lies below artefactual line } j & \text{if } i \in [n], j \in [2n]\backslash[n] \\ \text{deliberate point } i \text{ lies below deliberate line } j & \text{if } i \in [2n]\backslash[n], j \in [n] \\ \text{deliberate point } i \text{ lies below artefactual line } j & \text{if } i, j \in [2n]\backslash[n] \end{cases}$$

We get the corresponding inequalities, if we apply the definition of $\hat{a}_i$ and $\hat{b}_i$ in (4.5) to

(4.6).

$$
\begin{aligned}
&-\hat{b}_i < -\frac{\hat{a}_j\hat{b}_i}{\hat{a}_i} + \hat{b}_j\,.\\
\Leftrightarrow &\begin{cases}
-b'_i < -\frac{a'_j b'_i}{a'_i} + b'_j & \text{if } i,j \in [n]\\
-b'_i < -\frac{y'_{j-n} b'_i}{x'_{j-n} a'_i} - y'_{j-n} & \text{if } i \in [n], j \in [2n]\backslash[n]\\
y'_{i-n} < \frac{a'_j y'_{i-n} x'_{i-n}}{y'_{i-n}} + b'_j & \text{if } i \in [2n]\backslash[n], j \in [n]\\
y'_{i-n} < \frac{y'_{j-n} y'_{i-n} x'_{i-n}}{x'_{j-n} y'_{i-n}} - y'_{j-n} & \text{if } i,j \in [2n]\backslash[n]
\end{cases}\\
\Leftrightarrow &\begin{cases}
-(b_i - \delta a_i) < -\frac{a_j(b_i-\delta a_i)}{a_i} + (b_j - \delta a_j) & \text{if } i,j \in [n]\\
-(b_i - \delta a_i) < -\frac{y_{j-n}(b_i-\delta a_i)}{(x_{j-n}+\delta)a_i} - y_{j-n} & \text{if } i \in [n], j \in [2n]\backslash[n]\\
y_{i-n} < a_j(x_{i-n}+\delta) + (b_j - \delta a_j) & \text{if } i \in [2n]\backslash[n], j \in [n]\\
y_{i-n} < \frac{y_{j-n}(x_{i-n}+\delta)}{x_{j-n}+\delta} - y_{j-n} & \text{if } i,j \in [2n]\backslash[n]
\end{cases}\\
\Leftrightarrow &\begin{cases}
\delta a_i - b_i < -\frac{a_j}{a_i} b_i + b_j & \text{if } i,j \in [n]\\
(\delta a_i - b_i)(x_{j-n}+\delta) < -y_{j-n}(\frac{b_i}{a_i} + x_{j-n}) & \text{if } i \in [n], j \in [2n]\backslash[n]\\
y_{i-n} < a_j x_{i-n} + b_j & \text{if } i \in [2n]\backslash[n], j \in [n]\\
x_{j-n} + \delta < \frac{y_{j-n}}{y_{i-n}}(x_{i-n} - x_{j-n}) & \text{if } i,j \in [2n]\backslash[n]
\end{cases}
\end{aligned}
$$

Note that the third case is equivalent to the fact that point $i-n$ lies below line j in the original P_3 instance (before shifting). Furthermore, the other three cases cannot appear if we choose δ sufficiently large, as in all cases the left hand side of the inequality is increasing with δ and the right hand side does not depend on δ. If $\delta \geq \frac{1}{a_i}\left(-\frac{a_j b_i}{a_i} + b_j + b_i\right)$ for $i,j \in [n]$, then the first case cannot appear. We may estimate more roughly and will choose δ such that $\delta \geq -\frac{a_j b_i}{a_i^2}$ for $i,j \in [n]$. Similarly, we can avoid the second and fourth case. If we ensure that $\delta \geq 1$, then $\delta a_i - b_i \geq -\frac{y_{j-n} b_i}{a_i} - y_{j-n}x_{j-n}$ for $i \in [n], j \in [2n]\backslash[n]$ would suffice to make the second case impossible. Recall that $-b_i$ is positive. Consequently, if we set $\delta \geq \max\{1, -\frac{y_{j-n} b_i}{a_i^2}\}$ for $i \in [n], j \in [2n]\backslash[n]$, then the second case cannot appear. If $\delta \geq \frac{y_{j-n}x_{i-n}}{y_{i-n}}$ for $i,j \in [2n]\backslash[n]$, then the fourth case cannot appear. Hence, if we set δ such that $\delta \geq \max\{-\frac{a_j b_i}{a_i^2}, 1, -\frac{y_j b_i}{a_i^2}, \frac{y_j x_i}{y_i}\}$ for all $i,j \in [n]$, then we avoid all cases that involve artefactual lines or points.

Figure 4.3 exemplifies the fourth case. On the left we see two deliberate points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ of the P_4 instance that has been obtained from the P_3 instance without shifting. l_1 and l_2 are the corresponding lines with slopes $a_1 = \frac{y_1}{x_1}$, $a_2 = \frac{y_2}{x_2}$ and vertical axis intercepts $b_1 = -y_1$, $b_2 = -y_2$. As p_2 lies below l_1, the P_4 instance is negative although the P_3 instance may be positive. If all points and lines of the P_3 instance are shifted by δ to the right before transforming it to the P_4 instance, then we obtain points $p'_1 = (x_1 + \delta, y_1)$ and $p'_2 = (x_2 + \delta, y_2)$ and lines l'_1 and l'_2 with slopes $a'_1 = \frac{y_1}{x_1+\delta}$ and

$a'_2 = \frac{y_2}{x_2+\delta}$ in place of points p_1, p_2 and lines l_1, l_2. The axis intercepts do not change. This is illustrated at the right side of Figure 4.3. Now p'_2 lies above l'_1, as desired.

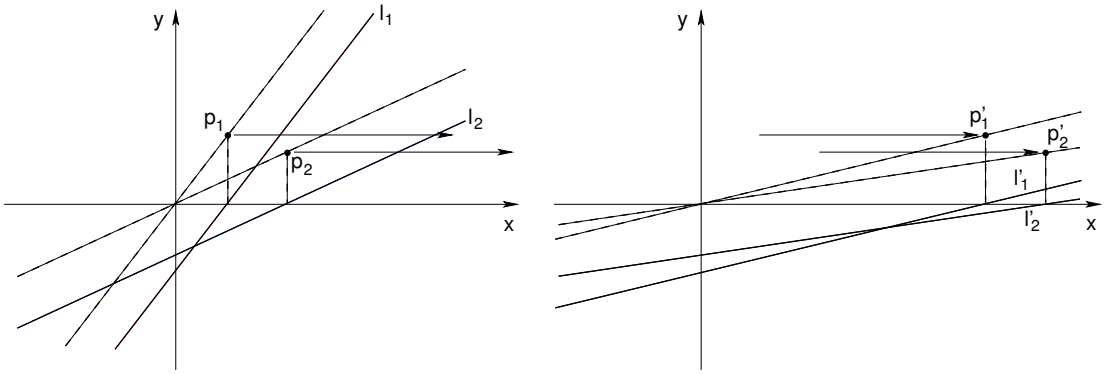

Figure 4.3: Left: Deliberate point p_2 lies below artefactual line l_1. Right: After shifting p'_2 lies above l'_1.

To determine δ in linear time we estimate the terms and set

$$\delta = \max\{-\frac{a_{max}b_{min}}{a^2_{min}}, 1, -\frac{y_{max}b_{min}}{a^2_{min}}, \frac{y_{max}x_{max}}{y_{min}}\} \tag{4.7}$$

where

$$\begin{aligned} a_{max} &= \max_{i\in[n]} a_i, & a_{min} &= \min_{i\in[n]} a_i, \\ y_{max} &= \max_{i\in[n]} y_i, & y_{min} &= \min_{i\in[n]} y_i, \\ b_{min} &= \min_{i\in[n]} b_i, & x_{max} &= \max_{i\in[n]} x_i\,. \end{aligned} \tag{4.8}$$

Let us state all the above in the following lemma.

Lemma 4.9. $P_3 \leq^{\mathcal{O}(n)}_{const-T} P_4$.

Proof. Let $I = (a_1, \ldots, a_n, b_1, \ldots, b_n, x_1, \ldots, x_n, y_1, \ldots, y_n)$ be an instance of problem P_3. Set δ according to (4.7) and (4.8). For $i \in [2n]$ set

$$\hat{a}_i = \begin{cases} a_i & i \in [n] \\ y_{i-n}/(x_{i-n}+\delta) & i \in [2n]\backslash[n] \end{cases} \text{ and } \hat{b}_i = \begin{cases} b_i - \delta a_i & i \in [n] \\ -y_{i-n} & i \in [2n]\backslash[n] \end{cases}.$$

Then $\hat{I} = (\hat{a}_1, \ldots, \hat{a}_{2n}, \hat{b}_1, \ldots, \hat{b}_{2n})$ is an instance of problem P_4.
We will show that $\hat{I}$ is positive for P_4, if and only if I is positive for P_3.
Assume first that I is negative. Then there exist $i, j \in [n]$ such that the ith point lies below

the jth line of I. That is,

$$\begin{aligned}
& y_i < a_j x_i + b_j \\
\Leftrightarrow \quad & y_i < \frac{a_j y_i (x_i + \delta)}{y_i} + (b_j - \delta a_j) \\
\Leftrightarrow \quad & -\hat{b}_{i+n} < -\hat{a}_j \frac{\hat{b}_{i+n}}{\hat{a}_{i+n}} + \hat{b}_j \\
\Leftrightarrow \quad & \hat{y}_{i+n} < \hat{a}_j \hat{x}_{i+n} + \hat{b_j}
\end{aligned}$$

The last inequality implies that point $(-\frac{\hat{b}_{i+n}}{\hat{a}_{i+n}}, -\hat{b}_{i+n})$ lies below the jth line, and hence $\hat{I}$ must be negative, too.
Now assume that $\hat{I}$ is negative. Then there exist $i, j \in [2n]$ such that the point $(-\frac{\hat{b}_i}{\hat{a}_i}, -\hat{b}_i)$ lies below the jth line of $\hat{I}$. That is,

$$\hat{b}_i < \hat{a}_j \left(-\frac{\hat{b}_i}{\hat{a}_i} \right) + \hat{b}_j$$

$$\Leftrightarrow \begin{cases}
-(b_i - \delta a_i) < -\frac{a_j (b_i - \delta a_i)}{a_i} + (b_j - \delta a_j) & \text{if } i, j \in [n] \\
-(b_i - \delta a_i) < -\frac{y_{j-n}(b_i - \delta a_i)}{(x_{j-n} + \delta) a_i} - y_{j-n} & \text{if } i \in [n], j \in [2n] \backslash [n] \\
y_{i-n} < a_j (x_{i-n} + \delta) + (b_j - \delta a_j) & \text{if } i \in [2n] \backslash [n], j \in [n] \\
y_{i-n} < \frac{y_{j-n}(x_{i-n} + \delta)}{x_{j-n} + \delta} - y_{j-n} & \text{if } i, j \in [2n] \backslash [n]
\end{cases}$$

$$\Leftrightarrow \begin{cases}
\delta a_i^2 < -a_j b_i + a_i b_j + a_i b_i & \text{if } i, j \in [n] \\
(\delta a_i - b_i)(x_{j-n} + \delta) a_i < -y_{j-n} b_i - a_i x_{j-n} y_{j-n} & \text{if } i \in [n], j \in [2n] \backslash [n] \\
y_{i-n} < a_j x_{i-n} + b_j & \text{if } i \in [2n] \backslash [n], j \in [n] \\
y_{i-n}(x_{j-n} + \delta) < y_{j-n} x_{i-n} - x_{j-n} y_{j-n} & \text{if } i, j \in [2n] \backslash [n]
\end{cases}$$

$$\Rightarrow \begin{cases}
\delta < -\frac{a_j}{a_i^2} b_i & \text{if } i, j \in [n] \\
\delta^2 < -\frac{y_{j-n} b_i}{a_i^2} & \text{if } i \in [n], j \in [2n] \backslash [n] \\
y_{i-n} < a_j x_{i-n} + b_j & \text{if } i \in [2n] \backslash [n], j \in [n] \\
\delta < \frac{y_{j-n} x_{i-n}}{y_{i-n}} & \text{if } i, j \in [2n] \backslash [n]
\end{cases}$$

$$\Rightarrow \begin{cases}
\delta < -\frac{a_{max} b_{min}}{a_{min}^2} & \text{if } i, j \in [n] \\
\delta < 1 \vee \delta < -\frac{y_{max} b_{min}}{a_{min}^2} & \text{if } i \in [n], j \in [2n] \backslash [n] \\
\text{I is negative for } P_3 & \text{if } i \in [2n] \backslash [n], j \in [n] \\
\delta < \frac{y_{max} x_{max}}{y_{min}} & \text{if } i, j \in [2n] \backslash [n]
\end{cases}.$$

As the first, second and fourth case contradict (4.7), this implies that I is negative. □

We continue with the next step of the reduction chain. On first sight problem P_2 seems to be more general than P_3, as the constraints on the input for P_3 are more restrictive. But,

as we will see in the proof of the next lemma, we can decide P_2 by deciding four instances of P_3.

Lemma 4.10. $P_2 \leq^{\mathcal{O}(n)}_{const-T} P_3$.

Proof. Let I be an input instance for P_2. Assume the n halfplanes are given by base points $q_1, \ldots, q_n \in \mathbb{R}^2$ and normal vectors $n_1, \ldots, n_n \in \mathbb{R}^2 \backslash \{(0,0)\}$, where q_i lies on the boundary of halfplane i and n_i is directed into the interior of halfplane i. That is, halfplane i is defined by the set of points $\{p \in \mathbb{R}^2 | (p - q_i) \bullet n_i \geq 0\}$, where '$\bullet$' denotes the inner product. Let the input points be given by $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^2$. First, process all halfplanes with axis parallel boundary line (or, equivalently, with axis parallel normal vector). Therefor, sweep over the halfplanes once and determine the intersection area of all halfplanes with horizontal or vertical boundary line and, at the same time, remove all those halfplanes. Note that the intersection area is an axis parallel rectangle. Now check for each point whether it lies in this rectangle. If not, then we may report that the given P_2 instance is negative. Otherwise, partition the remaining halfplanes into four groups according to their normal vector. For $j \in \{1, 2, 3, 4\}$ group j contains all halfplanes with normal vectors lying in the jth quadrant of the coordinate plane. A vector $v = (x, y)$ lies in the

- 1st quadrant, if $0 < x, y$,
- 2nd quadrant, if $x < 0 < y$,
- 3rd quadrant, if $x, y < 0$,
- 4th quadrant, if $x > 0 > y$.

Note that $x, y \neq 0$ applies to all normal vectors, as halfplanes with axis parallel normal vectors have been removed. Hence the quadrant of each remaining normal vector is well defined.

For $j \in [4]$ we define the function $f_j : \mathbb{R}^2 \to \mathbb{R}^2$ by

$$f_j((x,y)) = \begin{cases} (-y, x) & \text{for } j = 1 \\ (x, y) & \text{for } j = 2 \\ (y, -x) & \text{for } j = 3 \\ (-y, -y) & \text{for } j = 4 \end{cases}.$$

For each group j of halfplanes, we now construct a new instance I'_j consisting of n points and m_j halfplanes, where m_j is the number of halfplanes in group j. For I'_j we obtain the n points by applying f_j to each input point of the P_2 instance. The m_j halfplanes are obtained by applying f_j to both, the base point and the normal vector of each halfplane in group j. Note that all normal vectors of I'_1, I'_2, I'_3, I'_4 lie in the 2nd quadrant. Furthermore, we did not change the relative position of the points and the halfplanes that belong to the same instance I'_j, as the transformation affects all coordinates similarly. Thus, the original

instance I is positive exactly if I'_j is positive for all $j \in \{1,2,3,4\}$.

For each $j \in \{1,2,3,4\}$ we now obtain from I'_j an instance I''_j. For each halfplane of I'_j, given by its base point and its normal vector, we compute the slope and vertical axis intercept of its boundary line. As all normal vectors lie in the 2nd quadrant we obtain positive slopes only. Now we shift all lines and points in positive x-axis direction such that afterwards all of the n points have a positive x-coordinate. The amount by which we have to shift can be computed by seeking the minimal x-coordinate of any point and adding an arbitrary positive constant. After that we compute the minimal y-coordinate y_{min} of a point and the maximal y-axis interception y_{max} of a line in the modified (by the shift in positive x-axis direction) instance we obtained from I'_j. If $y_{max} \geq y_{min}$, then some point lies below some line and we may report that the instance is negative immediately. Otherwise, we shift all lines and points by $(y_{max} + y_{min})/2$ in negative y-axis direction. Both steps do not affect the relative position of the lines and points, but now all lines have negative y-axis intercepts and all points have positive coordinates.

As the number of lines in I'_j, m_j, might be lower than the number of points, n, the last step is to copy one of the lines $n - m_j$ times. Let us denote the instance we obtained from I'_j in this way by I''_j. Then, for $j \in \{1,2,3,4\}$, I''_j is an instance for P_3, and I, the original instance for P_2, is positive, exactly if for all $j \in \{1,2,3,4\}$ I''_j is positive for P_3. □

Problem P_2 is very similar to the *Halfspace Emptiness Problem* for the two dimensional case, which is well known in the area of computational geometry (see e.g. [22]). In both cases a set of points and a set of halfplanes is given. P_2 asks whether every point lies in every halfplane. The halfspace emptiness problem asks whether all halfplanes are empty, which is exactly the case if every point lies in the complement of every halfplane. But as the halfplanes are closed, their complements are open halfplanes, that is, they do not contain their boundaries. In order to get closed halfplanes, open halfplanes are required as input. Hence, one could consider P_2 as the two dimensional *Open Halfspace Emptiness Problem* which asks whether, given a set of open halfplanes and a set of points, all halfplanes are empty. The following lemma shows that we can reduce P_1, a combinatorial problem, to P_2.

Lemma 4.11. $P_1 \leq^{\mathcal{O}(n)}_{const-T} P_2$.

Proof. Let $a_1, \ldots, a_n, b_1, \ldots, b_n, c_1, \ldots, c_n \in \mathbb{R}$ be an input to P_1. If for some $i \in [n]$ $a_i \geq b_i$, then the instance is negative. Otherwise, define $x : \mathbb{R} \to \mathbb{R}$ and $y : \mathbb{R} \to \mathbb{R}$ by $x(r) = \frac{1-r^2}{1+r^2}$ and $y(r) = \frac{2r}{1+r^2}$, and set $g_i = (x(a_i), y(a_i))$, $h_i = (x(b_i), y(b_i))$ and $p_i = (x(c_i), y(c_i))$ for all $i \in [n]$. We now construct an input to P_2. $p_1, \ldots, p_n$ are the points of the input to P_2. The halfplanes of the P_2 input are given by their base points $q_1 = (g_1 + h_1)/2, \ldots, q_n = (g_n + h_n)/2$ and normal vectors $n_1 = -q_1, \ldots, n_n = -q_n$. Figure 4.4 shows a plot of the two functions used to map a real number of the P_1 instance to the horizontal and vertical coordinate of a point in the plane. As $(x(r))^2 + (y(r))^2 = 1$ for any $r \in \mathbb{R}$, the set of real numbers is mapped to the unit circle in the plane with its center at the origin. Figure 4.5 (left) illustrates this mapping, where the set of real numbers is shown as the vertical line $((1, r))_{r \in \mathbb{R}}$. A number c_j of the P_1 instance translates to a point

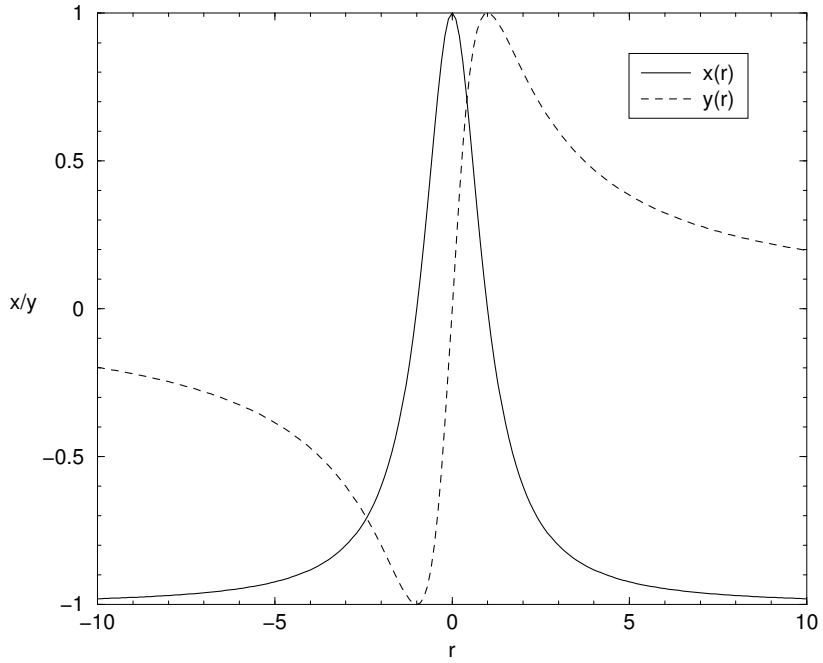

Figure 4.4: $x(r) = \frac{1-r^2}{1+r^2}$ and $y(r) = \frac{2r}{1+r^2}$

of the P_2 instance that lies on the unit circle and an interval of the P_1 instance translates to a halfplane of the P_2 instance whose boundary line is a secant of the unit circle (see Figure 4.5 (right)). If a number $c_i, i \in [n]$, is contained in an interval $(a_j, b_j), j \in [n]$, in the P_1-instance, then point p_i lies out of halfplane j in the P_2-instance, and vice versa. That is, the P_2-instance is positive, exactly if the P_1 instance is positive. □

Finally, we bound the complexity of problem P_1 from below using Ben-Or's theorem.

