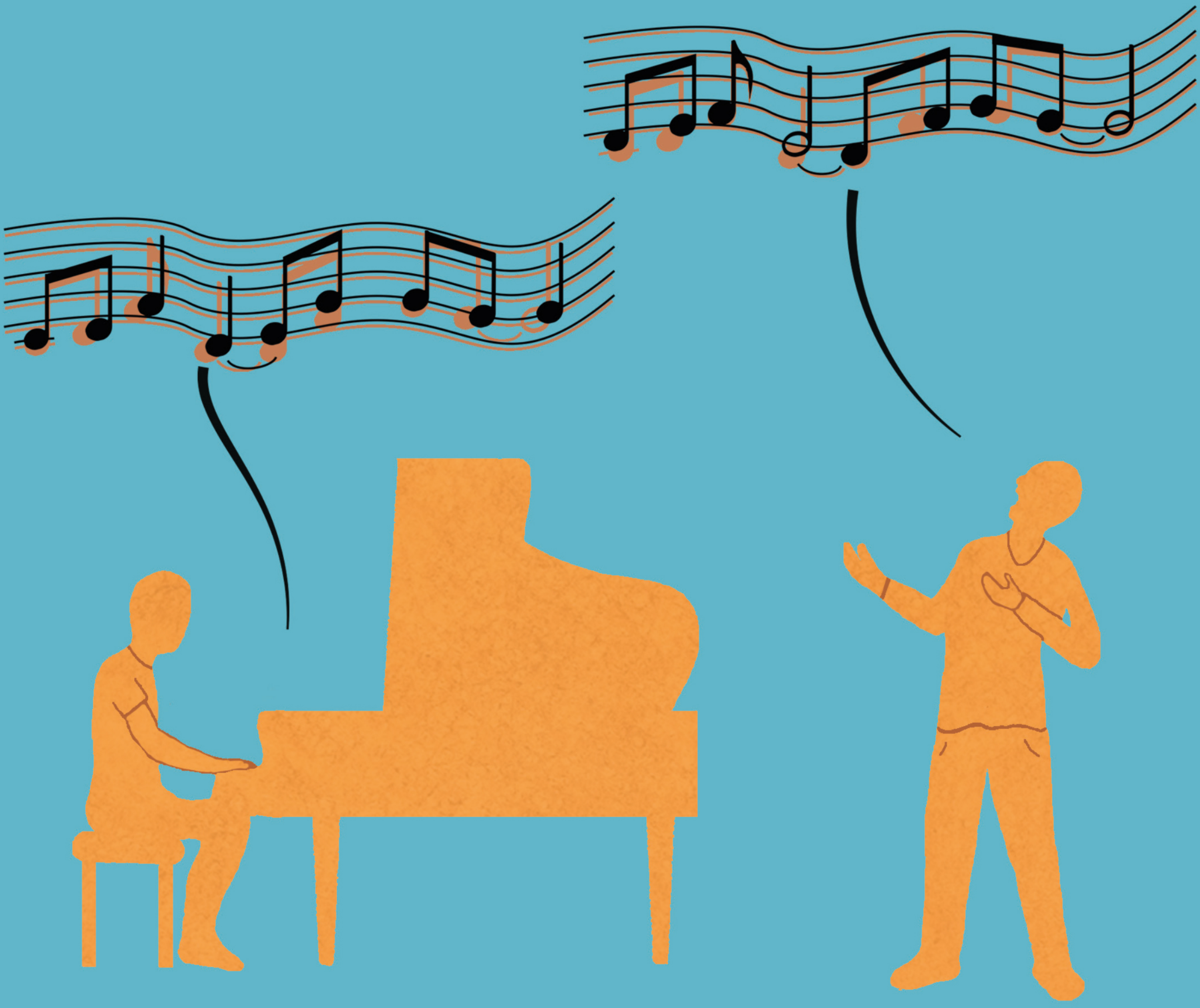
HOW CLOSE DOES IT GET?

From Near-Optimal Network Algorithms to Suboptimal Equilibrium Outcomes



RUBEN BROKKELKAMP

How Close Does It Get?

From Near-Optimal Network Algorithms
to Suboptimal Equilibrium Outcomes

ILLC Dissertation Series DS-2022-04

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

For further information about ILLC-publications, please contact

Institute for Logic, Language and Computation
Universiteit van Amsterdam
Science Park 107
1098 XG Amsterdam
phone: +31-20-525 6051
e-mail: `illc@uva.nl`
homepage: `https://www.illc.uva.nl/`

This research has been carried out in the Networks and Optimization group at the Centrum Wiskunde & Informatica in Amsterdam, The Netherlands, supported by the Netherlands Organisation for Scientific Research (NWO) through the Gravitation grant NETWORKS-024.002.003.

CWI
Centrum Wiskunde & Informatica

NWO

Cover design by Remco Wetzels - remcowetzels.nl

Printed and bound by Ipskamp printing

ISBN: 978–94–6421–825–1

Copyright © 2022 by Ruben Brokkelkamp

How Close Does It Get?

From Near-Optimal Network Algorithms to Suboptimal Equilibrium Outcomes

Academisch Proefschrift

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. G.T.M. ten Dam
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op woensdag 21 september 2022, te 11.00 uur

door

Klaas Ruben Brokkelkamp

geboren te Amsterdam

Promotiecommisie

Promotor:	prof. dr. G. Schäfer	Universiteit van Amsterdam
Copromotor:	prof. dr. U. Endriss	Universiteit van Amsterdam
Overige leden:	dr. E. Markakis	Athens University of Economics and Business
	prof. dr. M. Uetz	University of Twente
	dr. ir. R.A. Sitters	Vrije Universiteit Amsterdam
	prof. dr. K.R. Apt	Universiteit van Amsterdam
	dr. R. de Haan	Universiteit van Amsterdam
	prof. dr. D. Grossi	Universiteit van Amsterdam
	prof. dr. J.A. Ellis-Monaghan	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Contents

Acknowledgments

The first person I want to express my gratitude to is my promotor, Guido Schäfer. Guido, thank you so much for supervising, guiding, educating, motivating, entertaining, supporting, and helping me over the last four years. You were always a bottomless pit of new ideas when we got stuck and your enthusiasm and positive attitude have always been a great motivation. I think I can count on one hand the number of meetings in which we did not laugh. I have learned a great deal from you over the years. Many many thanks for everything!

I would like to thank all members of my doctorate committee, Vangelis Markakis, Marc Uetz, René Sitters, Krzysztof Apt, Ronald de Haan, Davide Grossi, and Jo Ellis-Monaghan, for taking the time to read and approve my thesis. A special mention to Krzysztof who, ten years ago, supervised the second-year project in the bachelor mathematics at the UvA of Mees (one of my paranymphs) and me, resulting in our first paper, and here we are together again.

Thank you, Lex Schrijver, for being my copromotor when the project started, and thank you, Ulle Endriss, for being so kind to take on this role as it comes to an end.

For all the research meetings, blackboard and whiteboard sessions, and discussions online and offline I thank my coauthors: Guido, Yllka (also for inviting me to collaborate on the research that is presented in Chapter 4), Sven, Andries, Arkaprava, Arijit, Francesco, and Sjir.

The Networks & Optimization group of CWI has been a great place to carry out research. Thanks to all my office mates throughout the years: Pieter, Dylan, Yorgos, Dario, Jan-Hendrik, Willem, Sophie, and Arthur for the silence when it was time to concentrate and the chats and laughs when it was time to relax. And to the other group members: Monique, Daniel, Nikhil, Neil, Sophie, Lucas, Luis, Andries, Samarth, Sander, Sander, Sven, Danish, Yllka, Simon, Irving Christian, Marek, Makrand, Akshay, Corby, Arthur, Lex, and Krzysztof. Next to being a group of extremely skilled researchers, I found N&O always very *gezellig*: someone knocking on the door at 11:59 so that we can have lunch at 12:00 all together, walking to the coffee machine to keep on chatting after lunch, table tennis and foosball sessions in the afternoons pre-covid (unfortunately, I am a bit ashamed to admit that I have not kept the tradition alive post-lockdowns, but hopefully this is a trigger for the new members to start it up again), and the coffee chats during the lockdowns with the fun little presentations have all contributed to a great atmosphere.

It has been a lot of fun being part of the CWI PhD activity committee with many members throughout the years: Arjan, Vlad, Hema, Isabella, Mathé, Esteban, Mark, Jan, Carl, Max, Sanne, Nikhil, Shane, Ludo, Sebastian, and (unofficial member) Dick. We have entertained and connected many CWI colleagues with all our gaming nights, table tennis, chess and foosball tournaments, barbecues, scavenger hunt, pub quizzes and many more events, but for me, the most fun were the online and offline meetings and get-togethers we have had with the committee, thanks for that!

I want to thank all the support staff of CWI and in particular Susanne, not only for all your secretarial duties but also for winning the N&O *jeu de boules* (pétanque for the non-dutchies) tournament together with Sander and me.

Also, a big thanks to all the people I have met and talked to at CWI during the (activity committee) events, Praethuys, intense foosball matches, or just in the hallways.

My PhD project was part of the Networks project through which I have attended many Networks training weeks and Networks (PhD) days. Thanks to all the Networks members who have made these events so enjoyable.

Buiten werk ben ik ook veel mensen dankbaar die door de jaren heen voor de nodige afleiding hebben gezorgd.