Lemma 4.12. *Every algebraic computation tree that decides P_1-instances with input size n, has depth at least $\Omega(n \log n)$.*

Proof. Let $W = \{I = (a_1, \ldots, a_n, b_1, \ldots, b_n, c_1, \ldots, c_n) \in \mathbb{R}^{3n} | I \text{ is positive for } P_1\}$ be the set of positive P_1 instances.
The instance with $a_i = i + 0.25, b_i = i + 0.75$ and $c_i = \sigma(i)$ for $i \in [n]$ is positive for any permutation σ on $[n]$. There are $n!$ permutations which yield $n!$ distinct inputs to P_1. All of them lie in different connected components of W, because there is no way to get from one to the other by continuously mutating the input vector without obtaining a negative instance. Thus, the number of disjoint connected components of W is at least $n!$. The theorem of Ben-Or (see Theorem 4.6) implies that for the height h of any algebraic

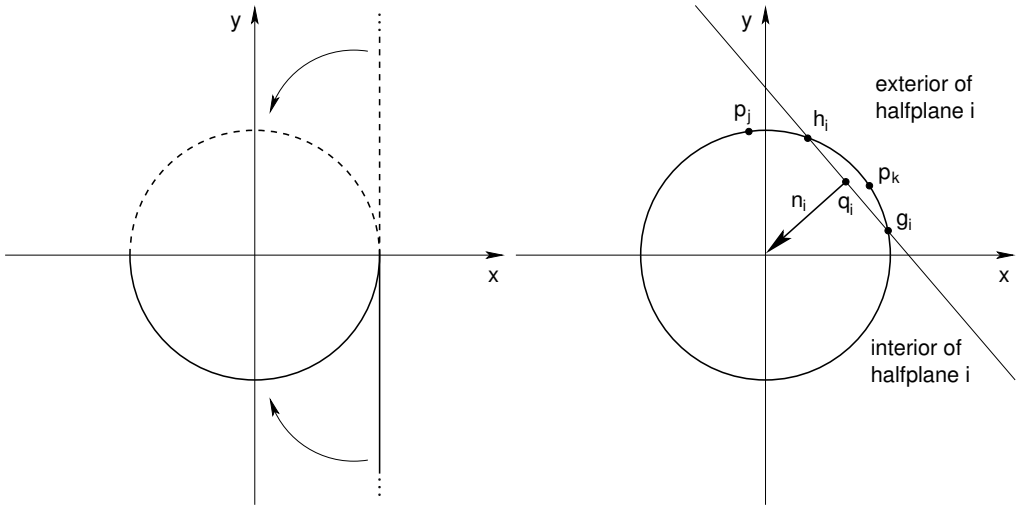

Figure 4.5: Left: Mapping the real numbers to the unit circle. Right: An interval is transformed to a halfplane.

computation tree deciding P_1 we have

$$\begin{aligned} & 2^h 3^{n+h} \geq n! \\ \Rightarrow \quad & h \log 2 + (n+h) \log 3 \geq \log n! \\ \Rightarrow \quad & h \geq \frac{\log n! - n \log 3}{\log 2 + \log 3} = \Omega(\log n! - n) = \Omega(n \log n)\,. \end{aligned}$$

□

Theorem 4.13. *Every algebraic computation for the Nash Equilibrium Verification Problem takes at least $\Omega(n + m \log m)$ steps in the worst case, where n is the number of users and m is the number of links.*

Proof. The claim follows from Lemma 4.9-4.12 and the fact that processing the input takes $\Omega(n + m)$ steps. □

— 5 -

Parallel Links with Restricted Capacity

In this chapter we explore non-cooperative routing on parallel links which cannot process an arbitrary amount of flow.

In Section 5.1 we assume that the links behave like $M/M/1$ queues. The congestion on a link is bounded by its capacity. The latency function for link $j \in [m]$ is $l_j(x) = \frac{1}{c_j - x}$ where c_j is the capacity of link j.

In Section 5.2 we extend the KP model by an additional constraint. Each link has an associated capacity which gives the size of the maximal user traffic that may be routed along that link.

5.1 Parallel M/M/1-Queues

5.1.1 Motivation

Assume we have n users that wish to send streams of packets at different rates $r_1, \ldots, r_n$. That is, r_i is the average number of packets per unit of time that have to be delivered for user i. The arrivals are randomly spaced in time and the interarrival times are distributed exponentially. Each user can choose one of m parallel links to route its stream. Each link represents a queue with exponentially distributed service times. This setting can be modeled by parallel $M/M/1$-queues (according to Kendall's notation, see e.g. [11]). It is well known that the expected delay of such a queue is $\frac{1}{c-x}$, where c is the service rate and x is the expected arrival rate for that queue [11]. The arrival rate must be smaller than the service rate, otherwise the expected delay is unbounded.

5.1.2 Definition

In accordance with our notation in previous chapters, we speak of user weights or user flows, denoted $w_1, \ldots, w_n$, instead of rates. We define the latency function of link $j \in [m]$ as $l_j(x) = \frac{1}{c_j - x}$ for $x < c_j$, where c_j is the capacity of link j (corresponding to the service rate) and x is the link flow (corresponding to the arrival rate). We assume that the latency becomes infinitely large as the flow on a link attains or goes beyond its capacity. The social cost of an assignment A is defined as $SC(A) = SC^{MAX}(A)$, as in the KP model. We denote an instance of the routing model with parallel $M/M/1$-queues as $I = (\mathbf{w}, \mathbf{c})$.

5.1.3 Optimal Routing and Approximation

Assume the flows $w_1, \ldots, w_n \in \mathbb{N}$ of $n = 3m$ users are given. Set all capacities to $c_j = W/m + \frac{1}{2}$ for $j \in [m]$, where W is the total flow. An assignment A for the resulting instance can only have bounded social cost, if all flows are distributed equally among the links, that is, if $x_j(A) = W/m$ for all $j \in [m]$. Obviously, this yields a reduction from the 3-PARTITION Problem which is known to be NP-complete in the strong sense [36] on the problem to decide whether there exists a routing with bounded social cost for a given instance of the routing model with parallel $M/M/1$-queues. Hence, this problem is NP-complete even for identical links. Furthermore, the computation of an optimal assignment or any constant factor approximation is NP-complete as well.

Hence, there does not exist a polynomial ϵ-approximation algorithm for identical links for any $\epsilon > 0$, unless $P = NP$. Henceforth, we investigate the dual approximation. Let an instance $I = (\mathbf{w}, \mathbf{c})$ with identical links be given by the vector of user weights $\mathbf{w} \in \mathbb{R}^n_{>0}$ and the link capacity vector $\mathbf{c} = (c, \ldots, c) \in \mathbb{R}^m_{>0}$. That is, c is the common capacity of all links. An ϵ-primal approximation for the routing problem on I is an assignment A with social cost $SC(A, I) \leq (1 + \epsilon)OPT(I)$ where $SC(A, I) = \max_{j \in [m]} \frac{1}{c_j - x_j(A)}$. A dual approximation achieves optimal social cost but at the price of capacity dilation. That is, if A is an ϵ-dual approximation, then $SC(A, I) \leq OPT(I)$ where $SC(A, I) = \max_{j \in [m]} \frac{1}{(1+\epsilon)c_j - x_j(A)}$. Note that we define any assignment to be optimal, if the optimal social cost is unbounded.

Hochbaum and Shmoys [41] have developed an ϵ-approximation scheme for the Minimal Makespan Problem on identical machines, or, equivalently, for the routing problem in the KP model with identical links (see Section 3.3). We will now show that every ϵ-primal approximation for the routing problem in the KP model with identical links is an ϵ-dual approximation for the routing problem on identical parallel $M/M/1$-queues.

Lemma 5.1. *For an instance $I = (\mathbf{w}, \mathbf{s})$ of the KP model with identical links and $\epsilon > 0$ let assignment A be an ϵ-approximation for the routing problem on I. Then, A is an ϵ-dual approximation for the routing problem on an instance $\tilde{I} = (\mathbf{w}, \mathbf{c})$ of the routing model with identical parallel $M/M/1$-queues and $\mathbf{c} = \mathbf{s}$.*

Proof. From the assumption of the lemma we get that

$$SC(A, I) = \max_{j \in [m]} \frac{x_j(A)}{s_j} \leq (1 + \epsilon)OPT(I)\,.$$

This implies $x_j(A) \leq (1+\epsilon)s_j OPT(I)$ for all $j \in [m]$. If $OPT(\tilde{I}) = \infty$, then any assignment is an ϵ-dual approximation and nothing is to show. Otherwise we know that there exists an assignment A^* (the optimal assignment for $\tilde{I}$) with $x_j(A^*) < c_j = s_j$ for all $j \in [m]$. Hence, $OPT(I) \leq SC(A^*, I) < 1$, which implies $SC(A, I) < (1+\epsilon)$ or, equivalently, $x_j(A) < (1+\epsilon)s_j = (1+\epsilon)c_j$ for all $j \in [m]$. Let c'_j, $j \in [m]$, be the extended link capacities for the $M/M/1$-links. That is, $c'_j = (1+\epsilon)c_j$. Then

$$\frac{1}{c'_j - x_j(A)} \leq \frac{1}{(1+\epsilon)c_j - (1+\epsilon)c_j OPT(I)} = \frac{1}{1+\epsilon} \cdot \frac{1}{c_j} \cdot \frac{1}{1 - OPT(I)}$$

for all $j \in [m]$. Note that all denominators are positive. We claim that

$$\frac{1}{1+\epsilon} \cdot \frac{1}{c_j} \cdot \frac{1}{1 - OPT(I)} \leq OPT(\tilde{I})$$

for all $j \in [m]$ which would conclude the proof, since then

$$\frac{1}{c'_j - x_j(A)} \leq OPT(\tilde{I})$$

for all $j \in [m]$.

Assume, by way of contradiction, that this is not true. Then, for an optimal assignment A^* according to $\tilde{I}$, we have

$$\frac{1}{1+\epsilon} \cdot \frac{1}{c_j} \cdot \frac{1}{1 - OPT(I)} > OPT(\tilde{I}) \geq \frac{1}{c_j - x_j(A^*)}$$

for all $j \in [m]$ (recall that the link capacities $c_j, j \in [m]$, are all the same). This implies that $c_j - x_j(A^*) > c_j(1 - OPT(I))$, or equivalently,

$$\frac{x_j(A^*)}{c_j} = \frac{x_j(A^*)}{s_j} < OPT(I)$$

for all $j \in [m]$, a contradiction, since there cannot exist an assignment for I with social cost less than $OPT(I)$. □

With Lemma 5.1 and the approximation scheme from Hochbaum and Shmoys we obtain a polynomial ϵ-dual approximation scheme for the routing problem on identical parallel $M/M/1$-links.

5.1.4 Nash Equilibria

According to our definition, the latency on a link becomes infinitely large when the congestion attains or exceeds the link capacity. That is, $l_j(x) = \infty$ for $x \geq c_j$. We do not distinguish to which extent the congestion exceeds the capacity.

The coordination ratio for routing on parallel $M/M/1$-links is unbounded, even for identical links. To see this, consider four users with weights $w_1 = w_2 = 2$ and $w_3 = w_4 =$

1 on two links with capacity $c_1 = c_2 = 4$. If both of the large users go to the first and both of the small users go to the second link, then no user can decrease its cost by switching to the opposite link. The latency experienced by the large users, and hence the social cost of this Nash equilibrium assignment, is infinitely large. The optimal assignment has social cost 1 (one small and one large user on each link).

Any assignment on not necessarily identical parallel $M/M/1$-links can be transformed into a Nash equilibrium by performing greedy steps for the users in order of non-increasing weights. After one sweep over all users a Nash equilibrium is reached. This approach was described in [50]. Note, that the same approach works for Nashification in the KP model with identical links (see the discussion after Lemma 3.1 in Section 3.4). In both cases a greedy step of some user cannot cause a smaller or equally sized user to become unsatisfied. A greedy step places an unsatisfied user to the minimal latency link. If the links are kept in a priority queue this can be done in time $\mathcal{O}(\log m)$ with precomputation time $\mathcal{O}(m \log m)$. Hence, it takes time $\mathcal{O}(n \log n + (m+n) \log m)$ to Nashify an assignment.

On the other hand, if the user to perform a greedy step is chosen randomly, then the number of steps until a Nash equilibrium is reached can be superpolynomial in the worst case, as stated by the following lemma.

Lemma 5.2. *For any $k \in \mathbb{N}$ there exists an instance $I = (\mathbf{w}, \mathbf{c})$ of the routing model with identical parallel $M/M/1$-links and $n \geq k$ users, such that the worst case number of greedy steps is at least $2.829^{\sqrt{n}-4}$.*

Proof. For $k \in \mathbb{N}$ let $I' = I(m, 7) = (\mathbf{w}, \mathbf{s})$ be an instance of the KP model according to Definition 3.8 with $n \geq k$ users. Let $I = (\mathbf{w}, \mathbf{c})$ be an instance of the routing model with identical parallel $M/M/1$-links having common link capacity $c = 1 + \sum_{i \in [n]} w_i$. Then, starting with an initial assignment where all users go to the first link and successively performing greedy steps for the respective smallest unsatisfied user (STF selection rule) yields a sequence of at least $2.829^{\sqrt{n}-4}$ steps. The proof for this follows from the proof of Theorem 3.9 and the discussion after, since the same steps are carried out here as for the KP instance I' under the STF policy. □

5.1.5 Generalization

The results of this section hold for a more general model with link capacities as well. Assume the latency on link $j \in [m]$ is defined as $l_j(x) = f(c_j - x)$, where $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ is a common non-increasing function for all links. That is, in this model the latency of a link (or, put in a more general context, the cost of a resource) depends on its unused capacity. The closer it is filled up to its capacity, the more costly it becomes to route additional flow along the link. In [50] such a system is called *residual capacity atomic non-cooperative network*. The model with $M/M/1$-links is a special case where f is defined by $f(y) = \frac{1}{y}$. Another reasonable choice for f is $f(y) = \frac{1}{e^y}$. In this case the latency can equivalently be defined as $l_j(x) = a_j e^x$. Libman and Orda explore the model with arbitrary non-increasing function f in [50]. They show that Nashification can be done in n steps. Concerning the problem to compute an optimal assignment for an instance with

identical links, one can show that an ϵ-dual approximation scheme exists for any choice of f. We omit the proof, because it is quite the same as that of Lemma 5.1.

5.2 Related Links with Weight Restriction

In this section we turn our attention to a model similar to the KP model but with additional restrictions for the maximal user traffic on a link. Let us name this model the *KP model with weight constraints*.

5.2.1 Definition

An instance of this model is a tuple $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$ consisting of the vector of user weights $\mathbf{w} = (w_1, \ldots, w_n)$, the vector of link speeds $\mathbf{s} = (s_1, \ldots, s_m)$ and the vector of maximal traffic sizes $\mathbf{u} = (u_1, \ldots, u_m)$. That is, link j can handle only user flows of size at most u_j. A routing A is *feasible*, if it obeys this additional weight constraint, i.e., if

$$w_i \leq u_{A(i)} \quad \text{for all } i \in [n]\,. \tag{5.1}$$

5.2.2 Computation of Nash Equilibrium

For any instance of this model, the LPT-algorithm, which assigns the users one by one to their respective best feasible link, computes a Nash equilibrium. This follows from the fact that according to the LPT-algorithm the users are taken up in order of non-increasing traffic size. Hence, when a user is placed, its set of feasible links contains all links that are feasible for any user that has already been assigned. This ensures that the arguments in the proof (see [29]) of the fact that LPT yields a Nash equilibrium in the KP model without constraints still work for the KP model with weight constraints.

Let us make an additional assumption. It is reasonable that, comparing two different links, a more powerful link can handle larger pieces of user flow and has higher speed as well. We hence constitute that the larger the maximal traffic size a link can handle is, the greater its speed must be. That is,

$$u_i > u_j \quad \Rightarrow \quad s_i > s_j \quad \text{for all } i, j \in [m]\,. \tag{5.2}$$

The remainder of this section refers to the model with this presumption. Note that nothing is implied for the speeds of two links i and j, if $u_i = u_j$. Thus, if we set $u_j = \max_{i \in [n]} w_i$ for all $j \in [m]$, then we obtain an instance that conforms to the original KP model. The KP model is hence a special case of the KP model with weight constraints and with 5.2.

5.2.3 Approximation of Optimum

The Computation of an optimal routing remains of course NP- complete in the constrained model. But we can apply the approximation scheme for makespan minimization

on related machines from Hochbaum and Shmoys [42] with some modifications to compute an ϵ-approximation, i.e., an assignment with social cost at most $(1+\epsilon)$ times the optimal social cost, for any $\epsilon > 0$ in polynomial time.

We give here a description of the modifications on the algorithm from Hochbaum and Shmoys in [42]. We will use their terminology and definitions without explicitly mentioning. So, the next few paragraphs are not self contained and it is advisable to read [42] first.

The algorithm is based on an ϵ-relaxed decision procedure for Bin Packing. In this context, the link speeds $s_1, \ldots, s_m$ are called bin sizes and the user weights $w_1, \ldots, w_n$ are termed piece sizes. Without loss of generality, we assume $s_1 \geq \cdots \geq s_m$. Given $B > 0$, the decision procedure either computes a packing of the pieces for bin sizes $(1+\epsilon)Bs_j$ or reports that no packing with bin sizes Bs_j exists. This decision procedure is executed repeatedly in a binary search loop. B is the binary search variable. We only have to modify the decision procedure which is based on dynamic programming. It builds up a layered graph, where each vertex corresponds to a state vector describing the distribution of unpacked pieces, i.e., an entry of the vector gives the number of pieces whose rounded size attains some certain value associated with that entry (see [42] for details on the rounding procedure). Each edge between two subsequent layers corresponds to packing one bin by *large* and *medium* pieces. Additionally, there are update layers which care for pieces that are packed as *small* pieces (see [42] for the definition of large, medium and small pieces). The layers are ordered according to the bin sizes. That is, the vertices in layer $j \in [m]$ represent the possible state vectors before packing bin j, and layer $j+1$ contains one vertex for each possible state vector after packing bin j. Consider layer j of the graph. Each vertex of this layer is labeled with a vector $(\mathbf{L}; \mathbf{M}; V_1, V_2, V)$, where vectors $\mathbf{L}$ and $\mathbf{M}$ describe the distribution of unpacked large and medium pieces, and V_1, V_2 and V store the unused slack in *enormous*, *huge* and *large* bins (see [42] for the definition of enormous, huge and large bins).

After building up the graph, a path from the initial vertex (representing the state where all pieces are unpacked) to the success vertex (corresponding to the state where all pieces are packed) has to be found by depth first search. If no such path exists, then we may return from the decision procedure and report failure. Otherwise, the path found corresponds to an ϵ-relaxed packing.

Our modifications in order to adapt the decision procedure to the case where the maximal piece size for a bin $j \in [m]$ is bounded by u_j concern the depth first search. We have to take care that these additional constraints are not violated during the evolvement of the depth first search path. Assume the path has reached some vertex in layer j. According to the depth first search algorithm we have to examine all outgoing edges of that vertex. In the modified algorithm we will not follow up edges that represent a packing of bin j which involves pieces that violate the weight constraint for this bin. Let us call those edges *infeasible edges*. To determine wether the weight constraint is violated, we have to compare the real piece sizes to the capacity u_j of bin j. Note that an edge does not describe exactly which pieces are packed into the bin, but only how many pieces of each subinterval in the interval of large and medium pieces have to be assigned to the bin. During building up the path, we mark each piece that is assigned to some bin with the

index of that bin. Unpacked pieces remain unmarked. If the packing represented by an edge puts p pieces of some subinterval into bin j, then we mark the p largest unpacked pieces of this subinterval that do not violate the weight constraint of bin j. If not enough such pieces are available, then the edge is infeasible.

The second modification concerns the packing of small pieces. After all bins whose sizes lie in the same interval $(\epsilon^{k+1}, \epsilon^k]$ are packed, the remaining large pieces are packed as small pieces in enormous bins. Therefor the unused slack of the enormous bins is stored in an entry V_1 of the state vector. In the modified algorithm we again have to care for the weight constraints not to be violated in this step. Assume the depth first search procedure is examining an edge that represents the packing of pieces as small pieces. Since we have marked all pieces packed so far with the index of their bin, we can compute the unused slack of any eligible bin. We grep up the small pieces to be packed in non-increasing order and pack each in the bin with the largest capacity that has unused slack. If we come across a piece that cannot be packed without violating the weight constraint, then the examined edge is infeasible. Note that we do not constrict the possibilities to pack small pieces in further stages in any way, as all small pieces packed later are smaller than the small pieces considered here.

By the two modifications on the PTAS for machine scheduling described above, we get a PTAS for the KP routing model with weight constraints obeying (5.2). We omit a formal proof here and refer the reader to [42].

5.2.4 Nashification

In Chapter 3 we presented an algorithm `Nashify` (see Figure 3.1) to convert any pure assignment for an instance of the KP model into a pure Nash equilibrium without increasing its social cost. Algorithm `Nashify` works for the KP model with weight constraints (and with (5.2)) as well, if it is modified such that users can only be put on those links where the weight constraint is not violated.

We make use of Proposition 3.5 to prove this fact. Instead of modifying the algorithm, we define the link latency functions such that no user will be placed on a link where it would violate the weight constraint (5.1). Let $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$ be an instance of the KP model with weight constraints. Define $l_j : 2^{[n]} \to \mathbb{R}_{\geq 0}$, the latency function for link $j \in [m]$, by

$$l_j(U) = \begin{cases} \frac{\sum_{i \in U} w_i}{s_j} & \text{if } \forall i \in U : u_j \geq w_i \\ \frac{M + \sum_{i \in U} w_i}{s_j} & \text{if } \exists i \in U : u_j < w_i \end{cases}, \tag{5.3}$$

where $M = W \frac{\max_{k \in [m]} s_k}{\min_{k \in [m]} s_k} + 1$. Starting algorithm `Nashify` on $I' = (\mathbf{1})$ and any assignment A that is feasible for I, it will at no time produce an unfeasible assignment. We state this as a lemma.

Lemma 5.3. *Let $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$ be an instance of the KP model with weight constraints, and let A be a feasible assignment for I. Furthermore, let $I' = (\mathbf{1})$ be an instance of the generic routing model obtained from I according to (5.3). Let A' be any assignment*

generated during the execution of `Nashify` *on input* (I', A)*. Then, no user violates the maximal weight constraint (5.1) in* A'*.*

Proof. Assume the opposite, i.e., there is a user $i \in [n]$ assigned to link $j = A'(i)$ such that $u_j < w_i$. As the social cost of A' for instance I' is at least the latency on link j, we get

$$\begin{aligned} SC(A', I') &\geq l_j(\{k \in [n] : A(k) = A(i)\}) = \frac{M + \sum_{k \in [n]: A(k) = A(i)} w_k}{s_j} \\ &\geq \frac{M}{s_j} = \left(W \frac{\max_{k \in [m]} s_k}{\min_{k \in [m]} s_k} + 1 \right) \frac{1}{s_j} > \frac{W}{\min_{k \in [m]} s_k} \\ &\geq SC(A, I') \,. \end{aligned}$$

This contradicts the fact that algorithm `Nashify` does not increase the social cost throughout its execution. □

The following lemma is the last step until we are ready to state the main result of this section in Theorem 5.5.

Lemma 5.4. *Let* $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$ *be an instance of the KP model with weight constraints obeying (5.2). Let* A *be any feasible assignment for* I*. Let the users and links be numbered such that* $\mathbf{w}$ *and* $\mathbf{s}$ *are sorted non-increasingly, i.e.,* $w_i \geq w_{i+1}$ *and* $s_j \geq s_{j+1}$ *for all* $i \in [n-1], j \in [m-1]$*. Define a corresponding instance* $I' = (\mathbf{l})$ *of the generic routing model with* n *users and latency functions* $\mathbf{l} = (l_1, \ldots, l_m)$ *according to (5.3). Then, properties (i), (ii) and (iii) of Proposition 3.5 are fulfilled for* I' *and* A*.*

Proof. Note that the non-increasing order of the entries of $\mathbf{s}$ implies that $\mathbf{u}$ is sorted non-increasingly, too, because of (5.2). We first prove that property *(i)* of Proposition 3.5 holds. Let $j \in [m], i \in [n-1]$ and $U \subseteq [n] - \{i, i+1\}$ be arbitrary. Then

$$l_j(U \cup \{i\}) = \frac{w_i + \sum_{k \in U} w_k}{s_j} + \begin{cases} 0 & \text{if } \forall k \in U \cup \{i\} : u_j \geq w_k \\ \frac{M}{s_j} & \text{if } \exists k \in U \cup \{i\} : u_j < w_k \end{cases},$$

and

$$l_j(U \cup \{i+1\}) = \frac{w_{i+1} + \sum_{k \in U} w_k}{s_j} + \begin{cases} 0 & \text{if } \forall k \in U \cup \{i+1\} : u_j \geq w_k \\ \frac{M}{s_j} & \text{if } \exists k \in U \cup \{i+1\} : u_j < w_k \end{cases}.$$

Since $w_{i+1} \leq w_i$, $(\exists k \in U \cup \{i+1\} : u_j < w_k)$ implies $(\exists k \in U \cup \{i\} : u_j < w_k)$. It follows that $l_j(U \cup \{i\}) \geq l_j(U \cup \{i+1\})$.

To prove *(ii)*, set $M_j(U) = M$ if $\exists i \in U : u_j < w_i$, and set $M_j(U) = 0$ otherwise for all $j \in [m], U \subseteq [n]$. Let A' and A'' be any two consecutive assignments that are attained during the execution of algorithm `Nashify` on input (I, A). Let user i be the unique user whose link is changed during the transition from A' to A''. Denote $j = A'(i)$ and $k = A''(i)$. Set $U = view(j, A') \backslash \{i\} = \{r \in [n] \backslash \{i\} \mid A'(r) = j\}$ and

$V = view(k, A') = \{r \in [n] \,|\, A'(r) = k\}$. Define $x_j = \sum_{r \in U} w_r$ and $x_k = \sum_{r \in V} w_r$. Assume $j > k$, and let $r \in [n] \backslash U$ with $r < i$ be arbitrary. Then,

$$
\begin{aligned}
& l_j(U \cup \{r\}) - l_k(V \cup \{r\}) - (l_j(U \cup \{i\}) - l_k(V \cup \{i\})) \\
&= \frac{x_j + w_r + M_j(U \cup \{r\})}{s_j} - \frac{x_k + w_r + M_k(V \cup \{r\})}{s_k} \\
&\quad - \left(\frac{x_j + w_i + M_j(U \cup \{i\})}{s_j} - \frac{x_k + w_i + M_k(V \cup \{i\})}{s_k} \right) \\
&= (w_r - w_i) \left(\frac{1}{s_j} - \frac{1}{s_k} \right) \frac{M_j(U \cup \{r\}) - M_j(U \cup \{i\}}{s_j} - \frac{M_k(V \cup \{r\}) - M_k(V \cup \{i\})}{s_k} \\
&\geq \frac{M_j(U \cup \{r\}) - M_j(U \cup \{i\})}{s_j} - \frac{M_k(V \cup \{r\}) - M_k(V \cup \{i\})}{s_k} \,.
\end{aligned}
$$

The last estimation makes use of the fact that $w_r \geq w_i$ and that $s_k \geq s_j$. Note that $M_j(U \cup \{i\}) = M$ implies $u_j < w_i$ and hence that A' is infeasible for I, contradicting Lemma 5.3. Similarly, $M_k(U \cup \{i\}) = M$ implies $u_k < w_i$ and hence that A'' is infeasible for I. Again, this contradicts Lemma 5.3. So, we have $M_j(U \cup \{i\}) = 0$ and $M_k(U \cup \{i\}) = 0$. If $M_k(V \cup \{r\}) = M$, then there is some user $v \in V \cup \{r\}$ with $u_k < w_v$. If $v \neq r$, then A' is infeasible for I, since $A'(v) = k$. But $v = r$ implies $w_r > u_k \geq u_j$, and consequently $M_j(U \cup \{r\}) = M$. Thus,

$$
\begin{aligned}
& l_j(U \cup \{r\}) - l_k(V \cup \{r\}) - (l_j(U \cup \{i\}) - l_k(V \cup \{i\})) \\
&\geq \frac{M_j(U \cup \{r\}) - M_j(U \cup \{i\})}{s_j} - \frac{M_k(V \cup \{r\}) - M_k(V \cup \{i\})}{s_k} \\
&\geq 0 \,,
\end{aligned}
$$

which implies *(ii)* of Proposition 3.5.

To show *(iii)* consider any assignment A' and any link j. Set $U = view(j, A') \backslash \{i\}$. Then, for any user i,

$$
\begin{aligned}
l_j(U \cup \{i\}) &= \frac{w_i + \sum_{k \in U} w_k}{s_j} + \begin{cases} 0 & \text{if } \forall k \in U \cup \{i\} : u_j \geq w_k \\ \frac{M}{s_j} & \text{if } \exists k \in U \cup \{i\} : u_j < w_k \end{cases} \\
&\geq \frac{\sum_{k \in U} w_k}{s_j} + \begin{cases} 0 & \text{if } \forall k \in U : u_j \geq w_k \\ \frac{M}{s_j} & \text{if } \exists k \in U : u_j < w_k \end{cases} \\
&= l_j(U) \,.
\end{aligned}
$$

□

It is now simple to prove that `Nashify` works for the model considered here.

Theorem 5.5. *Let $I = (\mathbf{w}, \mathbf{s}, \mathbf{u})$ be an instance of the KP model with weight constraints, and let A be any feasible assignment for I. Then, algorithm `Nashify` can be used to Nashify A.*

Proof. Renumber the users and links such that $\mathbf{w}$ and $\mathbf{s}$ are sorted non-increasingly, that is, $w_i \geq w_{i+1}$ and $s_j \geq s_{j+1}$ for all $i \in [n-1], j \in [m-1]$. Define an instance $I' = (\mathbf{l})$ of the generic routing model with n users and latency functions $\mathbf{l} = (l_1, \ldots, l_m)$ according to (5.3). Due to Lemma 5.4 and Proposition 3.5 `Nashify` Nashifies A according to I'. Due to Lemma 5.3 A' is feasible for I. Hence, A' is also a Nash equilibrium with non-increased social cost according to I. □

— 6 -

Nash Equilibria of Identical Users on Parallel Links

In this chapter we explore Nash equilibria of identical users on parallel links with quite general latency functions. We measure the social cost as the weighted (by the users' traffic sizes) average user cost, that is, $SC(A, I) = SC^{AVG}(A, I)$, or, equivalently, $SC(A, I) = SC^{MAX}(A, I)$, since both definitions coincide in case of identical users. We hence adopt the social cost definition of the Wardrop model. But in contrast to the Wardrop model we deal with atomic flows which may not be split. Thus, in this aspect we follow the KP model.

Section 6.1 is concerned with pure Nash equilibria. Here, we first prove that the coordination ratio is exactly $\frac{4}{3}$ if we restrict to related links (i.e., with latency functions $l_j(x) = \frac{x}{s_j}$). Furthermore, we give bounds on the coordination ratio for almost arbitrary latency functions that depend on the growth of the functions. Finally, we state a fast and simple algorithm to compute a Nash equilibrium and an optimal assignment.

Section 6.2 addresses general (mixed) Nash equilibria on links with arbitrary convex, non-decreasing and non-constant latency functions. In this setting the so called *fully mixed Nash equilibrium* has the worst social cost, if it exists. It is a special mixed Nash equilibrium where each user selects every link with strictly positive probability. We point out the important role of convex latency functions in this setting. Furthermore, we show that the fully mixed Nash equilibrium is unique if it exists and characterize the set of instances for which it does not exist. At last we show that the social cost of any Nash equilibrium for a given instance is bounded by that of the fully mixed Nash equilibrium for the instance induced by the links that are "sufficiently fast" when processing one user.

6.1 Pure Nash Equilibria

6.1.1 Coordination Ratio for Related Links

We will now prove the following theorem which states that the coordination ratio for pure Nash equilibria of identical users on related links is exactly $\frac{4}{3}$. The proof also reveals the very special structure of a worst case Nash equilibrium.

Theorem 6.1. *Consider any instance I of the congestion routing model with identical users and related links. Then,*

$$\max_{A \in \mathcal{N}(I)} \frac{SC(A,I)}{OPT(I)} = \frac{4}{3}.$$

To prepare the proof of Theorem 6.1, we first show an optimality criterion for our model (Lemma 6.2) and make some claims about the structure of a worst case Nash equilibrium (Lemma 6.3, 6.4 and 6.5). The proof will then derive a contradiction from the existence of an instance with coordination ratio larger than $\frac{4}{3}$. Note that the model under consideration is a special case of the congestion routing model with related links. We may hence denote an instance of this model as $I = (\mathbf{w}, \mathbf{s})$. Our first lemma shows that an assignment has optimal social cost exactly if it is locally optimal, i.e., if the social cost cannot be decreased by reassigning a single user.

Lemma 6.2. *Let A be any pure assignment for an instance $I = (\mathbf{w}, \mathbf{s})$ of the congestion routing model with identical users and related links, and let $\mathbf{w} = (w, \ldots, w)$. Then, A is an optimal assignment, i.e., $SC(A,I) = OPT(I)$, if and only if for every pair of links $i, j \in [m]$*

$$\frac{(x_i(A)+w)^2}{s_i} + \frac{(x_j(A)-w)^2}{s_j} \geq \frac{(x_i(A))^2}{s_i} + \frac{(x_j(A))^2}{s_j}.$$

Proof. 1. '⇒': Let A be an optimal assignment. If we create a new assignment A' by shifting one user from link j to link i for some $i, j \in [m]$, we get

$$SC(A') = SC(A) + \frac{(x_i(A)+w)^2}{s_i} + \frac{(x_j(A)-w)^2}{s_j} - \frac{(x_i(A))^2}{s_i} - \frac{(x_j(A))^2}{s_j}$$

for the social cost of A'. Now, $SC(A') \geq SC(A)$ yields the above inequality.
2. '⇐': Let A^* be an optimal assignment, and let A be any assignment, for which the above inequality holds. Let $x_j = x_j(A)$ and $x_j^* = x_j(A^*)$ be the traffic of link $j \in [m]$ according to A and A^*, respectively. According to the first part of this proof we have

$$\begin{aligned} & \frac{(x_j^*+w)^2}{s_j} + \frac{(x_i^*-w)^2}{s_i} \geq \frac{(x_i^*)^2}{s_i} + \frac{(x_j^*)^2}{s_j} \\ \Leftrightarrow \quad & \frac{2x_j^*w + w^2}{s_j} \geq \frac{2x_i^*w - w^2}{s_i} \quad \text{for all } i, j \in [m] \end{aligned} \tag{6.1}$$

By assumption,

$$\begin{aligned} & \frac{(x_i+w)^2}{s_i}+\frac{(x_j-w)^2}{s_j} \geq \frac{x_i^2}{s_i}+\frac{x_j^2}{s_j} \\ \Leftrightarrow \quad & \frac{2x_iw+w^2}{s_i} \geq \frac{2x_jw-w^2}{s_j} \quad \text{for all } i,j \in [m]\,. \end{aligned} \tag{6.2}$$

If $x_j = x_j^*$ for all $j \in [m]$, then $SC(A) = SC(A^*)$ and nothing is to show. Hence, we may assume that there exist links $i, j \in [m]$ with $x_i^* - x_i > 0$ and $x_j - x_j^* > 0$. I.e., link i has more traffic according to A^* than according to A, and link j has more traffic according to A than to A^*. As the traffic on any link is a multiple of w, we have

$$x_i^* - x_i \geq w \quad \text{and} \tag{6.3}$$
$$x_j - x_j^* \geq w \quad \text{for all } i,j \in [m] \tag{6.4}$$

But this yields

$$\begin{aligned} \frac{2x_j^*w+w^2}{s_j} &\overset{(6.1)}{\geq} \frac{2x_i^*w-w^2}{s_i} \overset{(6.3)}{\geq} \frac{2x_iw+w^2}{s_i} \\ &\overset{(6.2)}{\geq} \frac{2x_jw-w^2}{s_j} \overset{(6.4)}{\geq} \frac{2x_j^*w+w^2}{s_j} \quad \text{for all } i,j \in [m] \end{aligned}$$

Notice, that the left and right side of the inequality chain are identical and therefore all terms must be equal. On the one hand this implies $x_j = x_j^* + w$ and $x_i^* = x_i + w$. That is, for each link, the traffics according to A and A^* can differ by at most one user. This implies that the number of links with $x_i^* - x_i > 0$ and of those with $x_j - x_j^* > 0$ are equal. On the other hand we can follow

$$\begin{aligned} & \frac{2x_j^*w+w^2}{s_j} = \frac{2x_iw+w^2}{s_i} \\ \Leftrightarrow \quad & \frac{x_i^2}{s_i}+\frac{(x_j^*+w)^2}{s_j} = \frac{(x_i+w)^2}{s_i}+\frac{(x_j^*)^2}{s_j} \\ \Leftrightarrow \quad & \frac{x_i^2}{s_i}+\frac{x_j^2}{s_j} = \frac{(x_i^*)^2}{s_i}+\frac{(x_j^*)^2}{s_j} \quad \text{for all } i,j \in [m]\,. \end{aligned} \tag{6.5}$$

(6.5) tells us that the contribution of links i and j to the social cost is the same for A and A^*. As for every link j that has a greater traffic according to A than to A^* there exists some link i that has a lower traffic according to A than to A^*, we have $SC(A) = SC(A^*)$. □

The next lemma claims that in an optimal assignment any link carries at most one user more than according to any Nash equilibrium assignment.

Lemma 6.3. *Let $I = (\mathbf{w}, \mathbf{s})$ be an instance of the congestion routing model with identical users and related links, and let A^* and A be an optimal and a Nash equilibrium assignment, respectively. Without loss of generality, let $\mathbf{w} = (1, \ldots, 1)$. Then, $x_j(A^*, I) - x_j(A, I) \leq 1$ for all $j \in [m]$.*

Proof. Denote $x_j = x_j(A)$ and $x_j^* = x_j(A^*)$. Assume the claim is violated for some link k, i.e.,

$$x_k \leq x_k^* - 2\,. \tag{6.6}$$

Due to Lemma 6.2, optimality of A^* implies for all $j \in [m]$

$$\begin{aligned} & \frac{(x_j^*+1)^2}{s_j} + \frac{(x_k^*-1)^2}{s_k} \geq \frac{(x_j^*)^2}{s_j} + \frac{(x_k^*)^2}{s_k} \\ \Leftrightarrow \quad & \frac{2x_j^*+1}{s_j} \geq \frac{2x_k^*-1}{s_k} \\ \Leftrightarrow \quad & \frac{x_j^*}{s_j} \geq \frac{2x_k^*-1}{2s_k} - \frac{1}{2s_j}\,. \end{aligned} \tag{6.7}$$

Since A is at Nash equilibrium,

$$\frac{x_j}{s_j} \leq \frac{x_k+1}{s_k} \text{ for all } j \in [m]\,. \tag{6.8}$$

Now, let j be some link with

$$x_j \geq x_j^* + 1\,.$$

Such link exists, because $\sum_{i\in[m]} x_i^* = \sum_{i\in[m]} x_i = n$. Then,

$$\begin{aligned} & \frac{2x_k^*-1}{2s_k} - \frac{1}{2s_j} + \frac{1}{s_j} \overset{(6.7)}{\leq} \frac{x_j^*+1}{s_j} \leq \frac{x_j}{s_j} \\ & \overset{(6.8)}{\leq} \frac{x_k+1}{s_k} \overset{(6.6)}{\leq} \frac{x_k^*-1}{s_k} \\ \Rightarrow & \frac{1}{2s_k} + \frac{1}{2s_j} \leq 0\,, \end{aligned}$$

a contradiction. □

The following lemma reveals the very special structure of any minimal example of an instance with coordination ratio greater than a given constant.

Lemma 6.4. *Let $I = (\mathbf{w}, \mathbf{s})$ be a minimal (in the number of links) instance of the congestion routing model with identical users and related links for which a Nash equilibrium assignment A exists, such that $\frac{SC(A,I)}{OPT(I)} > \rho$ for a given $\rho \in \mathbb{R}_{\geq 1}$. Without loss of generality, let $\mathbf{w} = (1,\ldots,1)$. Then, $x_k(A,I) = x_k(A^*,I) + m - 1$ for some $k \in [m]$, and $x_j(A,I) = x_j(A^*,I) - 1$ for all $j \in [m]\backslash\{k\}$, where A^* is an optimal assignment for I.*

Proof. Let A^* be an optimal assignment for I, and let A be a Nash equilibrium with $\frac{SC(A)}{SC(A^*)} > \rho$. First, let us assume that there exists some link k with $x_k(A) = x_k(A^*)$. Then, removing that link and all associated users yields a new instance with coordination ratio

$$\rho' = \frac{SC(A) - (x_k(A^*))^2/s_k}{SC(A^*) - (x_k(A^*))^2/s_k}\,.$$

$\rho' < \frac{SC(A)}{SC(A^*)}$ would imply

$$\begin{aligned} & (SC(A) - (x_k(A^*))^2/s_k)SC(A^*) < SC(A)(SC(A^*) - (x_k(A^*))^2/s_k) \\ \Leftrightarrow \quad & SC(A^*) > SC(A)\,, \end{aligned}$$

a contradiction, since A^* has minimal social cost. Hence, the new instance has at least the same coordination ratio but fewer links, which contradicts our assumption that I is a minimal instance with coordination ratio greater than ρ.
Now, assume more than one link have $x_j(A) \geq x_j(A^*) + 1$, and let link k be one of them. By Lemma 6.3 and the first part of this proof, all links with $x_j(A) \not\geq x_j(A^*) + 1$ have $x_j(A) = x_j(A^*) - 1$. Note that there must exist at least $(x_k(A) - x_k(A^*))$ such links and let S be a set of exactly $(x_k(A) - x_k(A^*))$ such links. Setting

$$\begin{aligned} C &= \sum_{j \in [m]\setminus(\{k\}\cup S)} \frac{(x_j(A))^2}{s_j}, \\ D &= \sum_{j \in [m]\setminus(\{k\}\cup S)} \frac{(x_j(A^*))^2}{s_j}, \\ E &= \sum_{j \in S\cup\{k\}} \frac{(x_j(A))^2}{s_j} \text{ and} \\ F &= \sum_{j \in S\cup\{k\}} \frac{(x_j(A^*))^2}{s_j}, \end{aligned}$$

we can write the coordination ratio of A as

$$\frac{SC(A)}{SC(A^*)} = \frac{\sum_{j\in[m]} \frac{(x_j(A))^2}{s_j}}{\sum_{j\in[m]} \frac{(x_j(A^*))^2}{s_j}} = \frac{C+E}{D+F}\,.$$

Either $\frac{C}{D} > \frac{E}{F}$, or $\frac{C}{D} \leq \frac{E}{F}$. In the first case we can remove the links in $S \cup \{k\}$ together with the users assigned to them, obtaining an instance I_1 with fewer links. Let A_1 be the subassignment of A induced by I_1. Of course A_1 remains a Nash equilibrium for I_1. Also the subassignment of A^* induced by I_1 remains optimal for I_1. Hence I_1 has coordination ratio at least $\frac{C}{D} > \frac{C+E}{D+F} > \rho$. Similarly, in the second case, we can remove all links in $[m]\setminus(S \cup \{k\})$ and their associated users, obtaining an instance I_2 with fewer links and at least the same coordination ratio at least $\frac{E}{F} \geq \frac{C+E}{D+F} > \rho$. Both cases contradict the assumption, that I has minimal number of links among all instances with coordination ratio greater than ρ. □

Lemma 6.5. *Let $I = (\mathbf{w}, \mathbf{s})$ be a minimal (in the number of links) instance of the congestion routing model with identical users and related links, with a worst case Nash equilibrium assignment A and an optimal assignment A^*, such that $\frac{SC(A,I)}{SC(A^*,I)} > \rho$ for a given $\rho \in \mathbb{R}_{\geq 1}$. Without loss of generality, let $\mathbf{w} = (1, \ldots, 1)$. Then, there exists*

an instance $\tilde{I} = (\tilde{\mathbf{w}}, \tilde{\mathbf{s}})$ that has the same number m of links as I, with optimal assignment $\tilde{A}^*$ and Nash equilibrium assignment $\tilde{A}$, such that $x_k(\tilde{A}, \tilde{I}) = x_k(\tilde{A}^*, \tilde{I}) + m - 1$ for some link $k \in [m]$, $x_i(\tilde{A}^*, \tilde{I}) = x_j(\tilde{A}^*, \tilde{I})$ and $\tilde{s}_i = \tilde{s}_j$ for all $i, j \in [m]\backslash\{k\}$ and $\frac{SC(\tilde{A},\tilde{I})}{SC(\tilde{A}^*,\tilde{I})} \geq \frac{SC(A,I)}{SC(A^*,I)}$.