Jan, Mees en Raymond, de programmeerwedstrijden waar we als team *unsigned long long* aan mee hebben gedaan waren stuk voor stuk memorabel en ik keek er iedere keer maanden van te voren al naar uit. Dank voor alle uurtjes samen stuiteren achter onze laptops.

Lara, Luuk, Mees, Tristan en Zerline, dank voor alle avonden gezelligheid met het meest middelmatige pubquizteam van Amsterdam: *team Onderhond*.

Didier, dank voor alle keren dat je me naar buiten heb gesleept om een rondje te fietsen of hard te lopen en voor het organiseren van de wiskunde kerstdiners.

Amir, dank voor alle rondjes die we gewandeld hebben en de goede gesprekken.

Iedereen in de *Exfam*, de *Misschien*, de *OG's* en de *Kiss all kleppers* groepen, en iedereen die hier niet onder valt maar wel in deze acknowledgments thuishoort, enorm bedankt voor alle feestjes, spelletjes, rondjes fietsen, verjaardagen, festivals, kerstdiners, muziek, wintersporten, uitjes, borrels en andere evenementen die het leven zo leuk maken.

Mees, ondertussen is het alweer twaalf jaar geleden dat we samen aan de dubbele bachelor wiskunde & informatica aan de UvA begonnen. Naast dat je een ontzettend slim persoon bent waar ik al sinds de eerste dag dat we elkaar ontmoetten tegenop kijk, ben je bovenal een superleuk mens. Ik ben heel erg blij dat we na al die jaren nog goede vrienden zijn en dat je mijn paranimf wilde zijn.

Cara Isabella, sono incredibilmente felice di averti incontrata al CWI e che il mio scudo “non esco con i colleghi” non abbia potuto bloccarti. Sono molto grato di poter sempre contare su di te per il supporto e l’aiuto, anche quando stai attraversando un momento difficile. Mi diverto molto con te e imparo molto da te. Non vedo l’ora di tanti altri inverni in cui cercherò di tenere il passo con te sulle piste, di fare molti più sudoku insieme, di perfezionare le nostre paste, di ballare tante altre volte insieme fino al sorgere del sole e di scoprire la prossima fase della vita insieme. Ti amo.

Lieve Papa, Mama, Rona, Abel en Joram, ontzettend bedankt dat jullie er altijd voor me zijn, voor alle goede zorgen, en dat jullie het hebben van een fijne familie zo vanzelfsprekend en makkelijk maken. Ik hou van jullie.

Chapter 1

Introduction

The five chapters in this thesis cover a wide range of topics, and although the meaning of *close* and *it* varies in the different parts, the following question connects all of them:

How Close Does It Get?

Algorithms. Using computers, we are able to optimize all kinds of things in our lives. For example, a navigation app computes for us the fastest route to get to our destination, and a fitness watch tells us how long we need to rest before we should do our next set of intense exercises. Moreover, behind the scenes, even more things get optimized: the delivery routes of trucks, the timetable of the trains, the number of transistors on a computer chip, the maintenance work of the railways such that response teams are also quickly present when an incident occurs, and many more things.

A computer usually follows more or less the same procedure for solving most optimization problems: it takes some input, does some calculations, and produces output. For example, to compute the shortest route to a destination, the computer takes as input the road network, the starting location, and the destination. Then, after some computations, it outputs a route. Likewise, to create a work schedule, the computer takes as input the availability of employees and an overview of how much personnel is needed and when. Then, it does some computations and outputs a schedule. Such a procedure that consists of taking input, doing calculations, and producing output is called an *algorithm*.

If we use an algorithm to solve an optimization problem, such as finding the shortest route, we would like to have the optimal output as fast as possible. However, in many cases, as fast as possible can still take a very long time because there can be too many possibilities to consider, and we do not know of any smart trick to speed up the computation. Therefore, instead of always requiring an optimal solution, it is often good enough to have an algorithm that returns solutions that are near-optimal or an algorithm that returns solutions that are optimal most of the time. Algorithms that run fast and for which we can prove that they always return solutions that are near-optimal are called *approximation algorithms*. Algorithms that return solutions that are optimal most of the time are called *Monte Carlo algorithms*.

Network Pricing. The international tax system can be seen as a network of countries that are connected by tax treaties. Multinationals send profits through this network and use the paths on which they have to pay the least taxes. Regularly, there are news headlines in the media in which the Netherlands is called a *tax heaven* (Dekker and Kreling, 2022; NOS, 2014, 2019; Waard, 2011). However, the Netherlands is not a tax heaven in the sense that it is a (temporary) final destination for multinationals to store profits, but it is used as an intermediate host in tax avoidance schemes. The Netherlands is a so-called *conduit country* (Lejour, Riet, and Möhlmann, 2019; Riet and Lejour, 2013). An indicator of countries being conduit countries is the *betweenness centrality* (Freeman, 1977) of a country in this international tax network (Polak, 2014; Riet and Lejour, 2018). On a high level, the betweenness centrality of a country is proportional to how often a country occurs as an intermediate stop in the cheapest paths of all money flows in the network. A natural optimization question from the perspective of a country is what it can do to increase its betweenness centrality. By changing its tax rates, a country can become more attractive as an intermediate country, possibly attracting jobs in the financial sector. However, increasing betweenness centrality is not the only conceivable objective. Countries earn tax revenue on the money flows, and although lowering tax rates most likely increases the money flow, it may not outweigh the loss in revenue. Another natural question is, therefore, what a country can do to maximize its revenue.

We model this as a fundamental network optimization problem: countries correspond to nodes, bilateral tax treaties to edges with tax rates, and money flows to commodities that consist of a source node, a destination node, and a demand. We select a special node u that is able to change the tax rates on its outgoing edges to maximize the flow going through it or to maximize its revenue. A country might have limited resources and be unable to put effort into changing all the outgoing tax rates, so we also introduce a parameter τ that indicates on how many edges it can change the tax rate. We show that it is highly unlikely that there exists an algorithm that solves this general problem efficiently. We even show that it is highly unlikely that there exists an algorithm that can efficiently approximate the optimal solution within certain factors. However, for special cases, we give polynomial-time algorithms or approximation algorithms.

How close does it get? The approximation guarantees that our algorithms achieve are essentially the best possible given the inapproximability results.

Shortest Paths in Uncertain Networks. In a deterministic network, there is a clear candidate for what a shortest path is. For a source and destination node, one can consider all possible paths from the source to the destination node and take the one of shortest length. However, this definition does not extend to networks in which edges are sometimes unavailable. For example, the connection between two access points can fail in a telecommunications network, and in a road network, accidents can temporarily block a street. One way to model this is by assigning to each edge a probability with which the edge is present. It is still possible to determine a shortest path in this network when all edges are available. However, this path is not very useful if it exists with only a small probability. Therefore, we study the notion of the *Most Probable Shortest Path* (MPSP): the path that has the highest probability of being the shortest path.

We show that it is highly unlikely that there exists an algorithm that efficiently computes the probability that a given path is the shortest path. To still be able to find the MPSP, we develop an efficient sampling-based algorithm that tries to compute it.

How close does it get? We prove that the algorithm returns the correct path with high probability. On top of that, we conduct extensive experiments to show that our method works well in practice.

Machine Scheduling Game. In the problem of maximizing betweenness or revenue, we took the perspective of a single country. Of course, every country in the network can use the same procedure to optimize its revenue, but when one country changes its tax rate, this also affects the other countries. The field of *algorithmic game theory* is concerned with analyzing settings in which individuals or entities ('players') make selfish decisions that affect the other players in the 'game'. We study selfish behavior in *related machine scheduling*. In the optimization version of related machine scheduling, we want to schedule a set of jobs, each having a certain processing time, on machines, each with a speed. The time it takes to process a job on a machine is its processing time divided by the speed of the machine, and machines can only process one job at a time. This situation arises naturally on, for example, a supercomputer with many processors of different speeds.

We are interested in minimizing the sum of completion times. If there is just a single machine, the sum of completion times is minimized when the jobs are processed in order from shortest to longest processing time. Therefore, a simple greedy algorithm for the setting with multiple machines iterates through the jobs from shortest to longest and schedules them on a machine that minimizes their completion time.

An example of a run of the simple greedy algorithm on an instance with four jobs with processing times $1, 1, 2$, and 4 and three machines with speeds $1, 1$, and 2 (ties broken in favor of machine 3) is visualized in Figure 1.1(a). This schedule

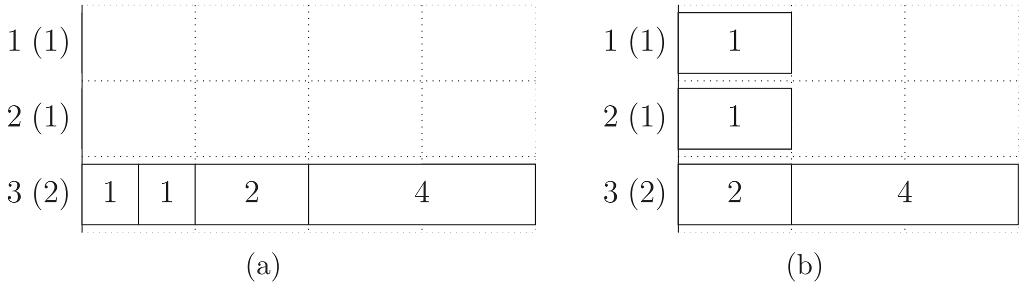


Figure 1.1: Related machine scheduling example

has a sum of completion times of $\frac{1}{2} + 1 + 2 + 4 = 7\frac{1}{2}$, while the optimal schedule (displayed in Figure 1.1(b)) has a sum of completion times of 6 which is a factor $\frac{5}{4}$ less than the simple greedy schedule.