Proof. Let $k \in [m]$ be the link with $x_k(A) = x_k(A^*) + m - 1$ due to Lemma 6.4. Let $S \subset [m]\backslash\{k\}$ be a maximal set of links, such that $x_i(A^*) = x_j(A^*)$ and $s_i = s_j$ for all $i, j \in S$, and let $s = |S|$ be the cardinality of S. If $s = m-1$, then the claim holds. Otherwise let $i \in [m]\backslash(S \cup \{k\})$ be arbitrary. Note that $x_i(A) = x_i(A^*) - 1$ for all $i \in [m]\backslash\{k\}$ due to Lemma 6.4. We now obtain two new instances $I_1 = (\mathbf{w}', \mathbf{s}')$ and $I_2 = (\mathbf{w}'', \mathbf{s}'')$ from $I = (\mathbf{w}, \mathbf{s})$ and corresponding pairs of assignments A_1, A_1^* and A_2, A_2^* by

a) setting the traffic of link i according to A_1 and A_1^* to the traffic of the links in S according to A and A^*, respectively (this yields A_1 and A_1^*); and setting the speed of link i to the speed of the links in S (this yields I_1),
that is $s'_j := s_t$, $x_j(A_1) := x_t(A)$ and $x_j(A_1^*) := x_t(A^*)$ for all $j \in S \cup \{i\}$ and $t \in S$ arbitrary, and $s'_j := s_j$, $x_j(A_1) := x_j(A)$ and $x_j(A_1^*) := x_j(A^*)$ for all $j \in [m]\backslash(S \cup \{i\})$.

b) setting the traffic of the links in S according to A_2 and A_2^* to the traffic of link i according to A and A^*, respectively (this yields A_2 and A_2^*); and setting the speed of the links in S to the speed of link i (this yields I_2)
that is $s''_j := s_i$, $x_j(A_2) := x_i(A)$ and $x_j(A_2^*) := x_i(A^*)$ for all $j \in S \cup \{i\}$, and $s''_j := s_j$, $x_j(A_2) := x_j(A)$ and $x_j(A_2^*) := x_j(A^*)$ for all $j \in [m]\backslash(S \cup \{i\})$.

We will now show that either

$$\frac{SC(A_1, I_1)}{SC(A_1^*, I_1)} \geq \frac{SC(A, I)}{SC(A^*, I)} \text{ or } \frac{SC(A_2, I_2)}{SC(A_2^*, I_2)} \geq \frac{SC(A, I)}{SC(A^*, I)}\,.$$

Set

$$\begin{aligned}
C &= \sum_{j \in [m]\backslash(S \cup \{i\})} \frac{(x_j(A))^2}{s_j}, \\
D &= \sum_{j \in [m]\backslash(S \cup \{i\})} \frac{(x_j(A^*))^2}{s_j}, \\
e &= \frac{(x_i(A^*) - 1)^2}{s_i}, \\
f &= \frac{(x_i(A^*))^2}{s_i}, \\
g &= \frac{(x_j(A^*) - 1)^2}{s_j} \forall j \in S \quad \text{and} \\
h &= \frac{(x_j(A^*))^2}{s_j} \forall j \in S\,.
\end{aligned}$$

Then,

$$\frac{SC(A,I)}{SC(A^*,I)} = \frac{C+e+sg}{D+f+sh}$$
$$\frac{SC(A_1,I_1)}{SC(A_1^*,I_1)} = \frac{C+(s+1)g}{D+(s+1)h}$$
$$\frac{SC(A_2,I_2)}{SC(A_2^*,I_2)} = \frac{C+(s+1)e}{D+(s+1)f}\,.$$

Assume,

$$\frac{SC(A_1,I_1)}{SC(A_1^*,I_1)} < \frac{SC(A,I)}{SC(A^*,I)} \text{ and } \frac{SC(A_2,I_2)}{SC(A_2^*,I_2)} < \frac{SC(A,I)}{SC(A^*,I)}\,.$$

This yields

$$\begin{aligned}
& (C+(s+1)g)(D+f+sh) < (C+e+sg)(D+(s+1)h) \\
& \wedge\, (C+(s+1)e)(D+f+sh) < (C+e+sg)(D+(s+1)f) \\
\Leftrightarrow\quad & Cf+Dg+(s+1)fg < Ch+De+(s+1)eh \\
& \wedge\, sCh+sDe+s(s+1)eh < sCf+sDg+s(s+1)fg \\
\Rightarrow\quad & Cf+Dg+(s+1)fg < Cf+Dg+(s+1)fg\,,
\end{aligned}$$

a contradiction. A_1 is a Nash equilibrium for I_1, because the Nash equilibrium condition holds for the links in S (no user can improve by moving from or to one of those links) and therefore is also valid for link i which is a copy of the links in S in the modified instance. Similarly, A_2 is a Nash equilibrium for I_2, because the Nash equilibrium condition holds for link i (no user can improve by moving from or to link i) and hence also for the links in S which are copies of link i in the modified instance. As $OPT(I_1) \leq SC(A_1^*, I_1)$ and $OPT(I_2) \leq SC(A_2^*, I_2)$, we have either

$$\frac{SC(A_1,I_1)}{OPT(I_1)} \geq \frac{SC(A_1,I_1)}{SC(A_1^*,I_1)} \geq \frac{SC(A,I)}{OPT(I)} \text{ or } \frac{SC(A_2,I_2)}{OPT(I_2)} \geq \frac{SC(A_2,I_2)}{SC(A_2^*,I_2)} \geq \frac{SC(A,I)}{OPT(I)}\,.$$

Thus, either I_1 or I_2 is an instance with at least the same coordination ratio as instance I.

In both modified instances, the number of equal links (the links in S) has increased to $s+1$. Thus, by applying this step repeatedly, we eventually equalize all links except for link k and obtain an instance $\tilde{I}$ and assignments $\tilde{A}$ and $\tilde{A}^*$ as stated in the lemma. Note, that the number m of links remains the same, while the number of users might change, but that the latter is bounded by m times the maximal link congestion according to assignment A for the original instance. □

Using Lemma 6.5 and 6.2, we are now ready to prove the theorem.

Proof of Theorem 6.1:

We first prove, by way of contradiction, that the coordination ratio is at most $\frac{4}{3}$. Hence, assume that (A, I) is a minimal counterexample. That is, $I = (\mathbf{w}, \mathbf{s})$ is a minimal (in the number of links) instance of the congestion routing model with identical users and

related links with a worst case Nash equilibrium A, such that $\frac{SC(A)}{OPT(I)} > \frac{4}{3}$. Let A^* be an optimal assignment for I, and, without loss of generality, let $\mathbf{w} = (1, \ldots, 1)$. Note, that $x_j(A), x_j(A^*) \in \mathbb{N}_0$ for all $j \in [m]$, and that $m \geq 2$.

Due to Lemma 6.5 we may assume that for some link k the traffic according to A exceeds the traffic according to A^* by $m-1$, whereas the traffic of any other link according to A is by one lower than according to A^*, and that all the latter links have equal speed, say s, and equal traffic according to A^*, say $x(A^*)$. Without loss of generality, we may assume $k = 1$ and $s = 1$. Thus,

$$\begin{aligned} &x_1(A) = x_1(A^*) + m - 1, \quad \text{and} \\ &x_j(A) = x_j(A^*) - 1 = x(A^*) - 1,\ s_j = 1 \quad \text{for } 2 \leq j \leq m\,. \end{aligned}$$

For the social costs of A and A^* we have

$$SC(A) = \frac{(x_1(A^*) + m - 1)^2}{s_1} + (m-1)(x(A^*) - 1)^2 \quad \text{and}$$
$$OPT(I) = SC(A^*) = \frac{(x_1(A^*))^2}{s_1} + (m-1)(x(A^*))^2\,.$$

Hence, $\frac{SC(A)}{OPT(I)}$ is bounded from above by the result of the following optimization problem with three unknowns x_1, x and s_1:

$$\underset{x_1 \in \mathbb{N}_0,\, x \in \mathbb{N},\, s_1 \in \mathbb{R}_{>0}}{\text{maximize}} \quad \frac{\frac{(x_1+m-1)^2}{s_1} + (m-1)(x-1)^2}{\frac{(x_1)^2}{s_1} + (m-1)x^2} \quad \text{subject to} \tag{6.9}$$

(a) $\frac{(x_1+1)^2}{s_1} + (x-1)^2 \geq \frac{(x_1)^2}{s_1} + x^2$
(b) $\frac{(x_1-1)^2}{s_1} + (x+1)^2 \geq \frac{(x_1)^2}{s_1} + x^2$
(c) $\frac{x_1+m-1}{s_1} \leq x$
(d) $\frac{x_1+m}{s_1} \geq x - 1$

x_1 and x correspond to $x_1(A^*)$ and $x(A^*)$, respectively. Because of $0 \leq x_j(A) = x(A^*) - 1$ for $j \geq 2$, $x(A^*)$ cannot be zero.

Because of Lemma 6.2, (a) and (b) are necessary and sufficient for the optimality of A^*. (c) and (d) express the fact that A is at Nash equilibrium.

Note that we may rewrite (a) as

$$\frac{2x_1 + 1}{s_1} \geq 2x - 1 \quad \text{or} \quad \frac{x_1 + \frac{1}{2}}{s_1} \geq x - \frac{1}{2}$$

and (b) as

$$2x + 1 \geq \frac{2x_1 - 1}{s_1} \quad \text{or} \quad x + \frac{1}{2} \geq \frac{x_1 - \frac{1}{2}}{s_1}\,.$$

Hence (a) implies (d) and (c) implies (b) and we may immediately drop (d) and (b).

Setting $C = (x_1+m-1)^2$, $D = (x_1)^2$, $E = (m-1)(x-1)^2$ and $F = (m-1)x^2$, we can rewrite the fraction to be maximized as $\frac{C+s_1E}{D+s_1F}$. Since, $\frac{C}{D} > 1 > \frac{E}{F}$, we have to minimize s_1 (subject to the constraints imposed by x_1 and x) in order to maximize the fraction.

(c) imposes a lower bound of $\frac{x_1+m-1}{x}$ on s_1 (while (a) only yields an upper bound).

Thus we have to set $s_1 = \frac{x_1+m-1}{x}$ in order to maximize the fraction which becomes now

$$\begin{aligned}
&\frac{(x_1+m-1)^2x + (m-1)(x-1)^2(x_1+m-1)}{(x_1)^2x + (m-1)x^2(x_1+m-1)} \\
=&1 + \frac{(m-1)^2x + 2x_1(m-1)x + (m-1)(-2x+1)(x_1+m-1)}{(x_1)^2x + (m-1)x^2(x_1+m-1)} \\
=&1 + \frac{-(m-1)^2x + (m-1)(x_1+m-1)}{(x_1)^2x + (m-1)x^2(x_1+m-1)}\,.
\end{aligned}$$

The only remaining constraint (a) becomes

$$\frac{2x_1+1}{x_1+m-1}x \geq 2x-1$$

or

$$x_1 + m - 1 \geq (2m-3)x\,.$$

This simplifies the optimization problem as follows.

$$\underset{x_1\in\mathbb{N}_0,\, x\in\mathbb{N}}{\text{maximize}} \;\; 1 + (m-1)\frac{-(m-1)x + (x_1+m-1)}{(x_1)^2x + (m-1)(x_1+m-1)x^2}$$

subject to
(a) $x_1 \geq x(2m-3) - (m-1)\,.$

The denominator of the fraction is positive and increasing in x. (a) implies that the numerator of the fraction is non-negative for all $m \geq 2$. As the numerator is decreasing in x, x has to be minimized in order to maximize the fraction. The lowest possible value for x (because of $x \in \mathbb{N}$) is $x = 1$. This yields

$$\underset{x_1\in\mathbb{N}_0}{\text{maximize}} \;\; 1 + f(x_1)$$

subject to
(a) $x_1 \geq m-2\,,$
where

$$f(x_1) = \frac{(m-1)x_1}{(x_1)^2 + (m-1)(x_1+m-1)}\,.$$

To maximize $f(x_1)$ over $\mathbb{N}_0$ we first compute an upper bound on the maximum by maximizing $f(x_1)$ over $\mathbb{R}_{\geq 0}$.

The first order derivative of $f_1(x_1)$ is

$$\begin{aligned}\frac{\partial f(x_1)}{\partial x_1} &= (m-1)\frac{((x_1)^2+(m-1)(x_1+m-1))-x_1(2x_1+m-1)}{[(x_1)^2+(m-1)(x_1+m-1)]^2}\\ &= (m-1)\frac{(m-1)^2-(x_1)^2}{[(x_1)^2+(m-1)(x_1+m-1)]^2}\,.\end{aligned}$$

$\frac{\partial f(x_1)}{\partial x_1}=0$ implies $(x_1)^2=(m-1)^2$. Hence $f(x_1)$ attains its only local extremum over $\mathbb{R}_{\geq 0}$ at $x_1=m-1$. As $f(x_1)$ is continuous over $\mathbb{R}_{\geq 0}$, $f(0)=0$, $\lim_{x_1\to\infty} f(x_1)=0$ and $f(m-1)=\frac{1}{3}>0$, this local extremum is the global maximum of $f(x_1)$ on $\mathbb{R}_{\geq 0}$. As $m-1$, the position where $f(x_1)$ attains its maximum over $\mathbb{R}_{\geq 0}$, is integral and lies in the feasible range of our optimization problem, $f(m-1)=\frac{1}{3}$ is the exact solution to the optimization problem rather than only an upper bound on it.

Hence, $\frac{SC(A)}{OPT(I)} \leq 1+f(m-1)=\frac{4}{3}$, contradicting the existence of an instance with $\frac{SC(A)}{OPT(I)}>\frac{4}{3}$.

Finally, we give an example that proves the bound tight. Let $m\geq 2$ be arbitrary. Define an instance $I=(\mathbf{w},\mathbf{s})$ of the congestion routing model with m related links and identical users. Set $s_1=2(m-2), s_2=\cdots=s_m=1$, and let A^* and A be assignments with

$$\begin{aligned} x_1(A^*)&=m-1, & x_2(A^*)&=\cdots=x_m(A^*)=1,\\ x_1(A)&=2(m-1), & x_2(A)&=\cdots=x_m(A)=0\,.\end{aligned}$$

A is at Nash equilibrium, because

$$\frac{x_j(A)+1}{s_j}=1\geq\frac{2(m-1)}{2(m-2)}=\frac{x_1(A)}{s_1}\quad\text{for } 2\leq j\leq m\,.$$

For the (pure) coordination ration of I we get

$$PCR(I)\geq\frac{SC(A)}{SC(A^*)}=\frac{\frac{(2(m-1))^2}{2(m-1)}}{\frac{(m-1)^2}{2(m-1)}+(m-1)}=\frac{4}{3}\,.$$

□

The lower bound example given in the proof of Theorem 6.1 works for any number $m\geq 2$ of links. Hence, the coordination ratio is the same for 2 links and an arbitrary large number of links.

If we allowed that flow is split arbitrarily, then every Nash equilibrium has optimal social cost, as users would distribute their flow uniformly (weighted by link speeds) among the links. If flow is splittable but the link latencies are given by arbitrary linear functions $l_j(x)=a_jx+b_j$, $a_j,b_j\geq 0$, then the coordination ratio is also $\frac{4}{3}$ [76].

6.1.2 Coordination Ratio for General Latency Functions

In this section, we carry over two upper bounds from Roughgarden and Tardos on the coordination ratio for splittable flows in arbitrary routing networks to our model with unit sized unsplittable flows on parallel links. The first [76, Corollary 2.10] can be adapted directly to our discrete setting. The second [74, Theorem 3.8] is a genuine bound on the coordination ratio, denoted *anarchy value*, for splittable flows. We will prove an analog bound with help of the discrete version of the anarchy value that will be introduced in Definition 6.10.

Recall that an instance of the congestion routing model with identical users is given by $I = (n, \mathbf{l})$. The latency on a link j with congestion x is given by $l_j(x)$. We say that $xl_j(x)$ is the cost function of link j. That is, $xl_j(x)$ is the contribution of link j to the social cost. Indeed, the social cost of a routing A is the sum of the cost contributions of all links.

If all cost functions are convex, then a splittable flow is globally optimal, exactly if it is locally optimal, i.e., if any reassignment of an infinitesimal amount of flow does not decrease the social cost [6, 20].

An analog property holds for unsplittable flows. Here, we have discrete latency functions of the form $l : [n] \to \mathbb{R}$, and hence discrete cost functions. The notion of convexity refers to what is known as discrete convexity in this case. Let us define this formally.

Definition 6.6. *We call a discrete function $f : [n] \to \mathbb{R}$ convex, if*

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

for all $x, y \in [n-1]$ with $x < y$.

Our notion of a convex function corresponds to the particular case of a *discretely convex function* as defined by Murota and Shioura in [59] for one dimension. Their definition is basically due to a corresponding definition of Miller [57].

The following lemma implies that for an instance of the congestion routing model with convex cost functions an unsplittable flow of unit sized users is globally optimal, exactly if it is locally optimal. We will use it in the proof of Lemma 6.8 and Lemma 6.14.

Lemma 6.7. *Let $f_j : [n] \cup \{0\} \to \mathbb{R}$ be a convex function for $j \in [m]$. Set $X = \{\mathbf{x} = (x_1, \ldots, x_m) \in \mathbb{N}_0^m \mid \sum_{j\in[m]} x_j = n\}$ and let $\mathbf{x} \in X$. Then, $\sum_{j\in[m]} f_j(x_j)$ is minimal among all $\mathbf{y} \in X$, i.e., $\sum_{j\in[m]} f_j(x_j) \leq \sum_{j\in[m]} f_j(y_j)$ for all $\mathbf{y} = (y_1, \ldots, y_m) \in X$, if and only if*

$$f_j(x_j + 1) + f_k(x_k - 1) \geq f_j(x_j) + f_k(x_k) \quad \text{for all } j, k \in [m], x_k \geq 1\,. \tag{6.10}$$

Proof. Define $f : X \to \mathbb{R}$ by $f(\mathbf{x}) = \sum_{j\in[m]} f_j(x_j)$. Assume $\sum_{j\in[m]} f_j(x_j)$ is minimal, but $f_j(x_j + 1) + f_k(x_k - 1) < f_j(x_j) + f_k(x_k)$ for some $j, k \in [m]$. Determine $\mathbf{y} = (y_1, \ldots, y_m) \in X$ by $y_j = x_j + 1$, $y_k = x_k - 1$ and $y_t = x_t$ for all $t \in [m]\backslash\{j, k\}$. Then,

$$\sum_{t\in[m]} f_t(y_t) = f(x_j + 1) + f(x_k - 1) + \sum_{t\in[m]\backslash\{j,k\}} f_t(x_t) < \sum_{j\in[m]} f_j(x_j)\,,$$

a contradiction.

Assume now that (6.10) holds, but that there exists $\mathbf{y} = (y_1, \ldots, y_m) \in X$ with $f(\mathbf{x}) > f(\mathbf{y})$. Set $J_+(\mathbf{x}, \mathbf{y}) = \{j \in [m] \,|\, y_j > x_j\}$, $J_-(\mathbf{x}, \mathbf{y}) = \{j \in [m] \,|\, y_j < x_j\}$ and $t = \sum_{j \in J_+}(y_j - x_j)$. Consider the sequence of vectors $\mathbf{x} = \mathbf{x}(0), ..., \mathbf{x}(t) = \mathbf{y}$, where $\mathbf{x}(i)$ is obtained from $\mathbf{x}(i-1)$ for $i \in [t]$ by setting $r = \min J_+(\mathbf{x}(i-1), \mathbf{y})$, $s = \min J_-(\mathbf{x}(i-1), \mathbf{y})$, $x_r(i) = x_r(i-1)+1$, $x_s(i) = x_s(i-1)-1$ and $x_j(i) = x_j(i-1)$ for $j \in [m]\backslash\{r, s\}$.

Claim: For any $i \in [t]$, $f(\mathbf{x}(i-1)) \leq f(\mathbf{x}(i))$.

If the claim is valid, then $f(\mathbf{x}) \leq f(\mathbf{x}(1)) \leq \cdots \leq f(\mathbf{x}(t-1)) \leq f(\mathbf{y})$, a contradiction with our assumption. This would complete the proof.

Proof of claim: Let $i \in [t]$ be arbitrary. Set $r = \min J_+(\mathbf{x}(i-1), \mathbf{y})$ and $s = \min J_-(\mathbf{x}(i-1), \mathbf{y})$. Then,

$$\begin{aligned} f(\mathbf{x}(i-1)) &= \sum_{j \in [m]} f_j(x_j(i-1)) \\ &= f_r(x_r(i-1)) + f_s(x_s(i-1)) + \sum_{j \in [m]\backslash\{r,s\}} f_j(x_j(i-1)) \end{aligned}$$

Note that $x_j(i) \geq x_j$ for all $j \in J_+$ and $i \in [t]$, and that $x_j(i) \leq x_j$ for all $j \in J_-$ and $i \in [t]$. Hence, $x_r(i-1) \geq x_r$ and $x_s(i-1) \leq x_s$. (6.10) implies

$$f_r(x_r + 1) + f_s(x_s - 1) \geq f_r(x_r) + f_s(x_s)$$

or equivalently,

$$f_r(x_r + 1) - f_r(x_r) \geq f_s(x_s) - f_s(x_s - 1)\,.$$

Since f_r and f_s are convex, we obtain

$$f_r(x_r(i-1) + 1) - f_r(x_r(i-1)) \geq f_r(x_r + 1) - f_r(x_r) \text{ and}$$

$$f_s(x_s(i-1)) - f_s(x_s(i-1) - 1) \leq f_s(x_s) - f_s(x_s - 1)\,.$$

Thus,

$$\begin{aligned} f_r(x_r(i)) - f_r(x_r(i-1)) &= f_r(x_r(i-1) + 1) - f_r(x_r(i-1)) \\ &\geq f_s(x_s(i-1)) - f_s(x_s(i-1) - 1) \\ &= f_s(x_s(i-1)) - f_s(x_s(i)) \end{aligned}$$

which yields

$$f_r(x_r(i)) + f_s(x_s(i)) \geq f_r(x_r(i-1)) + f_s(x_s(i-1))\,.$$

It follows that

$$f(\mathbf{x}(i-1)) \leq f_r(x_r(i)) + f_s(x_s(i)) + \sum_{j \in [m]\backslash\{r,s\}} f_j(x_j(i-1)) = \sum_{j \in [m]} f_j(x_j(i)) = f(\mathbf{x}(i))\,,$$

as desired. □

The next lemma states the first upper bound on the coordination ratio in our model. This bound is not tight, but the proof is simple.

Lemma 6.8. *Consider any instance I of the congestion routing model with identical users and non-decreasing latency functions. If $xl_j(x) \leq \alpha \sum_{t=1}^{x} l_j(t)$ for all $j \in [m], x \in [n]$, then*

$$SC(A, I) \leq \alpha OPT(I)$$

for any pure Nash equilibrium A.

Proof. Let $x^* = (x_1^*, \ldots, x_m^*)$ be the vector of link congestions in an optimal assignment. Set $f_j(x) = \sum_{t=1}^{x} l_j(t)$ for all $j \in [m]$. In a pure Nash equilibrium A that assigns $x_j = x_j(A)$ users to link j, we have $l_j(x_j + 1) \geq l_k(x_k)$ which implies $f_j(x_j + 1) + f_k(x_k - 1) \geq f_j(x_j) + f_k(x_k)$ for all $j, k \in [m]$. As l_j is non-decreasing, f_j is convex for all $j \in [m]$. Hence we can apply Lemma 6.7 and obtain that $\sum_{j\in[m]} f_j(x_j)$ is minimal among all assignments with $\sum_{j\in[m]} x_j = n$. Thus,

$$\begin{aligned} SC(A) &= \sum_{j=1}^{m} x_j l_j(x_j) \leq \alpha \sum_{j=1}^{m} \sum_{t=1}^{x_j} l_j(t) = \alpha \sum_{j=1}^{m} f_j(x_j) \leq \alpha \sum_{j=1}^{m} f_j(x_j^*) \\ &= \alpha \sum_{j=1}^{m} \sum_{t=1}^{x_j^*} l_j(t) \leq \alpha \sum_{j=1}^{m} x_j^* l_j(x_j^*) = \alpha \cdot OPT(I) \end{aligned}$$

□

The following corollary is an example for the application of the upper bound.

Corollary 6.9. *For the congestion routing model with identical users and where link latencies are polynomial functions with non-negative coefficients and maximal degree d, the coordination ratio for pure Nash equilibria is bounded by $d + 1$.*

Proof. Let $l(x) = \sum_{k=0}^{d} a_k x^k$, $a_d > 0$. Then, for $x \in [n]$,

$$\frac{xl(x)}{\sum_{t=1}^{x} l(t)} = \frac{\sum_{k=0}^{d} a_k x^{k+1}}{\sum_{k=0}^{d} a_k \sum_{t=1}^{x} t^k} \leq \frac{\sum_{k=0} a_k x^{k+1}}{\sum_{k=0}^{d} a_k \int_{t=0}^{x} t^d dt} = \frac{\sum_{k=0}^{d} a_k x^{k+1}}{\sum_{k=0}^{d} \frac{a_k x^{k+1}}{k+1}} \leq d + 1$$

The claim now follows from Lemma 6.8. □

We now establish another bound on the coordination ratio, following the ideas of Roughgarden [74, Lemma 3.5, 3.6, 3.7, Theorem 3.8]. He bounds the coordination ratio for splittable flows in general networks (i.e. in the Wardrop model) by the *anarchy value*, a value that depends on the latency functions only (and not on the structure of the network graph). Let us first define a corresponding value for our model. We will denote it as the *discrete anarchy value*.

Definition 6.10. *Let $l : [n] \to \mathbb{R}_{\geq 0}$ be a non-decreasing latency function. Define the marginal cost function $l^* : [n] \to \mathbb{R}_{\geq 0}$ corresponding to l by*

$$\begin{aligned} & l^*(1) = l(1), \\ \textit{and} \qquad & l^*(x) = xl(x) - (x-1)l(x-1) \textit{ for } x \in [n]\backslash\{1\}\,. \end{aligned}$$

Furthermore, define the discrete anarchy value of l by

$$\alpha(l) = \max_{r \in [n]} \left[\frac{k(r)l(k(r))}{rl(r)} + 1 - \frac{k(r)}{r} \right]^{-1},$$

where

$$k(r) = \max\{k \in [r] \mid l^*(k) \leq l(r)\} \textit{ for } r \in [n]\,.$$

Finally, define the discrete anarchy value of an instance $I = (n, \mathbf{l})$ of the congestion routing model with identical users by

$$\alpha(I) = \max_{j \in [m]} \alpha(l_j)\,.$$

Note that in the above definition $k(r)$ is well defined, because $l^*(1) = l(1) \leq l(r)$.

We are now ready to state the theorem which bounds the coordination ratio by the discrete anarchy value.

Theorem 6.11. *Let $I = (n, \mathbf{l})$ be an instance of the congestion routing model with identical users and non-decreasing latency functions $l_1, \ldots, l_m$ such that $xl_j(x)$ is convex for all $j \in [m]$. Let A be an assignment at Nash equilibrium, and let A^* be any assignment for I. Then,*

$$SC(A, I) \leq \alpha(I) SC(A^*, I)$$

Proof. Let j be any link with latency function l_j and marginal cost function l_j^*, and let $k_j(r) = \max\{k \in \{1, \ldots, r\} \mid l_j^*(k) \leq l_j(r)\}$ for $r \in [n]$. Let x_j and x_j^* be the flow on link j according to A and A^*, respectively, that is, $x_j = x_j(A)$ and $x_j^* = x_j(A^*)$. Denote $k_j = k_j(x_j)$.

Then, as l_j^* is non-decreasing (because $xl_j(x)$ is convex) and non-negative (because

$xl_j(x)$ is non-decreasing),

$$\begin{aligned}
l_j(x_j^*)x_j^* &= l_j(k_j)k_j + \begin{cases} \sum_{t=k_j+1}^{x_j^*} l_j^*(t) & x_j^* > x_j \\ \sum_{t=k_j+1}^{x_j^*} l_j^*(t) & x_j \geq x_j^* > k_j \\ -\sum_{t=x_j^*+1}^{k_j} l_j^*(t) & x_j^* \leq k_j \end{cases} \\
&\geq l_j(k_j)k_j + \begin{cases} (x_j^* - x_j)l_j^*(x_j+1) + (x_j - k_j)l_j^*(k_j+1) & x_j^* > x_j \\ (x_j^* - k_j)l_j^*(k_j+1) & x_j \geq x_j^* > k_j \\ -(k_j - x_j^*)l_j^*(k_j) & x_j^* \leq k_j \end{cases} \\
&\geq l_j(k_j)k_j + \begin{cases} (x_j^* - x_j)l_j(x_j+1) + (x_j - k_j)l_j(x_j) & x_j^* > x_j \\ (x_j^* - k_j)l_j(x_j) & x_j \geq x_j^* > k_j \\ (x_j^* - k_j)l_j(x_j) & x_j^* \leq k_j \end{cases} \\
&= l_j(k_j)k_j + (x_j^* - k_j)l_j(x_j) + \begin{cases} (x_j^* - x_j)(l_j(x_j+1) - l_j(x_j)) & x_j^* > x_j \\ 0 & x_j \geq x_j^* \end{cases}
\end{aligned}$$

The last estimation makes use of the fact that, by definition of k_j, $l_j^*(k_j) \leq l_j(x_j)$, and that $l_j(x_j) \leq l_j^*(k_j+1)$ if $k_j < n$. Let $L = \max_{j\in[m]} l_j(x_j)$. Then, as A is at Nash equilibrium,

$$l_j(x_j+1) \geq L \text{ for all } j \in [m]\,.$$

Furthermore, note that

$$\sum_{j\in[m]} x_j^* = \sum_{j\in[m]} x_j = n\,.$$

Thus we can bound the social cost of A^* from below by

$$\begin{aligned}
SC(A^*) &= \sum_{j\in[m]} x_j^* l_j(x_j^*) \\
&\geq \sum_{j\in[m]} \big(l_j(k_j)k_j + (x_j^* - k_j)l_j(x_j)\big) + \sum_{\substack{j\in[m] \\ x_j^* > x_j}} (x_j^* - x_j)(l_j(x_j+1) - l_j(x_j)) \\
&= \sum_{j\in[m]} \left(\frac{l_j(k_j)}{l_j(x_j)}k_j + (x_j - k_j)\right) l_j(x_j) + \sum_{j\in[m]} \big(x_j^* - x_j\big)\, l_j(x_j) \\
&\quad + \sum_{\substack{j\in[m] \\ x_j^* > x_j}} (x_j^* - x_j)(l_j(x_j+1) - l_j(x_j))
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j\in[m]} \left(\frac{k_j l_j(k_j)}{x_j l_j(x_j)} + 1 - \frac{k_j}{x_j}\right) x_j l_j(x_j) + \sum_{\substack{j\in[m]\\ x_j^*<x_j}} (x_j^* - x_j) l_j(x_j) \\
&\qquad + \sum_{\substack{j\in[m]\\ x_j^*>x_j}} (x_j^* - x_j) l_j(x_j+1) \\
&\geq \sum_{j\in[m]} \frac{1}{\alpha(l_j)} x_j l_j(x_j) + \sum_{\substack{j\in[m]\\ x_j^*<x_j}} (x_j^* - x_j) L + \sum_{\substack{j\in[m]\\ x_j^*>x_j}} (x_j^* - x_j) L \\
&\geq \frac{1}{\alpha(I)} \sum_{j\in[m]} x_j l_j(x_j) + L \underbrace{\sum_{j\in[m]} (x_j^* - x_j)}_{=0} \\
&= \frac{1}{\alpha(I)} SC(A)\,.
\end{aligned}
$$

□

To give an example we will now bound the discrete anarchy value for linear latency functions. Let the latency function of link j be $l_j(x) = a_j x + b_j$ with $a_j, b_j \in \mathbb{R}_{\geq 0}$. Then,

$$l_j^*(x) = a_j x^2 + b_j x - a_j(x-1)^2 - b_j(x-1) = 2a_j x + b_j - a_j\,.$$

To compute $k_j(r) = \max\{k \in \{1,\dots,r\} \mid l_j^*(k) \leq l_j(r)\}$ for $r \in [n]$, we have to maximize k subject to

$$l_j^*(k) \leq l_j(r) \quad \Leftrightarrow \quad 2a_j k + b_j - a_j \leq a_j r + b_j \quad \Leftrightarrow \quad k \leq \frac{r+1}{2}\,.$$

Hence, $k_j(r) = \lfloor \frac{r+1}{2} \rfloor$. Now we can bound $\alpha(l_j)$ by

$$
\begin{aligned}
\alpha(l_j) &= \max_{r\in[n]} \frac{1}{\lfloor \frac{r+1}{2} \rfloor \frac{1}{r} \frac{a_j \lfloor \frac{r+1}{2} \rfloor + b_j}{a_j r + b_j} + 1 - \lfloor \frac{r+1}{2} \rfloor \frac{1}{r}} \\
&\leq \max_{r\in[n]} \frac{1}{\lfloor \frac{r+1}{2} \rfloor \frac{1}{r} \frac{a_j \lfloor \frac{r+1}{2} \rfloor}{a_j r} + 1 - \lfloor \frac{r+1}{2} \rfloor \frac{1}{r}} \\
&= \max_{r\in[n]} \frac{1}{(\lfloor \frac{r+1}{2} \rfloor \frac{1}{r} - \frac{1}{2})^2 + \frac{3}{4}} \\
&\leq \frac{4}{3}\,.
\end{aligned}
\tag{6.11}
$$

It follows from Theorem 6.11 that for any network of parallel links with linear latency functions, the coordination ratio of pure Nash equilibria is bounded by $\frac{4}{3}$. Because of Theorem 6.1 and as related links are a special case of linear links, we know that this bound is tight. In general, Theorem 6.11 yields only an upper bound on the coordination ratio. However, the following proposition shows that, given an instance I, one can always construct an instance I', consisting of one constant latency link and one link from I, with $\alpha(I') = \alpha(I)$ and coordination ratio $\alpha(I)$.

Proposition 6.12. *Let $I = (n, \mathbf{l})$ be an instance of the congestion routing model with n identical users and discrete anarchy value $\alpha(I)$. Then there exists an instance I' with at most n users and two links and the same discrete anarchy value $\alpha(I') = \alpha(I)$ that has coordination ratio exactly $\alpha(I')$.*

Proof. Let $j \in [m]$ be a link of I with $\alpha(l_j) = \alpha(I)$. Let $k_j(r) = \max\{k \in [r] \mid l_j^*(k) \leq l_j(r)\}$ for $r \in [n]$. Let $n' \in [n]$ be such that

$$\left[\frac{k(n')l_j(k(n'))}{n'l_j(n')} + 1 - \frac{k(n')}{n'}\right]^{-1} = \alpha(l_j)\,.$$

Now construct a new instance I' with n' users and two links with latency functions $l_1'(x) = l_j(n')$ and $l_2'(x) = l_j(x)$ for all $x \in [n']$. That is, link 1 has constant latency and link 2 has the same latency function as link j of instance I.
Note that $\alpha(I') = \max\{\alpha(l_1'), \alpha(l_2')\} = \max\{1, \alpha(l_j)\} = \alpha(l_j) = \alpha(I)$. Theorem 6.11 implies that the coordination ratio for I' is bounded by $\alpha(I')$.
Consider an assignment A for I' where all n' users go to link 2, i.e., $x_1(A) = 0$ and $x_2(A) = n'$. Then $l_1'(x_1(A)) = l_j(n') = l_2'(x_2(A))$ and hence A is at Nash equilibrium. Now consider an assignment A^* with $x_1(A^*) = n' - k(n')$ users on the first link and $x_2(A^*) = k(n')$ users on the second. Then, comparing the social cost of A and A^* yields

$$\begin{aligned}\frac{SC(A)}{SC(A^*)} &= \frac{n'l_j(n')}{(n' - k(n'))l_j(n') + k(n')l_j(k(n'))} \\ &= \left[\frac{k(n')l_j(k(n')}{n'l_j(n')} + 1 - \frac{k(n')}{n'}\right]^{-1} = \alpha(I')\,.\end{aligned}$$

Hence, the coordination ratio of I' is exactly $\alpha(I')$. □

Proposition 6.12 shows that coordination ratio is not a matter of the number of links but only of the links' latency characteristics. Note that the construction of an instance I' with coordination ratio $\alpha(I')$ in the proof of Proposition 6.12 involves a constant latency link and hence is only feasible if links of arbitrary constant latency are allowed. For example it does not yield a lower bound on the coordination ratio for related links (see Theorem 6.1).

The definition of the discrete anarchy value $\alpha(I)$ of an instance I depends on the number n of its users. If we increase n, then also $\alpha(I)$ can increase. It cannot decrease because it is maximized over all possible numbers $1, \ldots, n$ of users. If we let n go to infinity, then either $\alpha(I)$ is unbounded, or it reaches its maximum for some value of n, or it remains below (but gets arbitrarily close to) some supremum. In the first case we can construct an instance with arbitrary coordination ratio as in the proof of Proposition 6.12. The second case is covered by Theorem 6.11 and Proposition 6.12 for the corresponding value of n. In the last case the coordination ratio is bounded by the supremum and we can construct an instance with coordination ratio arbitrarily close to it.

6.1.3 Computation of Pure Nash Equilibrium and Optimal Assignment for General Latency Functions

We complete this section by giving a fast algorithm to compute a pure Nash equilibrium in the congestion routing model with identical users and arbitrary non-decreasing latency functions. By the same algorithm we can also compute a globally optimal assignment, as we will see.

A simple approach to compute a Nash equilibrium is to assign the users one by one to their respective best link. This greedy algorithm can be implemented with running time $\mathcal{O}((n+m)\log m)$ if the links are kept in a priority queue according to their latency after the assignment of the next user. Our algorithm, `computeNash()` (Figure 6.1), takes time $\mathcal{O}(m \log m \log n)$ to compute the link congestions of a Nash equilibrium which is better if the number of users is large compared to the number of links or, more precisely, if $m = o(\frac{n}{\log n})$. Note that additional time $\mathcal{O}(n+m)$ is needed to generate a corresponding assignment.

The algorithm gets as input an arbitrary initial assignment of users to links given by the link congestions $x_1, \ldots, x_m$, i.e., x_j is the number of users on link j. It transforms this assignment into a Nash equilibrium by moving chunks of users at a time. The first chunk contains all users. In each phase the chunk size is cut in half until a chunk consists of one user only.

Proposition 6.13. *Consider the congestion routing model with identical users and arbitrary non-decreasing latency functions. Then algorithm* `computeNash()` *computes a pure Nash equilibrium in time* $\mathcal{O}(m \log m \log n)$.

Proof. Obviously, algorithm `computeNash()` has computed a Nash equilibrium after it terminates, because after the last iteration of the for-loop (with $\delta = 1$), it holds $l_k(x_k) \leq l_j(x_j + 1)$ for all $j, k \in [m]$.

We now consider the running time. We show that each iteration of the for-loop takes time $\mathcal{O}(m \log m)$, if implemented properly. Therefore, during each iteration of the for-loop, the links are kept in two priority queues. In one queue the priorities are according to $l_j(x_j)$, in the other according to $l_j(x_j + \delta)$. For each iteration of the for-loop we prove that a link's congestion is either increased once or decreased an arbitrary number of times.
(a) Assume first, the congestion of some link is increased and then decreased. Consider the first time that this happens on any link, and let t be the corresponding link. Let $x_1 \ldots, x_m$ be the link congestions just before the increment. Then $l_t(x_t + \delta)$ is minimal among all $l_j(x_j + \delta)$ before x_t is increased, and $l_t(x_t + \delta) > l_u(x'_u + \delta)$ for some $u \in [m]$ before it is decreased, where x'_u is the congestion of link u just before the decrement. As $l_t(x_t + \delta)$ was minimal before the increment of x_t (which implies $l_t(x_t + \delta) \leq l_u(x_u + \delta)$), the congestion on link u must have been decreased after x_t was increased (i.e., $x'_u < x_u$). Before the last decrement of link u's congestion the congestion on link u was $x'_u + \delta$ and $l_u(x'_u + \delta)$ was maximal among all links at that time. But this implies $l_u(x'_u + \delta) \geq l_t(x_t + \delta)$, a contradiction.
(b) Now assume, the congestion of some link is decreased and then increased. Consider the first time that this happens on any link, and let s be the corresponding link. Let

$x_1 \ldots, x_m$ be the link congestions just before the decrement. Then $l_s(x_s)$ is maximal among all $l_j(x_j)$ before x_s is decreased, and $l_s(x_s) < l_u(x'_u)$ for some $u \in [m]$ before it is increased, where x'_u is the congestion of link u just before the increment. As $l_s(x_s)$ was maximal before the decrement of x_s, the congestion on link u must have been increased after x_s was decreased. Before the last increment of link u's congestion the congestion on link u was $x'_u - \delta$ and $l_u(x'_u)$ was minimal at that time. But this implies $l_u(x'_u) \leq l_s(x_s)$, a contradiction.
(c) At last assume that there is a link t whose congestion is increased twice during one iteration of the for-loop. Let $x_1 \ldots, x_m$ be the link congestions at the beginning of that iteration. Then, $l_s(x'_s) > l_t(x_t + 2\delta)$ for some s, where x'_s is the congestion on link s just before the second increment of x_t. As the congestion on link s is going to be decreased, it cannot have been increased during this iteration due to **(a)**. Hence, $x'_s \leq x_s$. The post-condition of the last iteration of the for-loop implies $l_s(x_s) \leq l_t(x_t + \delta')$, where δ' is the value of the for-variable in the last iteration. As $\lceil y \rceil \leq 2\lceil \frac{y}{2} \rceil$ for every $y \geq 0$, we have $\delta' \leq 2\delta$ which implies $l_t(x_t + \delta') \leq l_t(x_t + 2\delta)$ and hence $l_s(x_s) \leq l_t(x_t + 2\delta)$. But this implies $l_s(x_s) \leq l_t(x_t + 2\delta) < l_s(x'_s) \leq l_s(x_s)$, a contradiction.