In the game-theoretic version of related machine scheduling, every job is controlled by a player, and they are interested in minimizing the completion time of their own job. On each machine, the jobs are processed in order from shortest to longest, but players are free to choose on which machine their job will be scheduled. A schedule in which no player has an incentive to *unilaterally* deviate to another machine is also displayed in Figure 1.1(a) and is called a *Nash equilibrium* (if a job would select one of the other machines, their completion time would be the same). The cost of the Nash equilibrium is a factor $\frac{5}{4}$ worse than the optimum. Quantifying this *inefficiency* (also called the *price of anarchy*) caused by self-interested behavior is a fundamental problem in algorithmic game theory.

Interestingly, for the related machine scheduling game, the approximation guarantee of the simple greedy algorithm and the price of anarchy is the same. Hence, answering the question of how close the approximation algorithm gets to optimal is equivalent to answering the question of how large the price of anarchy is.

How close does it get? Resolving this question is difficult and requires new insights. We outline a technique that approaches the problem from the price of anarchy perspective, which is able to recover already known bounds and might be strong enough to get better bounds.

Corrupt Auctions. We are interested in auctions in which one or multiple identical items are sold. The bidders have private valuations for the items that are unknown to the other bidders and the auctioneer. In a first-price auction, the bidders submit bids, and the items are allocated to the highest bidders. The winning bidders pay their bid to receive the items. Bidders try to maximize their utility, which is their valuation for the items they receive minus what they have to pay.

The auctioneer is often not the owner of the items on sale. Auction houses or auction websites host auctions on behalf of clients who lack the expertise to do so themselves. The client would like to obtain a good price for the items, while the auctioneer is interested in maximizing their own gains from hosting the auction. This misalignment can lead to the auctioneer manipulating the auction for their own benefit.

Corruption in auctions, where an auctioneer engages in bid rigging with one (or several) of the bidders, occurs rather frequently in practice, especially in the public sector (e.g., in construction and procurement auctions[1]). For example, in 1999, the procurement auction for the construction of the new Berlin Brandenburg airport had to be rerun after investigations revealed that the initial winner was able to change the bid after they had illegally acquired information about the application of one of their main competitors (The Wall Street Journal, 1999). As another example, in 1993, the New York City School Construction Authority caused a scandal when an investigation revealed that they used a simple (but effective) bid-rigging scheme in a procurement auction setting (Olmstead, 1993):

> *"In what one investigator described as a nervy scheme, that worker would unseal envelopes at a public bid opening, saving for last the bid submitted by the contractor who had paid him off. At that point, knowing the previous bids, the authority worker would misstate the contractor's bid, ensuring that it was low enough to secure the contract but as close as possible to the next highest bid so that the contractor would get the largest possible price."*

We capture corruption in a basic model in which the auctioneer approaches the winners of the auction with the offer to lower their bid to the highest losing bid in return for a fraction of the gains as a bribe. Bidders possibly change their strategic behavior when they know corruption is happening, and we are interested in quantifying the effect of corruption on the social welfare.

We study the price of anarchy of the corrupt auction with respect to the social welfare of various equilibrium notions, ranging from pure Nash equilibria to coarse correlated equilibria, as a function of the size of the bribe.

How close does it get? When players are allowed to overbid, we obtain a tight bound on the coarse correlated price of anarchy. Under a no overbidding assumption, we prove (almost) tight bounds for various equilibrium notions and variants of the corrupt auction.

Altruism in Mechanism Design. On a high level, mechanism design is concerned with guiding decision-making in a group context. Without incentives, selfishness can motivate an individual to over- or understate their actual preferences for the possible alternatives. By providing the right incentives, individuals are

[1] In a procurement auction the roles of buyer and seller are reversed. Instead of a single seller and multiple buyers there is a single buyer selecting the best option among multiple sellers.

nudged towards revealing their true preferences, which in turn can be used to make the best decision for the group as a whole. The incentives are often in the form of payments by or to the participants. An example of a mechanism in which players are incentivized to reveal their true preferences is the single-item second-price auction, in which the highest bidding bidder wins the item and pays an amount equal to the second-highest bid. In a first-price auction in which the winner pays their bid, there is always the stimulus to bid a bit lower (but hopefully above the second-highest bid) to save some money, while in a second-price auction, this stimulus is taken away.

Mechanism design relies heavily on the *self-interest hypothesis*, which is the assumption that individuals make decisions driven by purely selfish motives. We made this assumption in the related machine scheduling game and also for the corrupt auction. In fact, most models in mathematical economics rely on this assumption. Assuming that participants are selfish often simplifies analysis and, in many settings where this assumption is valid, it enables strong predictions for the outcomes of economic situations. However, various empirical studies show that individuals do not always act self-interestedly (Andreoni and Miller, 2002; Charness and Rabin, 2002; Kahneman, 2011). The field of behavioral economics is dedicated to studying when decisions of individuals or institutions vary from what would be the ‘rational’ choice.

If being fully selfish is on one end of the spectrum, being fully altruistic is on the other end of the spectrum. Many people are somewhere in between: they care about themselves but are also partially altruistic towards others. Taking into account that some individuals are partially altruistic might change the incentives that are needed to nudge the participants to reveal their true preferences. We consider incentives that come in the form of payments, and so we are interested in the question: what are the effects of (partial) altruism on the payments in mechanism design? We show that altruism leads to a wider range of possible payment functions that incentivize (partially) altruistic individuals to reveal their true preferences.

In an auction, the seller is usually interested in the revenue, so asking for substantial payments from the participants makes sense. As a result, money flows from the participants to some third party. However, often decisions have to be made in a group that would like the money to stay among the participants. An example is a group of siblings who have inherited a house from their parents. Only a single sibling can live in the house, and the parents want the sibling who wants it the most to live in the house. If one sibling has to buy the others out, as little money as possible should be spent on a third party, but the siblings should still be incentivized to reveal truthfully how much they want the house.

How close does it get? We show that mechanisms that are not designed with altruism in mind can be converted into ones that do and, in the process, make sure that there is less money going to a third party.

About This Thesis

The work presented in this thesis is the result of research carried out in the Networks and Optimization group at Centrum Wiskunde & Informatica (CWI) in Amsterdam, the Netherlands. The PhD position was funded by the Netherlands Organization for Scientific Research (NWO) through the Gravitation-grant NETWORKS-024.002.003.

Outline and Publications

Unless mentioned otherwise, the authors contributed equally to the papers mentioned in this subsection.

Chapter 2. We start with a chapter introducing the notation and most of the concepts needed to understand the results in the remaining chapters. It is divided in five subsections: basic notation and definitions, networks, algorithms, algorithmic game theory and mechanism design.

Chapter 3. The research in this chapter originated in the MSc thesis of Sven Polak (2014) and was extended and improved resulting in the following publication:

> Ruben Brokkelkamp, Sven C. Polak, Guido Schäfer, and Yllka Velaj (2019). "Approximate Pricing in Networks: How to Boost the Betweenness and Revenue of a Node". In: *30th International Symposium on Algorithms and Computation, ISAAC 2019, December 8-11, 2019, Shanghai University of Finance and Economics, Shanghai, China*. Edited by Pinyan Lu and Guochuan Zhang. Volume 149. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 13:1–13:15. DOI: `10.4230/LIPIcs.ISAAC.2019.13`

In this chapter we introduce and study two new pricing problems in networks. Suppose we are given a directed graph $G = (V, E)$ with non-negative edge costs $(c_e)_{e \in E}$, k commodities $(s_i, t_i, w_i)_{i \in [k]}$ and a designated node $u \in V$. Each commodity $i \in [k]$ is represented by a source-target pair $(s_i, t_i) \in V \times V$ and a demand $w_i > 0$, specifying that w_i units of flow are sent from s_i to t_i along shortest s_i, t_i-paths (with respect to $(c_e)_{e \in E}$). The demand of each commodity is split evenly over all shortest paths. Assume we can change the edge costs of some of the outgoing edges of u, while the costs of all other edges remain fixed; we also say that we *price* (or *tax*) the edges of u.

We study the problem of pricing the edges of u with respect to the following two natural objectives: (i) *max-flow*: maximize the total flow passing through u, and (ii) *max-revenue*: maximize the total revenue (flow times tax) through u. Both variants have various applications in practice. For example, the max flow objective is equivalent to maximizing the *betweenness centrality* of u, which is one of the most popular measures for the influence of a node in a (social) network.

We prove that, except for some special cases, both problems are NP-hard and inapproximable in general and therefore resort to approximation algorithms. We derive approximation algorithms for both variants and show that the derived approximation guarantees are best possible.

Chapter 4. The work presented in this chapter is based on the following paper:

> Arkaprava Saha, Ruben Brokkelkamp, Yllka Velaj, Arijit Khan, and Francesco Bonchi (2021). “Shortest Paths and Centrality in Uncertain Networks”. In: *Proceedings of the VLDB Endowment* 14.7, pages 1188–1201. DOI: `10.14778/3450980.3450988`

where the first two authors carried out the majority of the research.