In each iteration of the inner while-loop the congestion of some link is increased. We have just shown (**(a)**,**(b)** and **(c)**) that this can happen only once per link. So, we have at most m iterations of the while-loop during each iteration of the for-loop. As the links are kept sorted, the links with minimal $l_t(x_t + \delta)$ and maximal $l_s(x_s)$, respectively, can be determined in constant time. Updating of the data structures can be done in time $\mathcal{O}(\log m)$. Building up the data structures at the beginning of each iteration of the for-loop takes time $\mathcal{O}(m \log m)$. Thus, each iteration of the for-loop takes time $\mathcal{O}(m \log m)$. The for-loop is iterated $\mathcal{O}(\log n)$ times. The total running time is hence $\mathcal{O}(m \log m \log n)$. □

The following lemma shows that we can compute a globally optimal pure assignment in the same way as a Nash equilibrium, but according to other latency functions, if the cost function $xl_j(x)$ is convex for every link $j \in [m]$. A corresponding result holds for continuous latency functions and splittable flows, too (see e.g. [75]).

Lemma 6.14. *Consider an instance of the congestion routing routing model with identical users and m links with latency function $l_j(x)$ on link j, such that $xl_j(x)$ is convex and non-decreasing for all $j \in [m]$. Set $l^*_j(x) = xl_j(x) - (x-1)l_j(x-1)$ for all $j \in [m]$, and let A be any pure assignment. Then, A is globally optimal with respect to latency functions $l_1, \ldots, l_m$, if and only if A is at Nash equilibrium with respect to latency functions $l^*_1, \ldots, l^*_m$.*

Proof. Let $x = (x_1, \ldots, x_m)$ be the vector of link flows according to A, i.e., $x_j = x_j(A)$. A is at Nash equilibrium according to latency functions $l^*_1, \ldots, l^*_m$, exactly if $l^*_j(x_j) \leq l^*_k(x_k + 1)$ for all $j, k \in [m]$, or, equivalently, if for all $j, k \in [m]$

$$x_j l_j(x_j) - (x_j - 1)l_j(x_j - 1) \leq (x_k + 1)l_k(x_k + 1) - x_k l_k(x_k)\,.$$

This is true, if and only if for all $j, k \in [m]$

$$f_j(x_j) + f_k(x_k) \leq f_j(x_j - 1) + f_k(x_k + 1)\,,$$

```
computeNash(I, x)
Input:   instance I = (n, l) of the congestion routing
         model with identical users
         link congestions x_1, ..., x_m
Output:  Nash equilibrium x_1, ..., x_m

for δ = n, ⌈n/2⌉, ⌈n/4⌉, ..., 1 do
    while ∃ s, t ∈ [n] with x_s ≥ δ and l_s(x_s) > l_t(x_t + δ) do
        let t be such that l_t(x_t + δ) is minimal;
        let s ∈ [m] be such that l_s(x_s) is maximal
            w.r.t.  x_s ≥ δ
        x_s = x_s − δ;  x_t = x_t + δ;
```

Figure 6.1: An algorithm to compute a Nash equilibrium from any assignment for an instance of the congestion routing model with identical users.

where $f_j(x) = xl_j(x)$. Due to Lemma 6.7 this is necessary and sufficient for $SC(A) = \sum_{j\in[m]} f_j(x_j)$ to be minimal. □

Thus, algorithm `computeNash()` can be used to compute a globally optimal pure assignment by applying it to the instance with latency function $l_j^*(x) = xl_j(x) - (x-1)l_j(x-1)$ on link $j \in [m]$.

6.2 Mixed Nash Equilibria

In this section we turn our attention to general (mixed) Nash equilibria in the model of identical users and almost arbitrary link latency functions. Recall that a mixed assignment $\mathbf{P}$ determines for each user $i \in [n]$ and each link $j \in [m]$ the probability p_{ij} for user i to use link j. $\mathbf{P}$ is at Nash equilibrium if no user can decrease its expected individual cost by changing its vector of probabilities (the ith row of $\mathbf{P}$) solely. A pure assignment is hence a special case of a mixed assignment where each probability is either zero or one. On the other hand we can identify the opposite extremal case of a mixed assignment where no probability is zero or one, that is, $p_{ij} \in (0,1)$ for all $i \in [n], j \in [m]$. The latter is called a *fully mixed assignment*. If a fully mixed assignment is at Nash equilibrium, then it is called a *fully mixed Nash equilibrium*. This special Nash equilibrium plays an important role, because the social cost achieves its maximum among all Nash equilibria for a given instance with the fully mixed Nash equilibrium, if latency functions are convex, and provided that the fully mixed Nash equilibrium exists (see Theorem 6.19).
There is strong indication that also for the KP model (i.e., the congestion routing model with related links and social cost defined as SC^{MAX}) the fully mixed Nash equilibrium has maximal social cost, but, unfortunately, this claim remains unproven yet. However, it could be proven for several special cases of the KP model [29, 35, 52, 27] and is conjec-

tured to hold in general [35].

In the first subsection we show that for any instance of our model, the fully mixed Nash equilibrium, if it exists, is unique.
In the next subsection we provide an example that shows that the Fully Mixed Nash Equilibrium Conjecture can only hold for instances with convex latency functions. In fact we show that even pure Nash equilibria may have larger social cost than the fully mixed Nash equilibrium, if the latency functions are not required to be convex.
The last subsection characterizes the instances for which it exists or does not exist. Finally we show that, by removing a subset of the links, one can always obtain an instance for which the fully mixed Nash equilibrium exists and has non-smaller social cost than any other equilibrium in the original instance.

6.2.1 Uniqueness of the Fully Mixed Nash Equilibrium

A fully mixed Nash equilibrium does not necessarily exist for a given instance. But if it exists, then it is unique, and we may hence speak of *the* fully mixed Nash equilibrium. To prove uniqueness we make use of the next two lemmas.

Definition 6.15. *For $r \in \mathbb{N}$, $\mathbf{p} = (p_1, \ldots, p_r) \in \mathbb{R}^r$ and a function $g : \mathbb{N}_0 \to \mathbb{R}$ define*

$$H(\mathbf{p}, r, g) = \sum_{A \subseteq [r]} \prod_{k \in A} p_k \prod_{k \in [r] \setminus A} (1 - p_k) \cdot g(|A|) \,.$$

Lemma 6.16. *[34] For every vector of r probabilities $\mathbf{p} = (p_1, \ldots, p_r) \in [0,1]^r$ and every non-decreasing function $g : \mathbb{N}_0 \to \mathbb{R}$, $H(\mathbf{p}, r, g)$ is non-decreasing in each probability p_i, $i \in [n]$. If, additionally, g is non-constant on $\{0, \ldots, \sum_{i \in [r]} \lceil p_i \rceil\}$ (i.e., $g(0) < g(\sum_{i \in [r]} \lceil p_i \rceil)$), then $H(\mathbf{p}, r, g)$ is strictly increasing in each probability p_i, $i \in [n]$.*

Proof. We prove that $H(\mathbf{p}, r, g)$ is non-decreasing (strictly increasing, respectively) in p_r. The claim then follows by symmetry of $H(\mathbf{p}, r, g)$ in all probabilities $p_i, i \in [r]$. It is

$$\begin{aligned} H(\mathbf{p}, r, g) &= \sum_{A \subseteq [r]} \prod_{k \in A} p_k \prod_{k \in [r] \setminus \notin A} (1 - p_k) \cdot g(|A|) \\ &= \sum_{A \subseteq [r-1]} \prod_{k \in A} p_k \prod_{k \notin A \cup \{r\}} (1 - p_k) \cdot [g(|A|) + p_r \cdot (g(|A| + 1) - g(|A|))] \end{aligned}$$

As $g(|A| + 1) - g(|A|) \geq 0$ for all $A \subseteq [r - 1]$ (g is non-decreasing), $H(\mathbf{p}, r, g)$ is non-decreasing in p_r. Furthermore, if g is non-constant on the interval $\{0, \ldots, \sum_{i \in [r]} \lceil p_i \rceil\}$, then there is some $A \subset [r-1]$ with $p_i > 0$ for all $i \in A$, such that $g(|A|+1) - g(|A|) > 0$, and hence $H(\mathbf{p}, r, g)$ is strictly increasing in p_r. □

Lemma 6.17. *Consider an instance $I = (n, \mathbf{l})$ of the congestion routing model with n identical users and m links with non-decreasing and non-constant (on $[n]$) latency functions. If a fully mixed Nash equilibrium $\mathbf{F}$ exists, then every user is assigned to link $j \in [m]$ with the same probability in $\mathbf{F}$. That is $f_{ij} = f_{hj}$ for all $i, h \in [n], j \in [m]$.*

Proof. For $m = 1$ the claim is trivial. So let $m > 1$. Let i and h be any two users. The expected individual cost of user i on link j is

$$\lambda_i(\mathbf{F}|j) = \sum_{A \subseteq [n] \setminus \{i,h\}} \left[(1 - f_{hj}) l_j(|A| + 1) + f_{hj} l_j(|A| + 2)\right] r(\mathbf{F}, A, \{i, h\}, j)$$

with

$$r(\mathbf{F}, A, \{i, h\}, j) = \prod_{k \in A} f_{kj} \prod_{k \in [n] \setminus (A \cup \{i,h\})} (1 - f_{kj}) .$$

In a fully mixed Nash equilibrium a user experiences the same expected individual latency on all links, i.e., $\lambda_i(\mathbf{F}|j) = \lambda_i$ and $\lambda_h(\mathbf{F}|j) = \lambda_h$ for all $j \in [m]$ for some λ_i and λ_h. Thus, for all $j \in [m]$,

$$\lambda_i - \lambda_h = (f_{hj} - f_{ij}) \sum_{A \subseteq [n] \setminus \{i,h\}} (l_j(|A| + 2) - l_j(|A| + 1)) r(\mathbf{F}, A, \{i, h\}, j) .$$

As $\mathbf{F}$ is fully mixed, $f_{kj} \in (0, 1)$ for all $k \in [n], j \in [m]$. Thus $r(\mathbf{F}, A, \{i, h\}, j) > 0$ for all $A \subseteq [n], i, h \in [n], j \in [m]$. As l_j is non-constant on $[n]$, we have $l_j(|A|+2) - l_j(|A|+1) > 0$ for some $A \subseteq [n] \setminus \{i, h\}$. Thus the sum is positive for all $j \in [m]$. Hence, $\lambda_i > \lambda_h$ ($\lambda_i < \lambda_h$, respectively) implies $f_{hj} > f_{ij}$ ($f_{hj} < f_{ij}$, respectively) for all $j \in [m]$, which contradicts the fact that $\sum_{j \in [m]} f_{hj} = 1 = \sum_{j \in [m]} f_{ij}$. Thus, $\lambda_i = \lambda_h$, which implies $f_{hj} = f_{ij}$ for all $j \in [m]$. □

Theorem 6.18. *Consider an instance $I = (n, \mathbf{l})$ of the congestion routing model with identical users and non-decreasing and non-constant (on $[n]$) latency functions. If a fully mixed Nash equilibrium $\mathbf{F}$ exists, then it is unique.*

Proof. Lemma 6.17 yields that a certain link is used by every user with the same probability in any fully mixed Nash equilibrium $\mathbf{F}$, i.e., $f_{hj} = f_{ij}$ for all $i, h \in [n]$ and $j \in [m]$. We will use this fact to establish uniqueness of the fully mixed Nash equilibrium.
Assume there are two fully mixed Nash equilibria $\mathbf{F}'$ and $\mathbf{F}''$ with $f'_{ij} = f'_j$ and $f''_{ij} = f''_j$ for all $i \in [n], j \in [m]$. The expected individual cost of any user i on link j according to $\mathbf{F}'$ is

$$\begin{aligned} \lambda_i(\mathbf{F}'|j) &= \sum_{A \subseteq [n] \setminus \{i\}} \prod_{k \in A} f'_k \prod_{k \in [n] \setminus A \cup \{i\}} (1 - f'_k) \cdot l_j(|A| + 1) \\ &= H(\underbrace{(f'_j, \ldots, f'_j)}_{\in \mathbb{R}^{n-1}}, n - 1, g_j) , \end{aligned}$$

where $g_j : [n] \cup \{0\} \to \mathbb{R}$ is defined by $g_j(x) = l_j(x + 1)$.

Due to Lemma 6.16, $\lambda_i(\mathbf{F}'|j) = H((f'_j, \ldots, f'_j), n - 1, g_j)$ is strictly increasing in f'_j for all $i \in [n]$. Similarly, $\lambda_i(\mathbf{F}''|j) = H((f''_j, \ldots, f''_j), n - 1, g_j)$ is strictly increasing in f''_j for any $i \in [n]$. If $\mathbf{F}'$ and $\mathbf{F}''$ are different, then there exist two links $j, k \in [m]$ with $f'_j > f''_j$ and $f'_k < f''_k$. But this implies $\lambda_i(\mathbf{F}'|j) > \lambda_i(\mathbf{F}''|j) = \lambda_i(\mathbf{F}''|k) > \lambda_i(\mathbf{F}'|k)$ for any $i \in [n]$, and hence contradicts the Nash equilibrium condition for $\mathbf{F}'$. This completes the proof. □

6.2.2 Convex versus Non-convex Latency Functions

For instances of the routing model under consideration where all latency functions are convex the following theorem applies.

Theorem 6.19. *[34] Consider an instance I of the congestion routing model with identical users and parallel links with non-decreasing and convex latency functions. If the fully mixed Nash equilibrium $\mathbf{F}$ exists, then for every mixed Nash equilibrium $\mathbf{P}$, $SC(\mathbf{P}, I) \leq SC(\mathbf{F}, I)$.*

Thus, for instances with convex latency functions, the fully mixed Nash equilibrium, provided that it exists, has the worst social cost among all equilibria. Now we show that it is necessary to require convex latency functions, as the claim might not hold otherwise, even for identical links.

Proposition 6.20. *There exists an instance $I = (n, \mathbf{l})$ of the congestion routing model with identical users and identical links with a non-decreasing, non-convex latency function for which a pure Nash equilibrium A and a fully mixed Nash equilibrium $\mathbf{F}$ exist, such that $\lambda_i(A, I) > \lambda_i(\mathbf{F}, I)$ for all $i \in [n]$.*

Proof. Consider an instance with $m = 2$ links and $n = 4$ (unit sized) users. Define l, the common latency function of both links, as follows: $l(1) = 1, l(2) = 2, l(3) = 2+\epsilon, l(4) = 2 + 2\epsilon$, where $\epsilon > 0$. Then, any pure Nash equilibrium A assigns exactly 2 users to each link, and hence $\lambda_i(A) = 2$ for all $i \in [n]$. Now, consider the fully mixed Nash equilibrium $\mathbf{F}$. Here $f_{ij} = \frac{1}{2}$ for all $i \in [n], j \in [m]$. It follows that

$$\lambda_i(\mathbf{F}) = \frac{1}{8}(l(1) + 3l(2) + 3l(3) + l(4)) = \frac{15}{8} + \frac{5\epsilon}{8} \quad \text{for all } i \in [n]\,.$$

For $\epsilon < \frac{1}{5}$ we have $\lambda_i(A) > \lambda_i(\mathbf{F})$ for all $i \in [n]$. □

Note that because of Lemma 2.6, for the example given in Proposition 6.20 the social cost of the pure Nash equilibrium exceeds that of the fully mixed Nash equilibrium.

6.2.3 Existence of Fully Mixed Nash Equilibrium

For instances with identical links, that is, where all links $j \in [m]$ have the same latency function $l_j = l$, a fully mixed Nash equilibrium $\mathbf{F}$ always exists and has probabilities $f_{ij} = \frac{1}{m}$ for all $i \in [n]$, $j \in [m]$. In general, the existence of the fully mixed Nash equilibrium is not granted, but depends on the latency functions $l_1, \ldots, l_m$. We will now shed light on this dependence.

Without loss of generality, we assume the links to be ordered non-decreasingly according to $l_j(1)$. Furthermore, we assume l_j to be non-constant on $[n]$, i.e., $l_j(n) > l_j(1)$.

Definition 6.21. *For an instance $I = (n, \mathbf{l})$ of the congestion routing model with identical users and ordered (non-decreasingly according to $l_j(1)$) and non-constant (on $[n]$) latency functions, that is, $l_1(1) \leq \cdots \leq l_m(1)$ and $l_j(1) < l_j(n)$ for all $j \in [m]$, let g_j :*

$[n-1] \cup \{0\} \to \mathbb{R}$ *be defined by* $g_j(x) = l_j(x+1)$ *for all* $j \in [m]$. *For* $k \in [m]$, $j \in [k-1]$, *define* $p_j(k) \in \mathbb{R}$, *such that*

$$H(\underbrace{(p_j(k), \ldots, p_j(k))}_{\in \mathbb{R}^{n-1}}, n-1, g_j) = \min\{l_k(1), l_j(n)\}\,.$$

Due to Lemma 6.16, $H((p_j(k), \ldots, p_j(k)), n-1, g_j)$ is strictly increasing in $p_j(k)$ for $p_j(k) \in (0,1)$. Furthermore, it is $H((0,\ldots,0), n-1, g_j) = l_j(1) \leq l_k(1)$ and $H((1,\ldots,1), n-1, g_j) = l_j(n)$ for $j \in [k-1]$. Hence, for every $j \in [k-1]$, $p_j(k) \in [0,1]$ is uniquely determined by the above definition. Note that $H((p_j(k), \ldots, p_j(k)), n-1, g_j)$ is the expected individual latency of any user on link $j \in [m]$, if $p_{ij} = p_j(k)$ for all $i \in [n]$.

Definition 6.22. *Let* $I = (n, \mathbf{l})$ *be an instance of the congestion routing model with identical users and non-constant (on* $[n]$*) latency functions. Without loss of generality let the links be ordered non-decreasingly according to* $l_j(1)$. *Links* $k \in [m]$ *with* $\sum_{j\in[k-1]} p_j(k) > 1$ *or* $\exists j \in [k-1] : l_j(n) < l_k(1)$ *are called* dead links. *Links* $k \in [m]$ *with* $\sum_{j\in[k-1]} p_j(k) = 1$ *and* $\forall j \in [k-1] : l_j(n) < l_k(1)$ *are called* special links.

Lemma 6.23. *[34] Let* $g : \mathbb{N}_0 \to \mathbb{R}$ *be a convex function, and let* $\mathbf{p} = (p_1, \ldots, p_r) \in [0,1]^r$ *be a vector of probabilities. Set* $\tilde{p} = \frac{\sum_{i\in[r]} p_i}{r}$. *Then,*

$$H(\mathbf{p}, r, g) \leq H(\underbrace{(\tilde{p}, \ldots, \tilde{p})}_{\in \mathbb{R}^r}, r, g)\,.$$

Lemma 6.24. *Let* $I = (n, \mathbf{l})$ *be an instance of the congestion routing model with identical users and non-decreasing, non-constant latency functions. If* $j \in [m]$ *is a dead link, then in any Nash equilibrium* $\mathbf{P}$ *for* I, $p_{ij} = 0$ *for all* $i \in [n]$.

Proof. Let $\mathbf{P}$ be a Nash equilibrium. Denote $\Theta_{ij} = \Theta_{ij}(\mathbf{P}) = \sum_{t\in[n]\setminus\{i\}} p_{tj}$ for $i \in [n], j \in [m]$, and define $g_j : [n-1] \cup \{0\} \to \mathbb{R}$ by $g_j(x) = l_j(x+1)$ for all $j \in [m]$. Note, that $\sum_{j=1}^{m} \frac{\Theta_{ij}}{n-1} = 1$ for all $i \in [n]$.
Assume $p_{ik} > 0$ for some user $i \in [n]$ and some dead link k. Then, for all $j \in [m]$

$$l_k(1) \leq \lambda_i(\mathbf{P}|k) \leq \lambda_i(\mathbf{P}|j) \tag{6.12}$$

$$= H(\mathbf{p}(j), n-1, g_j) \overset{Lemma\ 6.23}{\leq} H((\frac{\Theta_{ij}}{n-1}, \ldots, \frac{\Theta_{ij}}{n-1}), n-1, g_j)\,, \tag{6.13}$$

where $\mathbf{p}(j) = (p_{1,j}, \ldots, p_{i-1,j}, p_{i+1,j}, \ldots, p_{n,j})$. Since k is a dead link, we have

$$\sum_{j\in[k-1]} p_j(k) > 1 \quad \text{or} \quad \exists j \in [k-1] : l_j(n) < l_k(1)\,.$$

In the latter case we get that $\lambda_i(\mathbf{P}|j) \leq l_j(n) < l_k(1)$ for some $j \in [k-1]$, a contradiction with (6.12). In the first case,

$$\sum_{j=1}^{k-1} \frac{\Theta_{ij}}{n-1} \leq 1 < \sum_{j=1}^{k-1} p_j(k)\,.$$

Hence there must exist some $j \in [k-1]$, such that $p_j(k) > \frac{\Theta_{ij}}{n-1}$. This implies

$$H((\frac{\Theta_{ij}}{n-1},\dots,\frac{\Theta_{ij}}{n-1}),n-1,g_j) \overset{Lemma\ 6.16}{<} H((p_j(k),\dots,p_j(k)),n-1,g_j) = l_k(1)\,,$$

a contradiction with (6.13). □

Lemma 6.25. *Consider an instance $I = (n,\mathbf{l})$ of the congestion routing model with identical users and non-decreasing, non-constant latency functions. Let S be the set of special links. In any Nash equilibrium $\mathbf{P}$, there exists at most one user i with $p_{ij} > 0$ for some $j \in S$.*

Proof. Let $\mathbf{P}$ be a Nash equilibrium. Denote $\Theta_{ij} = \Theta_{ij}(\mathbf{P}) = \sum_{t\in[n]\setminus\{i\}} p_{tj}$ for $i \in [n], j \in [m]$, and define $g_j : [n-1]\cup\{0\} \to \mathbb{R}$ by $g_j(x) = l_j(x+1)$ for all $j \in [m]$. Assume $p_{ik} > 0$ and $p_{hr} > 0$ for two users $i,h \in [n], i \neq h$, and special links $k,r \in S$ (k may be equal to r). Hence, $\Theta_{ir} \geq \frac{p_{hr}}{n-1} > 0$. Without loss of generality, $k \leq r$. Then for all $j \in [m]$

$$\begin{aligned} l_k(1) &\leq \lambda_i(\mathbf{P}|k) \leq \lambda_i(\mathbf{P}|j) \\ &= H(\mathbf{p}(j),n-1,g_j) \overset{Lemma\ 6.23}{\leq} H((\frac{\Theta_{ij}}{n-1},\dots,\frac{\Theta_{ij}}{n-1}),n-1,g_j)\,, \end{aligned} \tag{6.14}$$

where $\mathbf{p}(j) = (p_{1,j},\dots,p_{i-1,j},p_{i+1,j},\dots,p_{n,j})$. Since $\Theta_{ir} > 0$,

$$\sum_{j=1}^{k-1}\frac{\Theta_{ij}}{n-1} < \sum_{j=1}^{m}\frac{\Theta_{ij}}{n-1} = 1 = \sum_{j=1}^{k-1} p_j(k)\,.$$

Hence, there exists some link $j \in [k-1]$, such that $p_j(k) > \frac{\Theta_{ij}}{n-1}$. This implies

$$H((\frac{\Theta_{ij}}{n-1},\dots,\frac{\Theta_{ij}}{n-1}),n-1,g_j) \overset{Lemma\ 6.16}{<} H((p_j(k),\dots,p_j(k)),n-1,g_j) = l_k(1)\,,$$

a contradiction with (6.14). □

Theorem 6.26. *Consider an instance $I = (n,\mathbf{l})$ of the congestion routing model with (at least two) identical users and non-decreasing, non-constant latency functions. Then there exists a fully mixed Nash equilibrium, if and only if the instance contains no special and no dead links.*

Proof. In a fully mixed Nash equilibrium, every user chooses each link with non-zero probability. Therefore, as a direct consequence of Lemma 6.24 and Lemma 6.25, a fully mixed Nash equilibrium cannot exist, if $n \geq 2$ and the instance possesses dead or special links.
Now, assume there are no dead or special links. We will show that there exists a fully mixed Nash equilibrium $\mathbf{F}$ with probabilities $f_{ij} = f_j$ for $i \in [n], j \in [m]$.

Define $g_j : [n-1]\cup\{0\} \to \mathbb{R}$ by $g_j(x) = l_j(x+1)$ for all $j \in [m]$. Let $\hat{x}_j = H((1,\dots,1),n-1,g_j) - l_m(1)$ for $j \in [m-1]$. For $x \in [0,\hat{x}_j]$ determine $f_j(x)$ by

$H((f_j(x),\dots,f_j(x)),n-1,g_j) = l_m(1)+x$ for $j \in [m-1]$, and for $x \in [0, \min_{j\in[m-1]} \hat{x}_j]$ set $f_m(x) = 1-\sum_{j=1}^{m-1} f_j(x)$. Note, that $f_j(x)$ is strictly increasing in x for all $x \in [0,\hat{x}_j]$, $j \in [m-1]$ because of Lemma 6.16, and therefore $f_m(x)$ is strictly decreasing in x for $x \in [0, \min_{j\in[m-1]} \hat{x}_j]$.
Define $\hat{x}$ by $\hat{x} = \max\{x \in [0, \min_{j\in[m-1]} \hat{x}_j] \mid f_m(x) \geq 0\}$. Then, the function $H((f_m(x),\dots,f_m(x)),n-1,g_m)$ is strictly decreasing in x for $x \in [0,\hat{x}]$.
As link m is neither a special nor a dead link, it holds that $f_m(0) = 1-\sum_{j\in[m-1]} p_j(m) > 0$. This yields $H((f_m(0),\dots,f_m(0)),n-1,g_m) > l_m(1)$ (because $H((0,\dots,0),n-1,g_m) = l_m(1)$ and $H((p,\dots,p),n-1,g_m)$ is strictly increasing in p for $p \in (0,1)$). On the other hand, as $f_m(\hat{x}) = 0$, $H((f_m(\hat{x}),\dots,f_m(\hat{x})),n-1,g_m) = l_m(1) < l_m(1)+\hat{x}$.

Since $H((f_m(x),\dots,f_m(x)),n-1,g_m)$ is a continuous function in x, we can follow from the intermediate value theorem that there must exist some $x_0 \in (0,\hat{x})$, such that $H((f_m(x_0),\dots,f_m(x_0)),n-1,g_m) = l_m(1)+x_0$. Setting $f_{ij} = f_j(x_0)$ for all $i \in [n], j \in [m]$ yields the desired fully mixed Nash equilibrium. □

Theorem 6.26 implies that if the fully mixed Nash equilibrium does not exist, then the instance contains dead or special links. But dead links are never used in any Nash equilibrium and may be removed from the instance. Special links can only be used by a single user. We now broaden the result from Theorem 6.19 by giving an upper bound on the social cost of any Nash equilibrium for any instance, including those that do not possess a fully mixed Nash equilibrium.

Theorem 6.27. *Consider an instance $I = (n,\mathbf{l})$ of the congestion routing model with identical users and non-decreasing, non-constant and convex latency functions. Then the social cost of any Nash equilibrium $\mathbf{P}$ is bounded by the social cost of the fully mixed Nash equilibrium $\mathbf{F}$ for the instance induced by the non-special and non-dead links.*

Proof. Consider any Nash equilibrium $\mathbf{P}$. Let $\mathbf{F}$ be the fully mixed Nash equilibrium for the instance where the links are restricted to the links which are not special and not dead. Due to Lemma 6.24 no user chooses a dead link with non-zero probability according to $\mathbf{P}$. If no user chooses a special link with non-zero probability, then the claim follows directly from Theorem 6.19. So assume that there exists a user that is assigned to a special link with non-zero probability. Lemma 6.25 shows that there is at most one such user.

Without loss of generality, let $[r]$ be the set of non-special and non-dead links. Let $i \in [n]$ be any user. Denote $\Theta_{ij}(\mathbf{P}) = \sum_{k\in[n]\setminus\{i\}} p_{kj}$ and $\Theta_{ij}(\mathbf{F}) = \sum_{k\in[n]\setminus\{i\}} f_{kj}$ for all $j \in [m]$. Note that $\Theta_{ij}(\mathbf{F}) = (n-1)f_{kj}$ for all $k \in [n]$ because of Lemma 6.17. It is $\sum_{j\in[r]} \Theta_{ij}(\mathbf{P}) \leq n-1 = \sum_{j\in[r]} \Theta_{ij}(\mathbf{F})$. This implies that there exists a link $j \in [r]$ where $\Theta_{ij}(\mathbf{P}) \leq \Theta_{ij}(\mathbf{F})$. Let $g_j : \mathbb{N}\cup\{0\} \to \mathbb{R}$ be defined by $g_j(x) = l_j(x+1)$ for all

$j \in [m]$. Then,

$$\begin{aligned}
\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{P}|j) &= H((p_{1j}, \ldots, p_{i-1,j}, p_{i+1,j}, \ldots, p_{nj}), n-1, g_j) \\
&\overset{Lemma\ 6.23}{\leq} H((\frac{\Theta_{ij}(\mathbf{P})}{n-1}, \ldots, \frac{\Theta_{ij}(\mathbf{P})}{n-1}), n-1, g_j) \\
&\overset{Lemma\ 6.16}{\leq} H((\frac{\Theta_{ij}(\mathbf{F})}{n-1}, \ldots, \frac{\Theta_{ij}(\mathbf{F})}{n-1}), n-1, g_j) \\
&\overset{Lemma\ 6.17}{=} H((f_{1j}, \ldots, f_{i-1,j}, f_{i+1,j}, \ldots, f_{nj}), n-1, g_j) \\
&= \lambda_i(\mathbf{F}|j) = \lambda_i(\mathbf{F})\,.
\end{aligned}$$

Thus, $SC(\mathbf{P}) = \sum_{i\in[n]} \lambda_i(\mathbf{P}) \leq \sum_{i\in[n]} \lambda_i(\mathbf{F}) = SC(\mathbf{F})$. □

— 7 —

Unrelated Links

In this chapter we address non-cooperative routing on unrelated parallel links. In contrast to the KP model with related links, here links may process different user flows at different speeds. Related links are a special case of unrelated links. Hence, the unrelated routing model is a generalization of the KP model.

We will explore the scope of the Fully Mixed Nash Equilibrium Conjecture (which is originally stated for the KP model [35]) for unrelated links. We prove that it holds for certain special cases of the unrelated routing model. On the other hand we give a minimal instance for which it does not hold.

7.1 Definition

For unrelated links it is not possible anymore to assign one speed to each link which holds for all users. In fact we must specify one vector of link speeds for each user. If s_{ij} is the speed at which link j processes the traffic of user i, then the latency on link j according to some pure assignment A is $l_j(A) = \sum_{A(i)=j} \frac{w_i}{s_{ij}}$. For convenience we incorporate the user weights and respective link speeds into one value and define $c_{ij} = \frac{w_i}{s_{ij}}$ to be the contribution of user i to the latency on link j. So, an instance of the *unrelated routing model* is given by $I = (\mathbf{C})$, where $\mathbf{C} = (c_{ij}) \in \mathbb{R}_{\geq 0}^{n \times m}$ is the *cost matrix*. Note that we have $c_{ij} = \frac{w_i}{s_j}$ for the special case of related links. The definition of social cost remains as for the KP model. That is, $SC(\mathbf{P}) = SC^{MAX}(\mathbf{P})$.

7.2 Pure versus Fully Mixed Nash Equilibrium

Our first result is that the social cost of a pure Nash equilibrium can never exceed that of a fully mixed Nash equilibrium, if the number of users is not larger than the number of links. This is a consequence of the corresponding claim for the individual cost that we prove first.

Proposition 7.1. *Consider an instance $I = (\mathbf{C})$ of the unrelated routing model where $n \leq m$ and for which a fully mixed Nash equilibrium $\mathbf{F}$ exists. Let $\mathbf{P}$ be any pure Nash equilibrium. Then, for any user i, $\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{F})$.*

Proof. Let A be the assignment corresponding to $\mathbf{P}$, i.e., $(A(i) = j \Leftrightarrow p_{ij} = 1)$ for all $i \in [n], j \in [m]$.

1. Assume first that each user is assigned to a different link, that is, $A(i) \neq A(k)$ for all $i, k \in [n]$ with $i \neq k$. By definition of individual cost, this implies that $\lambda_i(\mathbf{P}) = c_{ij}$, where $j = A(i)$ is the link to which player i is assigned by $\mathbf{P}$. However, $\lambda_i(\mathbf{F}) = c_{ij} + \sum_{k \in [n], k \neq i} f_{kj} c_{kj}$, and hence $\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{F})$.

2. Assume now that not all players are assigned to different links by $\mathbf{P}$. Since $n \leq m$, this implies that there is some empty link $j \in [m]$, that is, no player is assigned to link j by $\mathbf{P}$. By definition of individual cost and the Nash equilibrium property, $\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{P}|j) = c_{ij}$, while $\lambda_i(\mathbf{F}) = c_{ij} + \sum_{k \in [n], k \neq i} f_{kj} c_{kj}$, and hence $\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{F})$.

□

Theorem 7.2. *Consider an instance $I = (\mathbf{C})$ of the unrelated routing model where $n \leq m$, and for which a fully mixed Nash equilibrium $\mathbf{F}$ exists. Let $\mathbf{P}$ be any pure Nash equilibrium. Then, $SC(\mathbf{P}) \leq SC(\mathbf{F})$.*

Proof. Let k be a user with maximal individual cost according to pure Nash equilibrium $\mathbf{P}$. Then the social cost of $\mathbf{P}$ can be written as $SC(\mathbf{P}) = \max_{i \in [n]} \lambda_i(\mathbf{P}) = \lambda_k(\mathbf{P})$. Proposition 7.1 yields $\lambda_k(\mathbf{P}) \leq \lambda_k(\mathbf{F})$. The individual cost of user k according to $\mathbf{F}$ can be expressed by $\lambda_k(\mathbf{F}) = \sum_{A:[n]\to[m]} \prod_{i \in [n]} f_{i,A(i)} \lambda_k(A)$. Hence,

$$\begin{aligned}
\lambda_k(\mathbf{F}) &\leq \sum_{A:[n]\to[m]} \prod_{i \in [n]} f_{i,A(i)} \max_{r \in [n]} \lambda_r(A) \\
&= \sum_{A:[n]\to[m]} \prod_{i \in [n]} f_{i,A(i)} SC(A) \\
&= SC(\mathbf{F}) \,,
\end{aligned}$$

and the claim follows. □

In Section 7.4 we will see that neither of Proposition 7.1 and Theorem 7.2 remains true, if we drop the assumption that the number of users does not exceed the number of links. So, we cannot expect to get a stronger result in this direction.

7.3 Mixed Nash Equilibria of Two Users

We now investigate mixed Nash equilibria that involve only two users. The first result is the correspondent of Proposition 7.1 for this case. The second shows validity of the Fully Mixed Nash Equilibrium Conjecture for two users on two links.

Proposition 7.3. *Consider an instance $I = (\mathbf{C})$ of the unrelated routing model with $n = 2$ users for which the fully mixed Nash equilibrium $\mathbf{F}$ exists. Let $\mathbf{P}$ be any Nash equilibrium. Then, for any user $i \in [2]$, $\lambda_i(\mathbf{P}) \leq \lambda_i(\mathbf{F})$.*

Proof. We proceed by case analysis on the relation between $support(1)$ and $support(2)$.

1. Assume first that $support(1) = support(2)$. There are two subcases to consider.

   a) If $support(1) = support(2) = [m]$, then $p_{ij} = f_{ij}$ for all $i \in [2], j \in [m]$ with $c_{ij} > 0$ which follows from the Nash equilibrium conditions for $\mathbf{P}$ and $\mathbf{F}$ and the definition of individual cost. Note that the probabilities p_{ij} and f_{ij} for i and j with $c_{ij} = 0$ vanish in the computation of individual cost (see Lemma 2.5) according to $\mathbf{P}$ and $\mathbf{F}$, respectively. Hence, $\lambda_i(\mathbf{P}) = \lambda_i(\mathbf{F})$ for $i \in [2]$.

   b) If $support(1) = support(2) \neq [m]$, then there exists some link $k \in [m]$ such that both, $k \notin support(1)$ and $k \notin support(2)$. Note that $\lambda_1(\mathbf{P}) \leq \lambda_1(\mathbf{P}|k) = c_{1k}$, while $\lambda_1(\mathbf{F}) = \lambda_1(\mathbf{F}|k) = c_{1k} + f_{2k}c_{2k}$. Since $f_{2k} > 0$ and $c_{2k} \geq 0$, it follows that $\lambda_1(\mathbf{P}) \leq \lambda_1(\mathbf{F})$, as needed. Corresponding arguments for user 2 establish that $\lambda_2(\mathbf{P}) \leq \lambda_2(\mathbf{F})$, as well.

2. Assume now that $support(1) \neq support(2)$. Then, either we have $support(1) \setminus support(2) \neq \emptyset$ or $support(2) \setminus support(1) \neq \emptyset$.

   a) Consider first the subcase where $support(1) \setminus support(2) \neq \emptyset$. Then, there exists some link $k \in [m]$ such that $k \in support(1)$ while $k \notin support(2)$. Note that $\lambda_1(\mathbf{P}) = \lambda_1(\mathbf{P}|k) = c_{1k}$ while $\lambda_1(\mathbf{F}) = \lambda_1(\mathbf{F}|k) = c_{1k} + f_{2k}c_{2k}$. Since $f_{2k} > 0$ while $c_{2k} \geq 0$, it follows that $\lambda_1(\mathbf{P}) \leq \lambda_1(\mathbf{F})$. We proceed to consider user 2. To show that $\lambda_2(\mathbf{P}) \leq \lambda_2(\mathbf{F})$, assume, by way of contradiction, that $\lambda_2(\mathbf{P}) > \lambda_2(\mathbf{F})$. For all links $j \in support(2)$ we have $\lambda_2(\mathbf{P}) = \lambda_2(\mathbf{P}|j) = c_{2j} + p_{1j}c_{1j}$ and $\lambda_2(\mathbf{F}) = \lambda_2(\mathbf{F}|j) = c_{2j} + f_{1j}c_{1j}$. Since $\lambda_2(\mathbf{P}) > \lambda_2(\mathbf{F})$, it follows that $p_{1j} > f_{1j}$ for all $j \in support(2)$. Since, however, $\sum_{j=1}^{m} p_{1j} = 1 = \sum_{j=1}^{m} f_{1j}$, this implies that there exists some link $k \notin support(2)$ such that $p_{1k} < f_{1k}$. Thus, $\lambda_2(\mathbf{P}) \leq \lambda_2(\mathbf{P}|k) = c_{2k} + p_{1k}c_{1k} \leq c_{2k} + f_{1k}c_{1k} = \lambda_2(\mathbf{F}|k) = \lambda_2(\mathbf{F})$, a contradiction. It follows that $\lambda_2(\mathbf{P}) \leq \lambda_2(\mathbf{F})$.

   b) Corresponding arguments suffice to establish that both, $\lambda_1(\mathbf{P}) \leq \lambda_1(\mathbf{F})$ and $\lambda_2(\mathbf{P}) \leq \lambda_2(\mathbf{F})$ in the subcase where $support(2) \setminus support(1) \neq \emptyset$ as well.

□

Proposition 7.3 is tight in the sense that the claim is not valid for instances with more than two users (see Theorem 7.5).

Theorem 7.4. *For the set of instances of the unrelated routing model where $m = n = 2$ the Fully Mixed Nash Equilibrium Conjecture is valid.*

Proof. Let $I = (\mathbf{C})$ be an instance of the model with $n = m = 2$ for which the fully mixed Nash equilibrium $\mathbf{F}$ exists. We will show that for any Nash equilibrium $\mathbf{P}$, $SC(\mathbf{P}) \leq SC(\mathbf{F})$. We proceed by case analysis on the supports of $\mathbf{P}$.

1. Assume first that $\mathbf{P}$ is pure. Since $m = n$, Theorem 7.2 implies that $SC(\mathbf{P}) \leq SC(\mathbf{F})$, as needed.

2. Assume now that one user is pure and the other one is mixed in $\mathbf{P}$. Without loss of generality, assume that user 1 is pure and that it is assigned to link 1, while user 2 is mixed. We calculate the individual cost of user 2 in $\mathbf{P}$. Clearly, $\lambda_2(\mathbf{P}|1) = c_{21} + c_{11}$ and $\lambda_2(\mathbf{P}|2) = c_{22}$. Since $\mathbf{P}$ is at Nash equilibrium, $\lambda_2(\mathbf{P}|1) = \lambda_2(\mathbf{P}|2)$ or $c_{21} + c_{11} = c_{22}$. The social cost of $\mathbf{P}$ is

$$\begin{aligned} SC(\mathbf{P}) &= p_{21}(c_{11} + c_{21}) + p_{22} \max\{c_{11}, c_{22}\} \\ &= p_{21}c_{22} + p_{22}c_{22} \quad \text{(since } c_{22} = c_{11} + c_{21} \geq c_{11}\text{)} \\ &= c_{22}\,. \end{aligned}$$

   Thus,

$$\begin{aligned} SC(\mathbf{F}) &= \sum_{A:[2]\to[2]} \prod_{i\in[2]} f_{i,A(i)} \max_{k\in[2]} \lambda_k(A) \\ &\geq \sum_{A:[2]\to[2]} \prod_{i\in[2]} f_{i,A(i)} \lambda_2(A) \\ &= \lambda_2(\mathbf{F}) \\ &= \lambda_2(\mathbf{F}|2) \quad \text{(since } \mathbf{F} \text{ is fully mixed)} \\ &\geq c_{22} \quad \text{(since } \lambda_2(\mathbf{F}|2) = c_{22} + f_{12}c_{12}\text{)} \qquad (7.1) \\ &= SC(\mathbf{P})\,, \end{aligned}$$

   as needed.