In this chapter, we look at *uncertain graphs*, i.e., graphs in which every edge has some probability of existence. In these graphs, we are interested in finding the path between two nodes with the highest probability of being the shortest path. This path is called the *most probable shortest path*. We show that it is #**P**-hard to compute the probability that a given path is the shortest path. To still have some reasonable approximation, we develop a sampling-based Monte Carlo type algorithm that is able to quickly find a most probable shortest path. Based on this notion of a shortest path we also define a new betweenness centrality measure and give a sampling-based algorithm for computing it. We do extensive experiments on sensor networks, road networks and brain networks to show the effectiveness and usefulness of our solution.

Chapter 5. This chapter is based on unpublished work. It is the result of discussions and cooperation with Guido Schäfer. It contains some (minor) new results and serves as a bridge between Chapters 3 and 4, which have a more optimization perspective, and Chapters 6 and 7, in which game theory plays a role. We look at the problem of related machine scheduling and, in particular, into analyzing the pure price of anarchy. The best-known upper bounds for the price of anarchy are proved using *smoothness*, a powerful technique to get bounds on the price of anarchy which extend up to coarse correlated equilibria. However, for identical machine scheduling, it is known that there is a gap between the pure price of anarchy and the mixed price of anarchy. Therefore, smoothness cannot be used to get a tight bound on the pure price of anarchy.

We do not know yet if this gap also exists for related machine scheduling, but if it does, we need specialized techniques tailored for pure Nash equilibria. We outline how a variation of the primal-dual method by Bilò can be used to approach this problem. It easily recovers the (already known) bound of 2 using only a seemingly small subset of its possibilities. Unfortunately, we did not succeed in proving a better bound, but we conjecture that the technique as such is powerful enough to solve this problem. We hope this chapter serves as inspiration and as a basis for future work. Further, we provide better lower bounds on the pure price of anarchy for a fixed number of machines and jobs than previously known in the literature.

Chapter 6. This chapter is based on the following paper:

> Andries van Beek, Ruben Brokkelkamp, and Guido Schäfer (2022). "Corruption in Auctions: Social Welfare Loss in Hybrid Multi-Unit Auctions". In: *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. Edited by Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems, pages 1283–1291. DOI: 10.5555/3535850.3535993

In this chapter, we initiate the study of the social welfare loss caused by corrupt auctioneers, both in single-item and multi-unit auctions. In our model, the auctioneer may collude with the winning bidders by letting them lower their bids in exchange for a (possibly bidder-dependent) fraction γ of the surplus. We consider different corruption schemes. In the most basic one, all winning bidders lower their bid to the highest losing bid. We show that this setting is equivalent to a *γ-hybrid auction* in which the payments are a convex combination of first-price and second-price payments. More generally, we consider corruption schemes that can be related to *γ-approximate first-price auctions (γ-*FPA*)*, where the payments recover at least a γ-fraction of the first-price payments. Our goal is to obtain a precise understanding of the robust price of anarchy of such auctions. If no restrictions are imposed on the bids, we prove a bound on the robust price of anarchy of γ-FPA, which is tight (over the entire range of γ) for the single-item and the multi-unit auction setting. On the other hand, if the bids satisfy the no-overbidding assumption, a more fine-grained landscape of the price of anarchy emerges, depending on the auction setting and the equilibrium notion. Albeit being more challenging, we derive (almost) tight bounds for both auction settings and several equilibrium notions, basically leaving open some (small) gaps for the coarse-correlated price of anarchy only.

Chapter 7. The research in this chapter originated in the MSc thesis of Sjir Hoeijmakers (2014) and was extended and improved resulting in the following submission:

> Ruben Brokkelkamp, Sjir Hoeijmakers, and Guido Schäfer (2022). "Greater Flexbility in Mechanism Design Through Altruism". In: *International Symposium on Algorithmic Game Theory*. Springer

We study the problem of designing truthful mechanisms for players that are (partially) altruistic. Our approach is to extend the standard utility model by encoding other-regarding preferences of the players into the utility functions. By doing so, we leave the original domain where VCG mechanisms can be applied directly.

We derive a characterization of the class of truthful mechanisms under the new model, crucially exploiting the specific form of the other-regarding preferences. We also derive sufficient conditions for truthfulness which we then exploit to derive mechanisms for two specific models of altruism and with respect to two natural social welfare objectives. As it turns out, altruistic dispositions lead to the positive effect that the designer needs to extract smaller payments from the players to ensure truthfulness. Further, we investigate the effect of redistribution mechanisms that can redistribute the payments among the players. Also, here, it turns out that altruism has a positive effect in the sense that the payments needed to guarantee truthfulness can be further reduced.

Finally, we illustrate our theoretical results by applying them to well-studied mechanism design problems such as the public project problem and the multi-unit auction problem. Among other results, we show that the problem of funding a public project can be resolved by our mechanism even for moderate altruistic dispositions, while this is impossible in the standard utility setting.

Other. During the PhD the author has also contributed to the following papers

> Ruben Brokkelkamp, Raymond van Venetië, Mees J. de Vries, and Jan Westerdiep (2020). "PACE Solver Description: tdULL". in: *15th International Symposium on Parameterized and Exact Computation, IPEC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)*. Edited by Yixin Cao and Marcin Pilipczuk. Volume 180. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 29:1–29:4. DOI: `10.4230/LIPIcs.IPEC.2020.29`
>
> Alex Kuiper, Michel Mandjes, Jeroen de Mast, and Ruben Brokkelkamp (2021). "A flexible and optimal approach for appointment scheduling in healthcare". In: *Decision Sciences*. DOI: `https://doi.org/10.1111/deci.12517`

Chapter 2

Preliminaries

This chapter contains an introduction to most of the concepts needed to understand the content of this thesis. It is not a full overview of the respective areas, but it highlights notions used in later chapters. The section on algorithms is mainly based on the book by Cormen et al. (2009) with parts from Williamson and Shmoys (2011). We refer to those books for the interested reader that needs more details. For additional background on the algorithmic game theory and mechanism design sections, we refer to Nisan et al. (2007).

2.1 Basic Notation and Definitions

We use $\mathbb{N}$ to denote the set of positive integers and $\mathbb{R}$ to denote the set of real numbers. When restricting to the non-negative reals, we write $\mathbb{R}_{\geq 0}$. For an integer $n \in \mathbb{N}$ we define $[n]$ for the set $\{1, 2, \ldots, n\}$.

Let $X : \Omega \to \mathbb{R}$ be a real-valued random variable that maps from some sample space Ω to $\mathbb{R}$. We denote its expectation by $\mathbb{E}[X]$. We write $x \sim X$ if x is a sample drawn according to the probability distribution underlying X.

Definition 2.1.1. Let N be a finite set and let $z : 2^N \to \mathbb{R}$ be a function, mapping every subset $S \subseteq N$ to a real value $z(S)$. The function z is called

(i) *non-negative* if $z(S) \geq 0$ for all $S \subseteq N$,

(ii) *monotone* if $z(S) \leq z(T)$ for every $S \subseteq T \subseteq N$,

(iii) *submodular* if for all sets $S \subseteq T \subseteq N$ and every element $e \in N \setminus T$ it holds that $z(S \cup \{e\}) - z(S) \geq z(T \cup \{e\}) - z(T)$.

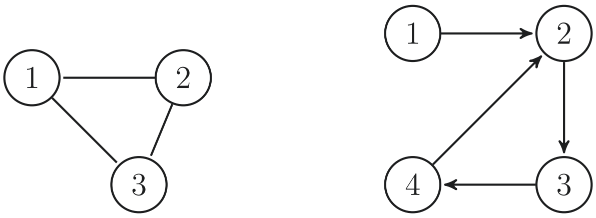


Figure 2.1: Two examples of graphs

2.2 Networks

A *network* or *graph* $G = (V, E)$ consists of a set of *nodes* V and a set of *edges* $E \subseteq V \times V$. An edge $e \in E$ can be *directed* or *undirected*. If e is an undirected edge connecting nodes u and v we write $e = \{u, v\}$, while if it is a directed edge going from u to v we write $e = (u, v)$ and sometimes uv. Examples of both a directed and an undirected graph can be found in Figure 2.1. In this thesis, we usually consider directed graphs unless mentioned otherwise.

We use the notation $\delta^+(u)$ to refer to the set of all outgoing edges of u, i.e., $\delta^+(u) = \{(u, v) \in E\}$, and define $d^+(u) = |\delta^+(u)|$ as the outdegree of u.

Given two nodes $s, t \in V$, we use two different ways interchangeably to denote a *path* P from s to t. The first option is that we represent P by an ordered sequence of edges $P = (e_1, e_2, \dots, e_n)$ such that $e_i = (u_i, u_{i+1}) \in E$ and $u_1 = s, u_{n+1} = t$. Observe that the path goes from edge e_i through node u_{i+1} to edge e_{i+1}. The second option is that we represent P by an ordered sequences of nodes $P = (u_1, u_2, \dots, u_{n+1})$ where $u_1 = s, u_{n+1} = t$ and $(u_i, u_{i+1}) \in E$ for all $i \in [n]$. If $u_i \neq u_j$ for all $i \neq j$, i.e., the path does not contain any cycles, we call the path P *simple*. When we refer to the *internal* nodes of a path P, we mean the nodes $u_2, u_3, \dots, u_n$, i.e., the nodes of the path without the endpoints, and we write $Int(P)$ to denote this set. The first node $s = u_1$ is called the *source node*, and the last node $t = u_{n+1}$ is called the *destination* or *terminal node*. For two paths $P = (e_1, \dots, (u, v))$ and $Q = ((v, w), \dots, e_n)$ such that the last node of P is the same as the first node in Q we write $P \cdot Q = (e_1, \dots, (u, v), (v, w), \dots, e_n)$ for their concatenation. Finally, we define the empty path, i.e., the path without edges, by P_ϕ.