3. Now assume that both users are not pure in $\mathbf{P}$, i.e., $support(1) = support(2) = \{1, 2\}$. We distinguish two subcases:

   a) Let $C > 0$, i.e., $c_{ij} > 0$ for all $i, j \in [2]$. This implies that the fully mixed Nash equilibrium is unique. Hence, as $\mathbf{P}$ is fully mixed, $\mathbf{P} = \mathbf{F}$ and nothing is to show.

   b) Now assume that $C > 0$ does not hold, i.e., $c_{ij} = 0$ for some $i, j \in [2]$. Without loss of generality let $c_{12} = 0$ (all other cases are symmetric). Then the Nash equilibrium condition for $\mathbf{P}$ yields $\lambda_1(\mathbf{P}|1) = \lambda_1(\mathbf{P}|2)$ and $\lambda_2(\mathbf{P}|1) = \lambda_2(\mathbf{P}|2)$, or equivalently

$$\begin{aligned} c_{11} + p_{21}c_{21} &= c_{12} + p_{22}c_{22} \quad \text{and} \\ c_{21} + p_{11}c_{11} &= c_{22} + p_{12}c_{12}\,. \end{aligned}$$

Applying $p_{22} = 1 - p_{21}$, $p_{12} = 1 - p_{11}$ and $c_{12} = 0$ yields

$$p_{21}(c_{21} + c_{22}) = c_{22} - c_{11} \quad \text{and}$$
$$p_{11}c_{11} = c_{22} - c_{21}\,.$$

The social cost of $\mathbf{P}$ is

$$\begin{aligned} SC(\mathbf{P}) = {} & p_{21}[p_{11}(c_{21} + c_{11}) + (1 - p_{11})\max\{c_{21}, c_{12}\}] \\ & + (1 - p_{21})[p_{11}\max\{c_{22}, c_{11}\} + (1 - p_{11})(c_{22} + c_{12})]\,. \end{aligned}$$

Since $c_{12} = 0$, and as $p_{21} \geq 0$ implies $c_{22} \geq c_{11}$, we have

$$\begin{aligned} SC(\mathbf{P}) &= p_{21}[p_{11}(c_{21} + c_{11}) + (1 - p_{11})c_{21}] \\ &\quad + (1 - p_{21})[p_{11}c_{22} + (1 - p_{11})(c_{22} + c_{12})] \\ &= p_{21}[p_{11}c_{11} + c_{21}] + (1 - p_{21})c_{22} \\ &= p_{21}[c_{22} - c_{21} + c_{21}] + (1 - p_{21})c_{22} \\ &= c_{22}\,. \end{aligned}$$

As $SC(\mathbf{F}) \geq c_{22}$ (see (7.1)), the claim follows.

Thus, in all cases $SC(\mathbf{P}) \leq SC(\mathbf{F})$, as needed. □

7.4 The Limit of the Fully Mixed Nash Equilibrium Conjecture

We finally show that the Fully Mixed Nash Equilibrium Conjecture does not hold for unrelated links in general. Moreover, even a pure Nash equilibrium can have greater social cost than a fully mixed Nash equilibrium.

Theorem 7.5. *There exists an instance I of the unrelated routing model with $n = 3$ users and a fully mixed Nash equilibrium $\mathbf{F}$ and a pure Nash equilibrium $\mathbf{P}$, such that $SC(\mathbf{P}) > SC(\mathbf{F})$.*

Proof. We simply define an instance $I = (\mathbf{C})$ with $n = 3$ users and $m = 2$ links and show that it has a fully mixed Nash equilibrium $\mathbf{F}$ and a pure Nash equilibrium $\mathbf{P}$ with $SC(\mathbf{P}) > SC(\mathbf{F})$.

- Set $c_{11} = c_{21} = 2$, $c_{31} = 0$ and $c_{12} = c_{22} = \frac{3}{2}$ and $c_{32} = 3$.

Consider first the pure assignment $\mathbf{P}$ assigning users 1 and 2 to link 1, and user 3 to link 2. We verify that $\mathbf{P}$ is at Nash equilibrium:

- $\lambda_1(\mathbf{P}|1) = c_{11} + c_{21} = 4 < \frac{9}{2} = c_{12} + c_{32} = \lambda_1(\mathbf{P}|2)$.
- $\lambda_2(\mathbf{P}|1) = c_{21} + c_{11} = 4 < \frac{9}{2} = c_{22} + c_{32} = \lambda_2(\mathbf{P}|2)$.

- $\lambda_3(\mathbf{P}|2) = c_{32} = 3 < 4 = c_{31} + c_{11} + c_{21} = \lambda_3(\mathbf{P}|1)$.

It follows that no user can improve by switching to the opposite link, and hence, $\mathbf{P}$ is at Nash equilibrium. The social cost of $\mathbf{P}$ is $SC(\mathbf{P}) = \max\{c_{11} + c_{21}, c_{32}\} = 4$.

Consider now the fully mixed assignment $\mathbf{F}$ with $f_{11} = f_{21} = \frac{6}{7}$, $f_{31} = \frac{1}{3}$, $f_{12} = f_{22} = \frac{1}{7}$ and $f_{32} = \frac{2}{3}$. We verify that $\mathbf{F}$ is a (fully mixed) Nash equilibrium:

- $\lambda_1(\mathbf{F}|1) = c_{11} + f_{21}c_{21} + f_{31}c_{31} = \frac{26}{7} = c_{12} + f_{22}c_{22} + f_{32}c_{32} = \lambda_1(\mathbf{F}|2)$.
- $\lambda_2(\mathbf{F}|1) = c_{21} + f_{11}c_{11} + f_{31}c_{31} = \frac{26}{7} = c_{22} + f_{12}c_{12} + f_{32}c_{32} = \lambda_2(\mathbf{F}|2)$.
- $\lambda_3(\mathbf{F}|1) = c_{31} + f_{11}c_{11} + f_{21}c_{21} = \frac{24}{7} = c_{32} + f_{12}c_{12} + f_{22}c_{22} = \lambda_3(\mathbf{F}|2)$.

It follows that $\mathbf{F}$ is at Nash equilibrium. The social cost of $\mathbf{F}$ is

$$\begin{aligned} SC(\mathbf{F}) &= f_{11}f_{21}f_{31}(c_{11} + c_{21} + c_{31}) + f_{11}f_{21}f_{32}\max\{c_{11} + c_{21}, c_{32}\} \\ &\quad + f_{11}f_{22}f_{31}\max\{c_{11} + c_{31}, c_{22}\} + f_{11}f_{22}f_{32}\max\{c_{11}, c_{22} + c_{32}\} \\ &\quad + f_{12}f_{21}f_{31}\max\{c_{12}, c_{21} + c_{31}\} + f_{12}f_{21}f_{32}\max\{c_{12} + c_{32}, c_{21}\} \\ &\quad + f_{12}f_{22}f_{31}\max\{c_{12} + c_{22}, c_{31}\} + f_{12}f_{22}f_{32}(c_{12} + c_{22} + c_{32}) \\ &= \frac{1}{7 \cdot 7 \cdot 3} \cdot (144 + 288 + 12 + 54 + 12 + 54 + 3 + 12) \\ &= \frac{193}{49}. \end{aligned}$$

Thus, $SC(\mathbf{F}) = \frac{193}{49} < 4 = SC(\mathbf{P})$, as needed. □

Theorem 7.5 shows that the Fully Mixed Nash Equilibrium Conjecture is not valid for the unrelated routing model, even for instances with $n = 3$ and $m = 2$. On the other hand, Theorem 7.4 shows that it holds for instances with two users and two links. It is an open problem whether it holds for instances with two users in general.

8

Conclusion and Open Questions

Selfish routing and other settings that involve selfish behavior form an important and ongoing field of research. This is not only shown by the multitude of recent publications. Circumstances like the growth of economic freedom in divers spheres (e.g. telecommunication and energy market) or the immense evolvement of the internet make it a non-surprising fact.

To get a deep insight into the rather complicated real world applications of selfish behavior, it is necessary to establish a comprehensive theoretical background. The purpose of this work is to expand the collection of results and to get a better understanding of selfish behavior in some concrete and well defined settings.

By introducing the KP model [47], Koutsoupias and Papadimitriou initiated a recent surge of research in the intersection of economics and computer science. We have presented several results on the KP model. Most notably, we gave an efficient Nashification algorithm which yields a PTAS for the problem to compute the best Nash equilibrium. Even though our algorithm does not increase the social cost, it involves reassignments of users that do not necessarily improve the users' individual costs. An open question is whether one can carry out Nashification in polynomial time with improving steps or greedy steps only. Furthermore, we have introduced the Nash Equilibrium Verification Problem and analyzed its algebraic complexity. Thereby we revealed an interesting relationship to a geometric problem which in turn helped us to give an optimal algorithm for the problem. The Nash Equilibrium Verification Problem can be considered for other settings as well. We leave this issue as a topic of future work.

For the KP model with identical links we have constructed an instance which takes a number of $2^{\frac{3}{2}\sqrt{n}-6}$ greedy steps to reach a Nash equilibrium in the worst case. On the other hand, the number of greedy steps is bounded by $2^n - 1$. It remains an open problem to close the gap between these bounds.

We followed up divers extensions of the KP model, like links with $M/M/1$-queue characteristics or links with weight constraints. Furthermore, we made a step toward the Wardrop model by adapting its definition of social cost and allowing very general latency functions, whereas we retained the indivisibility of flow and the restricted topology of the KP model. The challenge here is to surmount also the limitation to parallel links. Other relevant topologies can be the first step. E.g., one can show that the LPT algorithm computes a Nash equilibrium for unweighted users in series-parallel networks with non-decreasing link latency functions. What is the performance of the resulting routing? For weighted users it is not even clear whether a pure Nash equilibrium always exists for series-parallel networks.

Concerning our results on the coordination ratio for pure Nash equilibria of identical users on parallel links, the generalization to weighted users is still open. Another concrete question is whether the analysis of the algorithm in Figure 6.1 can improved as to obtain a better bound on its running time or if the given bound of $\mathcal{O}(m \log m \log n)$ is asymptotically tight.

For unweighted users and the definition of social cost as average cost per unit of flow, we have proved a coordination ratio of $\frac{4}{3}$ for related links. For the reciprocal case of weighted users on related links, the pure coordination ratio is $\frac{9}{8}$ ([51]). What is the coordination ratio for weighted users on related links?

The Fully Mixed Nash Equilibrium Conjecture was originally stated for the KP model [35]. Nevertheless, there is hope that it holds for other settings as well. It is known that it remains valid for the congestion routing game of unweighted users on parallel links with non-decreasing convex latency functions and the social cost being defined as sum of user costs. Hence, if the fully mixed Nash equilibrium exists for an instance of this model, then its social cost bounds the social cost of any Nash equilibrium. If an instance possesses some very fast and some very slow links at the same time, then the fully mixed Nash equilibrium, which requires that every link is used by each user with positive probability, may not exist. We have shown that if (some of) the slow links are removed as to get an instance that possesses a fully mixed Nash equilibrium, then the social cost of the fully mixed Nash equilibrium still is an upper bound on the social cost for any Nash equilibrium in the original instance. A further issue here is to generalize these results to the case of weighted users.

We have addressed the Fully Mixed Nash Equilibrium Conjecture also for the case of unrelated links (and everything else being defined as in the KP model). We could assert it for instances with two users and two links. On the other hand we gave an example to show that it does not hold for three users and two links. To fill the remaining gap is the next issue for this model. That is, we state the following open question: Is the Fully Mixed Nash Equilibrium Conjecture true for two users on unrelated links in general? Although there exists a pure Nash equilibrium for any instance of the model with unrelated links, it is an open problem to efficiently compute one (except for the two links case).

For the original KP model (with related links) the Fully Mixed Nash Equilibrium Conjecture was proved for a variety of restricted cases. Its general validity remains an open problem as well.

Publications

Many of the results presented in this thesis have been presented on international conferences or were submitted to journals. Here is a list of publications with the author's participation that have appeared by the printing time of the thesis.

[34] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and M. Rode. Nash equilibria in discrete routing games with convex latency functions. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3142 of *LNCS*, pages 645–657, 2004.

[51] T. Lücking, M. Mavronicolas, B. Monien, and M. Rode. A new model for selfish routing. In *Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2996 of *LNCS*, pages 547–558, 2004.

[27] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 2719 of *LNCS*, pages 514–526, 2003.

[28] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Selfish routing in non-cooperative networks: A survey. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 2747 of *LNCS*, pages 21–45, 2003.

[52] T. Lücking, M. Mavronicolas, B. Monien, M. Rode, P. Spirakis, and I. Vrto. Which is the worst-case nash equilibrium? In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 2747 of *LNCS*, pages 551–561, 2003.

[53] T. Lücking, B. Monien, and M. Rode. On the problem of scheduling flows on distributed networks. In *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 2402 of *LNCS*, pages 495–505, 2002.

Bibliography

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[2] E. Altman, T. Başar, T. Jiménez, and Nahum Shimkin. Routing into two parallel links: Game-theoretic distributed algorithms. *Journal of Parallel and Distributed Computing*, 61(9):1367–1381, 2001.

[3] E. Altman and L. Wynter. Equilibrium, games, and pricing in transportation and telecommunications networks. *Special Issue of Networks and Spatial Economics on "Crossovers between Transportation Planning and Telecommunications"*, 4(1):7–21, 2004.

[4] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[5] A. Archer and E. Tardos. Frugal path mechanisms. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 991–999, 2002.

[6] M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*. Yale University Press, 1956.

[7] M. J. Beckmann. On the theory of traffic flow in networks. *Traffic Quart*, 21:109–116, 1967.

[8] R. Beier, A. Czumaj, P. Krysta, and B. Vöcking. Computing equilibria for congestion games with (im)perfect information. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 746–755, 2004.

[9] M. Ben-Or. Lower bounds for algebraic computation trees. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–86, 1983.

[10] P. Berenbrink, L. A. Goldberg, P. Goldberg, and R. Martin. Utilitarian resource assignment. *The Computing Research Repository (CoRR)*, 2004. cs.GT/0410018.

[11] S. K. Bose. *An Introduction to Queueing Systems*. Kluwer Academic Publishers, 2001.

[12] D. Braess. über ein paradoxon der verkehrsplanung. *Unternehmensforschung*, 12:258–268, 1968.

[13] G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination mechanisms. In *Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 345–357, 2004.

[14] J. E. Cohen and P. Horowitz. Paradoxical behaviour of mechanical and electrical networks. *Nature*, 352:699–701, 1991.

[15] R. Cole, Y. Dodis, and T. Roughgarden. How much can taxes help selfish routing? In *Proceedings of the 4th ACM Conference on Electronic Commerce (EC)*, pages 98–107, 2003.

[16] R. Cole, Y. Dodis, and T. Roughgarden. Pricing network edges for heterogeneous selfish users. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 521–530, 2003.

[17] J. R. Correa, A. S. Schulz, and N. E. Stier Moses. Computational complexity, fairness, and the price of anarchy of the maximum latency problem. Working Paper 4447-03, Massachusetts Institute of Technology (MIT), Sloan School of Management, 2004. Available at http://ideas.repec.org/p/mit/sloanp/5051.html.

[18] 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 (STOC)*, pages 287–296. ACM Press, 2002.

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

[20] S.C. Dafermos and F.T. Sparrow. The traffic assignment problem for a general network. *Journal of Research of the National Bureau of Standards - B. Mathematical Sciences*, 73B(2):91–118, 1969.

[21] X. Deng, C. Papadimitriou, and S. Safra. On the complexity of equilibria. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 67–71, 2002.

[22] J. Erickson. New lower bounds for halfspace emptiness. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 472–481, 1996.

[23] E. Even-Dar, A. Kesselmann, and Y. Mansour. Convergence time to Nash equilibria. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 502–513, 2003.

[24] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *Proceedings of the 22nd Annual Symposium on Principles of Distributed Computing (PODC)*, pages 347–351, 2003.

[25] A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–612, 2004.

[26] J. Feigenbaum, C. Papdimitriou, and S. Shenker. Sharing the cost of multicast transmissions. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 218–227, 2000.

[27] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 514–526, 2003.

[28] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Selfish routing in non-cooperative networks: A survey. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 21–45, 2003.

[29] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. 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 (ICALP)*, pages 510–519. ACM Press, 2002.

[30] D. Fotakis, S. C. Kontogiannis, and P. G. Spirakis. Selfish unsplittable flows. In *Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 593–605, 2004.

[31] D. K. Friesen. Tighter bounds for lpt scheduling on uniform processors. *SIAM Journal on Computing*, 16(3):554–560, 1987.

[32] D. K. Friesen and M. A. Langston. Bounds for multifit scheduling on uniform processors. *SIAM Journal on Computing*, 12(1):60–70, 1983.

[33] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. Computing Nash equilibria for scheduling on restricted parallel links. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 613–622, 2004.

[34] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and M. Rode. Nash equilibria in discrete routing games with convex latency functions. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 645–657, 2004.

[35] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. Spirakis. Extreme Nash equilibria. In *Proceedings of the 8th Italian Conference on Theoretical Computer Science (ICTCS)*, pages 1–20, 2003.

[36] M. R. Garey and D. S. Johnson. *Computers And Intractability – A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[37] 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 (PODC)*, pages 131–140, 2004.

[38] G. Gottlob, G. Greco, and F. Scarcello. Pure Nash equilibria: hard and easy games. In *Proceedings of the 9th conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pages 215–230, 2003.

[39] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

[40] A. Haurie and P. Marcotte. On the relationship between Nash-Cournot and Wardrop equilibria. *Networks*, 15:295–308, 1985.

[41] D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems: Theoretical and practical results. *Journal of the ACM*, 34(1):144–162, 1987.

[42] D. S. Hochbaum and D. B. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988.

[43] C. Hurkens and T. Vredeveld. Local search for multiprocessor scheduling: how many moves does it take to a local optimum? *Operations Research Letters*, 31:137–141, 2003.

[44] Y. A. Korilis, A. A. Lazar, and A. Orda. Architecting noncooperative networks. *IEEE Journal on Selected Areas in Communications*, 13(7):1241–1251, 1995.

[45] Y. A. Korilis, A. A. Lazar, and A. Orda. The role of the manager in a noncooperative network. In *Proceedings of the 15th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1285–1293, 1996.

[46] E. Koutsoupias, M. Mavronicolas, and P. Spirakis. Approximate equilibria and ball fusion. *Theory of Computing Systems*, 36(6):683–693, 2003.

[47] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 404–413, 1999.

[48] V. S. A. Kumar and M. V. Marathe. Improved results for Stackelberg scheduling strategies. In *Proceedings of the 29th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 776–787, 2002.

[49] L. Libman and A. Orda. The designer's perspective to atomic noncooperative networks. *IEEE/ACM Transactions on Networking*, 7(6):875–884, 1999.

[50] L. Libman and A. Orda. Atomic resource sharing in noncooperative networks. *Telecommunication Systems*, 17(4):385–409, 2001.

[51] T. Lücking, M. Mavronicolas, B. Monien, and M. Rode. A new model for selfish routing. In *Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 547–558, 2004.

[52] T. Lücking, M. Mavronicolas, B. Monien, M. Rode, P. Spirakis, and I. Vrto. Which is the worst-case Nash equilibrium? In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 551–561, 2003.

[53] T. Lücking, B. Monien, and M. Rode. On the problem of scheduling flows on distributed networks. In *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 495–505, 2002.

[54] M. Mavronicolas and P. Spirakis. The price of selfish routing. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 123–134, 2001.

[55] R. D. McKelvey and A. McLennan. *Computation of Equilibria in Finite Games*. Elsevier, 1996.

[56] I. Milchtaich. Congestion games with player-specific payoff functions. *Games and Economic Behavior*, 13:111–124, 1996.

[57] B. L. Miller. On minimizing nonseparable functions defined on the integers with an inventory application. *SIAM Journal on Applied Mathematics*, 21(1):166–185, 1971.

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

[59] K. Murota and A. Shioura. Relationship of m-/l-convex functions with discrete convex functions by miller and favati-tardella. *Discrete Applied Mathematics*, 115(1–3):151–176, 2001.

[60] R. B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

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

[62] N. Nisan. Algorithms for selfish agents. In *Proceedings of the 16th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 1–15, 1999.

[63] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC)*, pages 129–140, 1999.

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

[65] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[66] G. Owen. *Game Theory*. Academic Press, 1995.

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

[68] F. P. Preparata and M. I. Shamos. *Computational Geometry - An Introduction*. Springer-Verlag, 1985.

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

[70] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

[71] T. Roughgarden. Designing networks for selfish users is hard. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 472–481, 2001.

[72] T. Roughgarden. Stackelberg scheduling strategies. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 104–113, 2001.

[73] T. Roughgarden. *Selfish Routing*. PhD thesis, Cornell University, 2002.

[74] T. Roughgarden. The price of anarchy is independent of the network topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003.

[75] T. Roughgarden and E. Tardos. How bad is selfish routing? In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 93–102, 2000.

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

[77] A. S. Schulz and N. Stier Moses. On the performance of user equilibria in traffic networks. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 86–87, 2003.

[78] S. Singh, M. Kearns, and Y. Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the 16th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 541–548, 2000.

[79] S. Suri, C. D. Tóth, and Y. Zhou. Selfish load balancing and atomic congestion games. In *Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 188–195, 2004.

[80] T. Ui. Discrete concavity for potential games. Working paper, Yokohama National University, Faculty of Economics, 2004. Available at http://www2.igss.ynu.ac.jp/ oui/d-potential.pdf.

[81] M. Voorneveld, P. Borm, F. van Megen, and S. Tijs. Congestion games and potentials reconsidered. Discussion Paper 98, Tilburg University, Center for Economic Research, 1999. Available at http://ideas.repec.org/p/dgr/kubcen/199998.html.

[82] M. Voorneveld and H. Norde. A characterization of ordinal potential games. *Games and Economic Behavior*, 19:235–242, 1997.

[83] J. G. Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institute of Civil Engineers (ICE), Part II*, volume 1, pages 325–378, 1956.