We often assign numbers to the edges, which define, for example, weights or costs of the edges. We usually write $(c_e)_{e \in E}$ where $c_e \in \mathbb{R}$ is the number assigned to edge e. The *length* or *cost* of a path $P = (e_1, e_2, \dots, e_n)$ with respect to the edge costs $(c_e)_{e \in E}$ is the sum of the cost of the edges and is denoted by $c(P) = \sum_{e_i \in P} c_{e_i}$. We define $c(P_\phi) = 0$, and we also define a special empty path P_∞ with $c(P_\infty) = \infty$, which will simplify exposition.

For two nodes $s, t \in V$, we define $\mathcal{P}(G, s, t)$ to be the set of all paths from s to t. If there is no path from s to t then $\mathcal{P}(G, s, t) = \emptyset$. By $\delta(G, c, s, t)$, we denote the *distance* from s to t. If there is a path from s to t that contains a cycle of negative cost we define $\delta(G, c, s, t) = -\infty$, otherwise

$$\delta(G, c, s, t) = \begin{cases} \min_{P \in \mathcal{P}(G,s,t)} c(P) & \text{if } \mathcal{P}(G, s, t) \neq \emptyset \\ \infty & \text{if } \mathcal{P}(G, s, t) = \emptyset. \end{cases}$$

A *shortest path* from s to t is a path $P \in \mathcal{P}(G, s, t)$ with cost $\delta(G, c, s, t)$, i.e., $c(P) = \delta(G, c, s, t)$. Next, by $\mathsf{SP}(G, c, s, t)$ we denote the set of all shortest paths from s to t in G with respect to c. For both δ, $\mathcal{P}$ and SP we omit G and/or c as an argument, e.g., $\delta(s, t)$ and $\mathsf{SP}(G, s, t)$, if it is clear from the context which graph and cost function are used.

For a directed graph $G = (V, E)$ with costs $(c_e)_{e \in E}$, we denote by $\pi(G, c, s, t)$ the number of shortest paths from s to t, i.e., $\pi(G, c, s, t) = |\mathsf{SP}(G, c, s, t)|$. For a node $v \in V$ we use $\pi_v(G, c, s, t)$ for the number of shortest paths from s to t going through v. Also here, we drop the arguments G and c if they are clear from the context. The *betweenness centrality* (Freeman, 1977) of a node $v \in V$ is then defined as

$$b_G(v) = \frac{1}{|V|(|V| - 1)} \sum_{\substack{(s,t) \in V \times V \\ s \neq v \neq t, \pi(s,t) \neq 0}} \frac{\pi_v(s, t)}{\pi(s, t)}. \tag{2.1}$$

If there is a unique shortest path between all pairs of nodes in the graph, the betweenness centrality of a node v indicates which fraction of those shortest paths has v as an internal node. If there are multiple shortest paths from a node s to t, the internal nodes of those paths get a contribution to their betweenness proportional to the number of shortest paths of which they are an internal node.

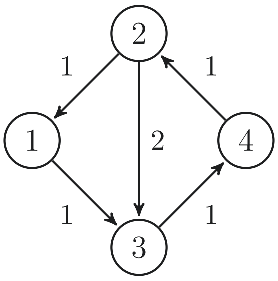


Figure 2.2: Example graph betweenness centrality

Example 2.2.1. Consider the graph in Figure 2.2. We have $V = [4]$ and $E = \{(1, 3), (2, 1), (2, 3), (3, 4), (4, 2)$. Each edge is labeled with its respective cost. The shortest paths are shown in Table 2.1: the cell in row r and column c contains the shortest paths from node r to node c. For example, to go from node 2 to 4 there are two paths of length 3 which is the minimum length. Hence, $\delta(2, 4) = 3$. One path is $2 \to 3 \to 4$ and the other path is $2 \to 1 \to 3 \to 4$. This implies that $\pi(2, 4) = 2$, $\pi_3(2, 4) = 2$ and $\pi_1(2, 4) = 1$.

	$1 \to 3 \to 4 \to 2$	$1 \to 3$	$1 \to 3 \to 4$
$2 \to 1$		$2 \to 3$ $2 \to 1 \to 3$	$2 \to 3 \to 4$ $2 \to 1 \to 3 \to 4$
$3 \to 4 \to 2 \to 1$	$3 \to 4 \to 2$		$3 \to 4$
$4 \to 2 \to 1$	$4 \to 2$	$4 \to 2 \to 3$ $4 \to 2 \to 1 \to 3$	

Table 2.1: Shortest paths in betweenness centrality example

By doing the calculations, we obtain $b_G(1) = \frac{1}{12} \cdot \frac{3}{2} = \frac{1}{8}, b_G(2) = b_G(3) = b_G(4) = \frac{1}{12} \cdot 3 = \frac{1}{4}$. Hence, nodes 2, 3, 4 have an equal betweenness centrality while the betweenness of node 1 is lower.

2.3 Algorithms

An *algorithm* is a well-defined computational procedure that takes *input*, performs a sequence of computational steps, and produces some *output*. As such, it can be used to solve well-specified *computational problems*. A computational problem consists of an input/output relationship. For example, for the sorting problem, in which we want to sort a list of numbers, this looks as follows:

Problem Sorting Problem

Input: List $L = [L[1], L[2], \ldots, L[n]]$ of n numbers.
Output: A permutation $[L[\pi_1], L[\pi_2], \ldots, L[\pi_n]]$ of the input numbers such that $L[\pi_1] \leq L[\pi_2] \leq \ldots \leq L[\pi_n]$, where π is a permutation of $[n]$.

An input is also called an *instance*. If an algorithm produces the correct output for every possible input that adheres to the input specifications, we say that the algorithm *solves* the problem.

Example 2.3.1. An algorithm to solve the problem of sorting a list of numbers can be found in Algorithm 1. It is called *insertion sort*, and it is an algorithm that is inspired by how we sort a set of cards in hand: from left to right, one takes out the next card and inserts it at the spot such that all cards up to that card are sorted. In Algorithm 1, this is done in an analogous way by swapping numbers with the preceding numbers in the list until it is in the correct spot.

Example 2.3.2. A more elaborate algorithm that is used to solve the problem of finding a shortest path between two nodes in a graph is *Dijkstra's Shortest Path*

Algorithm 1 Insertion Sort

```
Input: List $L = [L[1], L[2], \ldots, L[n]]$ of $n$ numbers.
Output: A sorted list $L$
for $i = 1 \to n$ do
    $j \leftarrow i$        /* Insert $L[i]$ into the sorted sequence $L[1..i-1]$ */
    while $j > 1$ and $L[j-1] > L[j]$ do
        swap $L[j]$ and $L[j-1]$
        $j \leftarrow j - 1$
    end while
end for
return $L$
```

Algorithm (Dijkstra, 1959). Its pseudocode can be found in Algorithm 2. This algorithm leverages the fact that a subpath of a shortest path is also a shortest path. It starts by assigning a path of length ∞ to every node other than s, which is assigned the empty path. Then, it repeatedly selects the unvisited node u with the shortest distance to s. From u, it scans all unvisited neighboring nodes and checks for each neighbor v if the path via u is shorter than the path currently stored in $\mathbf{P}[v]$. If so, it updates $\mathbf{P}[v]$. One can show that when a node u is selected in Line 11 of Algorithm 2 there does not exist a shorter path from s to v than $\mathbf{P}[v]$. The algorithm keeps selecting unvisited nodes u with the shortest distance to s until node t is reached, or a node u is selected that is not connected to s. The latter implies that t is also not connected to s.

Running Time

To compare two algorithms that solve the same problem, we can compare the resources that the algorithms require. Generally, we are most interested in the computational time that algorithms need, but sometimes memory or communication bandwidth requirements are important. One way of comparing the running times of algorithms is by trying them out on a large set of inputs and timing how long they take. If we run both algorithms on the same computer and have a set of inputs representative of the practical use case, this can be a perfectly valid way of making a comparison. Often, we want to make a more ‘theoretical’ comparison, in which we abstract the practical implementation details away.

We use a generic model of computation: a *random-access machine* (RAM). Based on operations that are normally found in real computers the RAM model contains the following operations: arithmetic (add, subtract, multiply, divide, remainder, floor, ceiling), data movement (load, store, copy), and control (conditional and unconditional branch, subroutine call and return). We assume that each operation takes constant time.

To store data, the model contains the data types integer and floating-point.

Algorithm 2 Dijkstra's Shortest Path Algorithm

```
Input: Graph $G = (V, E)$, source $s$, target $t$, edge lengths $(c_e)_{e \in E}$.
Output: A shortest path from $s$ to $t$
$u \leftarrow s$
$\mathit{visited} \leftarrow \{s\}$
$\mathbf{P}[v] \leftarrow P_\infty \; \forall v \in V$        /* to store the shortest path from $s$ to $v$ */
$\mathbf{P}[s] = P_\phi$
while $u \neq t$ and $\mathbf{P}[u] \neq P_\infty$ do
    for all $e = (u, v) \in E$ with $v \notin \mathit{visited}$ do
        if $c(\mathbf{P}[u]) + c_e < c(\mathbf{P}[v])$ then
            $\mathbf{P}[v] \leftarrow \mathbf{P}[u] \cdot (e)$
        end if
    end for
    $u \leftarrow \arg\min_{v \in V \setminus \mathit{visited}} c(\mathbf{P}[v])$        /* The closest unvisited node */
    $\mathit{visited} \leftarrow \mathit{visited} \cup \{u\}$
end while
return $\mathbf{P}[t]$
```

The latter is used to store real numbers. We assume that the size of the data to store one number, called a *word*, is big enough to fit the numbers in the input but is bounded by a constant, i.e., if the input has size n, then a word has size $c \ln(n)$ for some constant $c \geq 1$. Then, arithmetic operations take constant time, and a word cannot store an arbitrary amount of information.

The running time of an algorithm is expressed as a function of the input size. Let n denote the size of the input. Think, for example, of the number of nodes in a graph or the number of elements in a list. Determining the exact running time of an algorithm is usually not worth the effort. Suppose the exact running time of an algorithm is $12n^2 + 6n + 92$. If n grows, the smaller order terms get dominated by the n^2 term, and the same holds for the 12 in front of the n^2. We mainly care about the asymptotic behavior of an algorithm: what happens if the size of the input grows large. So we 'hide' the lower order terms and the coefficients and write $12n^2 + 6n + 92 = \mathcal{O}(n^2)$. More formally:

Definition 2.3.3. Let f and g be two real-valued functions. We write

$$f(n) = \mathcal{O}(g(n))$$

if there exist $c > 0$ and $n_0 > 0$ such that

$$0 \leq f(n) \leq c \cdot g(n) \quad \text{for all } n \geq n_0.$$

Note that f and g do not have to match on the highest order term, for example, $3n = \mathcal{O}(n)$, but also $3n = \mathcal{O}(n^2)$. To asymptotically lower bound functions there is a similar concept.

Definition 2.3.4. Let f and g be two real-valued functions. We write

$$f(n) = \Omega(g(n))$$

if there exist $c > 0$ and $n_0 > 0$ such that

$$0 \leq c \cdot g(n) \leq f(n) \quad \text{for all } n \geq n_0.$$

Example 2.3.5. The insertion sort algorithm in Algorithm 1 iterates through the list (n iterations), and for every iteration i, it does at most $i-1$ swaps. The running time of insertion sort is bounded by $\mathcal{O}(n^2)$.

Example 2.3.6. Suppose we use Algorithm 2 on a graph $G = (V, E, (c_e)_{e \in E})$ with $n = |V|$ nodes. With the right data structures, we can bound the running time of Algorithm 2 by $\mathcal{O}(n^2)$: Lines $1, 2,$ and 4 take constant time, while Line 3 takes $\mathcal{O}(n)$ time. Each node is visited at most once, and so the while loop (Line 5) has at most n iterations. Since a node is connected to at most $n-1$ other nodes, the for loop (Line 6) has at most n iterations. Instead of storing the whole path from s to v in $\mathbf{P}[v]$ we can store only the last edge that is used to end up at v, and, afterward, recursively construct the whole path so that Line 8 takes $\mathcal{O}(1)$ time (constant time). Finally, using a priority queue based on a Fibonacci heap, we can execute Line 11 in $\mathcal{O}(\log(n))$ time. Hence, in total the algorithm takes time $\mathcal{O}(n\,(n + \log(n)) = \mathcal{O}(n^2)$.[1]

To determine the running time in Examples 2.3.5 and 2.3.6 we used a *worst-case perspective*. If a list is reversely sorted, insertion sort takes time $\mathcal{O}(n^2)$, while if the list is already sorted it only takes time $\mathcal{O}(n)$. When analyzing running times of algorithms in this thesis, we are always interested in the worst-case running time, i.e., the longest running time for any input of size n.

We are interested in algorithms that are *efficient* in the worst case. By an efficient algorithm, we denote an algorithm that runs in polynomial time in the input size.

Definition 2.3.7. An algorithm runs in *polynomial time* if there exists a constant $c \in \mathbb{N}$, independent of the input size, such that for inputs of size n, the running time of the algorithm is $\mathcal{O}(n^c)$.

This contrasts with algorithms that run in super-polynomial time, such as those that run in exponential time: $\mathcal{O}(2^n)$, for example.

[1] Observe that if the graph has a small number of edges, the for loop in Line 6 does not have many iterations. In fact, for every edge, there is at most one iteration. We can give a more precise upper bound on the running time of Algorithm 2. If the graph has n nodes and m edges, the running time is $\mathcal{O}(m + n\log(n))$.

Complexity

With the tools in the previous subsection, we can analyze the running time of algorithms. We ended the subsection by stating that we are interested in efficient algorithms. However, an efficient algorithm might not exists for some given problem. We need a way to compare the difficulty of problems.

The two problems we discussed in Examples 2.3.5 and 2.3.6 are of a different nature. The sorting problem requires us to transform the input, while the second one requires us to find the best solution among many feasible solutions. The latter is an *optimization problem* and, in particular, a *minimization problem*. In this thesis, we mostly study optimization problems.

Definition 2.3.8. An *optimization problem* Π is a triple $(\mathcal{I}, F, C)$:

$\mathcal{I}$ is a set of instances.

$F(I)$ denotes a set of feasible solutions for an instance in $I \in \mathcal{I}$.

$C(I, x)$ denotes the cost or value of a feasible solution $x \in F(I)$.

The optimization problem Π is a *minimization problem* if the goal is to find a feasible solution $x \in F(I)$ for an instance $I \in \mathcal{I}$ such that

$$C(I, x) \leq C(I, x') \quad \text{for all } x' \in F(I).$$

The optimization problem Π is a *maximization problem* if the goal is to find a feasible solution $x \in F(I)$ for an instance $I \in \mathcal{I}$ such that

$$C(I, x) \geq C(I, x') \quad \text{for all } x' \in F(I).$$

We are interested in these optimization problems. However, the predominant way to compare problems makes use of *decision problems*. A decision problem is a problem for which the output is limited to two options, either *yes* or *no*. The instances for which the output is *yes* are called *yes-instances*; similarly, instances for which the output is *no* are called *no-instances*. An optimization problem can easily be converted into a decision problem by introducing an extra input number K. For a minimization problem $\Pi = (\mathcal{I}, F, C)$, we can define the following related decision problem:

Problem Decision problem related to minimization problem

Input: An instance $I \in \mathcal{I}$ of a minimization problem $\Pi = (\mathcal{I}, F, C)$, a number K
Output: *yes*, if there is an $x \in F(I)$ with $C(I, x) \leq K$, otherwise *no*.

A maximization problem has a similar related decision problem in which the inequality is reversed.

Note that the remaining definitions in this section will be stated in a slightly informal way, which is enough for our purposes. For formal definitions using formal-language theory, see, for example, Cormen et al. (2009).

A way to show that a decision problem B is ‘more difficult‘ than a decision problem A is by proving that if we can solve B efficiently, then we can solve A efficiently. If we then come up with a polynomial-time algorithm that solves B, we have actually solved both problems efficiently. To prove this, we make use of the notion of a reduction:

Definition 2.3.9. (informal). A decision problem A is *polynomial-time reducible* to a decision problem B if there exists a polynomial-time algorithm that transforms *yes*-instances of A into *yes*-instances of B and *no*-instances of A into *no*-instances of B.

Suppose A is polynomial-time reducible to B, and we have an algorithm that solves B in polynomial time. Then we also have an algorithm that solves A in polynomial time by first transforming an instance of A into an instance of B and then using the polynomial-time algorithm for B.

We define two important classes of decision problems: P and NP. The first is the class of decision problems that can be solved efficiently.

Definition 2.3.10. (informal). A decision problem is a member of the *complexity class* P if there exists a polynomial-time algorithm that solves it.

For the complexity class NP, we need the notion of *verifiability.* Consider the problem of finding a shortest path from node s to node t in a graph. The related decision problem asks whether there exists an s,t-path of length at most K. If, for some instance, we were somehow given a path P of length at most K. We can verify in polynomial time if this path is a so-called *certificate* showing that the instance is a *yes*-instance by checking if P connects s and t and has cost at most K. The class NP consists of decision problems of which *yes*-instances have a polynomial-size certificate that can be verified by a polynomial-time verification algorithm. The polynomial size and time are with respect to the instance size.

Definition 2.3.11. (informal). A decision problem is a member of the complexity class NP if the following holds: (i) there exists a polynomial-time verification algorithm V, and (ii) an instance of this decision problem is a *yes*-instance if and only if there exists a polynomial-size certificate for this instance which can be verified by V.

Any decision problem that is a member of P is also a member of NP, thus P $\subseteq$ NP. However, we do not know if NP is a strict superset of P or if they are equal.[2]

[2] This is one of the Millenium Prize Problems of the Clay Mathematics Institute (Carlson, Jaffe, and Wiles, 2006)

There are problems of which we know that they are a member of NP and that any problem in NP is polynomial-time reducible to it. If we can solve one of these problems in polynomial time, we can solve all problems in NP in polynomial time. These problems are called NP-*complete*.

Definition 2.3.12. (informal). A problem $\Pi \in \text{NP}$ is called NP-*complete* if all problems in NP are polynomial-time reducible to Π.

Suppose we have an NP-complete problem B and suppose we can polynomial-time reduce it to a decision problem A in NP. Then, any problem in NP can be polynomial-time reduced to A by first reducing it to B and then using the reduction to A. The composition of two polynomial-time reductions is also a polynomial-time reduction. So, A is also NP-complete.

The first problem that was shown to be NP-complete is the boolean satisfiability problem (SAT) (Cook, 1971; Levin, 1973). SAT asks to determine if a boolean formula is satisfiable, i.e., if there exists a truth assignment of the literals such that the formula evaluates to true. Since then, a lot of problems have been shown to be NP-complete by polynomial-time reductions from known NP-complete problems.

There are more problems than just the NP-complete problems for which it holds that if we can solve them in polynomial time, then we can solve all problems in NP in polynomial time. First, we need the notion of *oracle access*. If an algorithm has oracle access to a decision or optimization problem Π, the RAM model gets extended with an extra operation that runs in constant time: solving Π.

Definition 2.3.13. (informal). A problem Π is called NP-*hard*[3] if there exists a polynomial-time algorithm for an NP-complete problem Π' when the algorithm has oracle access to Π.

Suppose there exists a polynomial-time algorithm A for an NP-hard problem Π, then there exists a polynomial-time algorithm for the corresponding NP-complete problem Π' by taking the algorithm from the definition and replacing the oracle access by A. Since we now have a polynomial-time algorithm for an NP-complete problem, all problems in NP have a polynomial-time algorithm.

All NP-complete problems are NP-hard. If a problem is NP-hard, then we do not know for sure that a polynomial-time algorithm does not exist, but it also means that until now, nobody has found a polynomial-time algorithm for any NP-hard problem yet (and there are a lot of them (Crescenzi and Kann, 2000; Garey and Johnson, 1979)).

[3] Note that an NP-hard problem can also be an optimization problem.

Algorithm 3 Approximation Algorithm for Vertex Cover

```
Input: Undirected Graph $G = (V, E)$.
Output: A set of vertices covering all edges
1: $S \leftarrow \emptyset$
2: while $E$ is not empty do
3:         Pick $\{u, v\} \in E$
4:         $S \leftarrow S \cup \{u, v\}$
5:         $E \leftarrow E \setminus \{\{w, z\} \in E \mid w = u$ or $w = v\}$
6: end while
7: return $S$
```

Approximation Algorithms

If we assume P $\neq$ NP, then for the optimization problems that are NP-hard, there cannot exist a polynomial-time algorithm that solves them. However, if we relax the requirement of optimality a bit and only require that the solutions are near-optimal, it is sometimes possible to get an efficient algorithm.

Definition 2.3.14. (Approximation algorithm). Let $\Pi = (\mathcal{I}, F, C)$ be a maximization (minimization resp.) problem. Let $\mathrm{OPT}(I)$ be the value or cost of an optimal solution for an instance $I \in \mathcal{I}$. An algorithm ALG is an *α-approximation algorithm* for Π with $\alpha \leq 1$ ($\alpha \geq 1$ resp.) if for every instance I of Π it computes a feasible solution of cost $\mathrm{ALG}(I) \geq \alpha\mathrm{OPT}(I)$ ($\mathrm{ALG}(I) \leq \alpha\mathrm{OPT}(I)$ resp.) in time that is polynomially bounded in the input size of I.

Example 2.3.15. Consider the *Vertex Cover problem*: given an undirected graph $G = (V, E)$, what is the size of the smallest set of vertices S that covers every edge, i.e., every edge has an endpoint in S?

Algorithm 3 iteratively takes an edge $\{u, v\}$ from E, adds both endpoints to S, and removes all edges from E with endpoint u or v.

Let us analyze the size of the set that Algorithm 3 returns. For every two vertices added to S in Line 4, the optimal solution must include at least one. Hence, for any instance I, it holds that

$$\mathrm{OPT}(I) \geq \frac{1}{2}\mathrm{ALG}(I) \implies \mathrm{ALG}(I) \leq 2\mathrm{OPT}(I).$$

Therefore, we conclude that Algorithm 3 is a 2-approximation algorithm for the vertex cover problem.

It is possible to have a reduction for maximization problems that preserves approximation guarantees.

Definition 2.3.16. Let Π and Π' be two maximization problems. We say that there is an *L-reduction* with parameters $a, b > 0$ from Π to Π' if

(i) for each instance I of Π, we can construct in polynomial time an instance I' of Π',

(ii) $\text{OPT}(I') \leq a \cdot \text{OPT}(I)$,

(iii) given a solution of cost Z' for I', we can derive in polynomial time a solution of cost Z for I such that

$$\text{OPT}(I) - Z \leq b \cdot (\text{OPT}(I') - Z').$$

In particular, the definition above implies that if there is an L-reduction with parameters a and b from Π to Π' and Π' admits an α-approximation algorithm, then there is a $(1 - ab(1 - \alpha))$-approximation algorithm for Π (Williamson and Shmoys, 2011).

Similar to showing that problems are NP-hard, we can also show for some problems that approximating the optimal solution within a certain factor is NP-hard. If for a maximization (minimization resp.) problem, it is NP-hard to approximate it within a factor $\alpha - \epsilon$ ($\alpha + \epsilon$ resp.) for any $\epsilon > 0$ we say that the problem is *α-inapproximable*.

2.4 Algorithmic Game Theory

The formal definition of a strategic game is the following:

Definition 2.4.1. A game Γ in strategic form consists of

a set of players $N = [n]$.

a strategy set S_i for each player $i \in N$.

a cost function $c_i : S_1 \times \cdots \times S_n \to \mathbb{R}$ for each player i.

We write $S = S_1 \times \cdots \times S_n$. A vector $s = (s_1, \ldots, s) \in S$ is called a *strategy profile*. A game is called *finite* if S is finite.

We assume that all players know the cost functions and strategies of all other players: they have *full information*. Often, we want to let one player change their strategy. As a notational shorthand, we write $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ for the strategy profile without the strategy of player i. Similarly $S_{-i} = \times_{j \neq i} S_j$. Further, we denote by (s_i', s_{-i}) the strategy profile where player i plays some strategy s_i' and the other $n - 1$ players play their strategy in s_{-i}.

When players have a cost function, we take the perspective that players want to minimize their cost function. When we want to take the perspective of players maximizing some function, then we usually have utility functions u_i instead of the cost functions c_i. In this section, we keep the minimization perspective, and minimizing negative costs can be identified with maximizing utility.

1 \ 2	Silent	Betray
Silent	1 1	0 3
Betray	3 0	2 2

(a) Prisoner's Dilemma

1 \ 2	H	T
H	1 −1	−1 1
T	−1 1	1 −1

(b) Matching Pennies

Figure 2.3: Two examples of strategic games.

Example 2.4.2. Two examples of games in strategic form with two players are given in Figure 2.3. The strategies of player 1 are the row names, and the strategies of player 2 are the column names. Note that both players have the same strategy set. The cost function of player 1 is defined by the numbers in the bottom left of the cells, while the numbers in the top right are the costs of player 2.

In the Prisoner's Dilemma, two members of a criminal gang are arrested and are both offered a deal: if they betray the other while the other stays silent, they will walk away without sentence while the other has to serve three years in prison. However, if they both choose to betray, they will both serve two years in prison, and if they both stay silent, they will both serve one year in prison.

In Matching Pennis, there are two players, both with a coin that can show Heads or Tails. Player 1 would like to see that both coins show the same side, i.e., both H or both T, while player 2 would like to see different sides.

Example 2.4.3. The cost function can also be a bit more elaborate. Suppose we have m machines (all with speed 1) and n jobs with processing times $p_1 \leq p_2 \leq \ldots \leq p_n$. Every job is controlled by a player. The player controls on which machine the job is scheduled.

There are various possible cost functions. One option is that a job is only finished when all jobs on the machine are finished. Every job then incurs a cost equal to the sum of processing times of the jobs on that machine. Another option is that on each machine, jobs are scheduled from shortest to longest processing time, and the cost of a job is the time it gets completed.

We can represent a strategy profile as in Figure 2.4, where the numbers in the rectangles indicate the index of the job, and the size represents its processing time.

Equilibria

We are interested in the *stable outcomes* of games. In the Prisoner's Dilemma, as defined in Figure 2.3(a), it does not matter for player 2 if player 1 chooses Silent or Betray, it is always in their best interest to play Betray. The same holds for player 1.

Definition 2.4.4. Let Γ be a game. Let $s_i, t_i \in S_i$ be two strategies of player i. Strategy s_i *(weakly) dominates* r_i if for all $\boldsymbol{s}_{-i} \in S_{-i}$

$$c_i(s_i, \boldsymbol{s}_{-i}) \geq c_i(r_i, \boldsymbol{s}_{-i}).$$

and for at least one $\boldsymbol{s}_{-i}$ this inequality is strict. The strategy s_i *strictly dominates* r_i if the above inequality is strict for all $\boldsymbol{s}_{-i} \in S_{-i}$.

A strategy $s_i \in S_i$ is *(weakly) dominating* if it (weakly) dominates all other strategies $r_i \in S_i$. A strategy profile $\boldsymbol{s} \in S$ consisting of only (weakly) dominating strategies is called a *(weakly) dominant strategy equilibrium.*

In the Prisoner's Dilemma $(\mathit{Betray}, \mathit{Betray})$ is a dominant strategy equilibrium.

Consider the machine scheduling instance in Figure 2.4 where the cost of a job is the sum of processing times of jobs on that machine

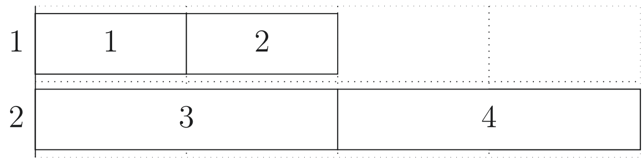


Figure 2.4: Machine scheduling example

If job 3 or 4 (currently cost 4) switches to machine 1, there is no gain as they will have the same cost. For jobs 1 and 2 there is also no incentive to change machines as that will increase their cost. So again we are in some sort of equilibrium. But, job 4 choosing machine 2 is not a weakly dominant strategy because if the other three jobs are already on machine 2 choosing machine 1 is the better strategy.

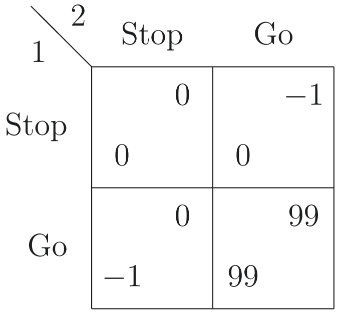
The game is actually in a *pure Nash Equilibrium*: no player has an incentive to *unilaterally* deviate from their strategy.

Definition 2.4.5. A *pure Nash Equilibrium* (PNE) is a strategy profile $\boldsymbol{s} \in S$ such that for every player i

$$c_i(\boldsymbol{s}) \leq c_i(\boldsymbol{s}_{-i}, s_i') \quad \text{for all } s_i' \in S_i.$$

Write PNE(Γ) for the set of all pure Nash equilibria of a strategic game Γ.

So, in the Prisoner's Dilemma, the strategy profile (Betray, Betray) is a PNE (any weakly dominant equilibrium is also a PNE), and in the machine scheduling game, the strategy profile $(1, 1, 2, 2)$, where players 1 and 2 choose machine 1 and players 3 and 4 choose machine 2 is a PNE. However, the Matching Pennies game



1 \ 2	Stop	Go
Stop	0 0	-1 0
Go	0 -1	99 99

Figure 2.5: Traffic Light game

in Figure 2.3(b) does not have a PNE. Player 1 wants to play the same strategy as player 2, while player 2 wants to play a different strategy from player 1. Instead, we can allow players to randomize their strategy.

Let σ_i be a probability distribution over S_i for all i. We call σ_i a *mixed strategy* for player i and write $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$ for the mixed strategy profile. This strategy profile is the product distribution of the individual mixed strategies: players are randomizing independently of each other. Write $\mathbb{E}_{\boldsymbol{s} \sim \boldsymbol{\sigma}}[c_i(\boldsymbol{s})]$ for the expected cost of player i when all the players randomize their strategy according to $\boldsymbol{\sigma}$.

Definition 2.4.6. A *mixed Nash Equilibrium* (MNE) is a randomized strategy profile $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$ such that for every player i

$$\mathbb{E}_{\boldsymbol{s} \sim \sigma}[c_i(\boldsymbol{s})] \leq \mathbb{E}_{\boldsymbol{s}_{-i} \sim \boldsymbol{\sigma}_{-i}}[c_i(\boldsymbol{s}_{-i}, s_i')] \quad \text{for all } s_i' \in S_i.$$

Write MNE(Γ) for the set of all mixed Nash equilibria of a strategic game Γ.

Observe that every pure Nash equilibrium is a mixed Nash equilibrium where every player plays a single strategy with probability 1.

A strategy profile in which no player wants to unilaterally deviate in the Matching Pennies game is when both players play both strategies with probability $\frac{1}{2}$.

As we saw for Matching Pennies, a pure Nash equilibrium does not always exist. However, for mixed Nash equilibria, Nash (1951) showed that any finite strategic game has at least one.

Theorem 2.4.7. ((Nash, 1951)). *Any finite game has a mixed Nash equilibrium*

In a mixed strategy profile, every player is randomizing independently. This does not always give us enough freedom to express any strategy profile that might be considered stable. For example, take a look at the Traffic Light game in Figure 2.5, in which two cars are approaching an intersection and can choose to either Stop or Go.

There are two pure Nash equilibria (Stop, Go) and (Go, Stop), but in both, one of the cars has cost 0 (while the other has cost -1). If we allow mixed Nash equilibria, then, next to the two pure Nash equilibria, there is also the strategy profile where both players choose Stop with probability 99/100, and Go with probability 1/100. Both players have a cost of 0, and more importantly, there is a probability of 1/10000 to get a crash.

Preferably, we would like the strategy profile in which the strategies (Stop, Go) and (Go, Stop) are both played with probability 0.5, to be a stable outcome. But, this is impossible with a mixed strategy profile: if players randomize independently and they have to randomize both over Stop and Go, the strategies (Stop, Stop), and (Go, Go) will also get positive probability.

What about a traffic light? When one car has a green light, the other car will surely stop for its red light. This analog is what is achieved with the definition of a correlated equilibrium.

Definition 2.4.8. Let $\boldsymbol{\sigma}$ be a probability distribution over $S = S_1 \times \cdots \times S_n$. The profile $\boldsymbol{\sigma}$ is called a *correlated equilibrium* (CE) if for every player i and for every deviation function $m_i : S_i \to S_i$ it holds that

$$\mathbb{E}_{\boldsymbol{s}\sim\boldsymbol{\sigma}}[c_i(\boldsymbol{s})] \leq \mathbb{E}_{\boldsymbol{s}\sim\boldsymbol{\sigma}}[c_i(\boldsymbol{s}_{-i}, m_i(s_i))].$$

Write CE(Γ) for the set of all correlated equilibria of a strategic game Γ.

For all strategy profiles $\boldsymbol{s} \sim \boldsymbol{\sigma}$ that suggest player i to play s_i, it is in player i's best interest to play s_i instead of some other strategy $m_i(s_i)$. Consider the correlated equilibrium which selects both (Stop, Go) and (Go, Stop) with probability $\frac{1}{2}$ in the Traffic Light game. If player 1 observes Stop, then they know player 2 observed Go, and so it is in their best interest to Stop.

Finally, there is the concept of a coarse correlated equilibrium. In this case, a player is allowed to do a fixed deviation instead of a deviation that depends on the draw of the distribution. It arises when we approach the game from an online learning perspective. If players use a no-regret strategy they will start playing some mixed strategy. Taking the average over time of all the mixed strategies is approximately a coarse correlated equilibrium (Hannan, 1957; Roughgarden, 2016).

Definition 2.4.9. Let $\boldsymbol{\sigma}$ be a probability distribution over $S = S_1 \times \cdots \times S_n$. The profile $\boldsymbol{\sigma}$ is called a *coarse correlated equilibrium* (CCE) if for every player i it holds that

$$\mathbb{E}_{\boldsymbol{s}\sim\boldsymbol{\sigma}}[c_i(\boldsymbol{s})] \leq \mathbb{E}_{\boldsymbol{s}_{-i}\sim\boldsymbol{\sigma}_{-i}}[c_i(\boldsymbol{s}_{-i}, s_i')] \quad \text{for all } s_i \in S_i.$$

Write CCE(Γ) for the set of all coarse correlated equilibria of a strategic game Γ.

Since any pure Nash equilibrium is also a mixed Nash equilibrium, any mixed Nash equilibrium is also a correlated equilibrium, and any correlated equilibrium is also a coarse correlated equilibrium, we have the following chain of inclusions:

$$\text{PNE}(\Gamma) \subseteq \text{MNE}(\Gamma) \subseteq \text{CE}(\Gamma) \subseteq \text{CCE}(\Gamma). \tag{2.2}$$

Inefficiency

A central question in algorithmic game theory is how *inefficient* equilibria are. For example, in the machine scheduling instance, we observed that the strategy profile in Figure 2.4 is a Nash equilibrium. The last job finishes at time 4. What if jobs 1 and 3 are scheduled on one machine and jobs 2 and 4 on the other? Then the last job finishes at time 3. The 'cost' of letting players play strategically is a factor $\frac{4}{3}$ larger than the optimal cost.

To compare strategy profiles on a global level, we need a *social cost function*: $SC : S_1 \times \cdots \times S_n \to \mathbb{R}$. A *socially optimal outcome* is then a strategy profile minimizing the social cost function. We denote the socially optimal outcome by $\boldsymbol{s}^*$, and in the remainder of this section, we assume $SC(\boldsymbol{s}^*) > 0$.

Given a strategic game Γ, the *pure price of anarchy* (PNE-PoA) is the largest relative gap between the social cost of a pure Nash equilibrium and the optimal social cost.

Definition 2.4.10. For a strategic game Γ with $\mathrm{PNE}(\Gamma) \neq \emptyset$, we define the *pure price of anarchy* as

$$\text{PNE-PoA}(\Gamma) = \frac{\max_{\boldsymbol{s} \in \mathrm{PNE}(\Gamma)} SC(\boldsymbol{s})}{SC(\boldsymbol{s}^*)}.$$

We do not have to restrict ourselves to the set of pure Nash equilibria. We analogously define the price of anarchy for mixed, correlated and coarse correlated equilibria.

Definition 2.4.11. Let $X \in \{\mathrm{PNE}, \mathrm{MNE}, \mathrm{CE}, \mathrm{CCE}\}$. For a strategic game Γ with $X(\Gamma) \neq \emptyset$ we define the X-PoA as

$$X\text{-PoA}(\Gamma) = \frac{\max_{\boldsymbol{s} \in X(\Gamma)} SC(\boldsymbol{s})}{SC(\boldsymbol{s}^*)}.$$

Assuming that $X(\Gamma) \neq \emptyset$ for $X \in \{\mathrm{PNE}, \mathrm{MNE}, \mathrm{CE}, \mathrm{CCE}\}$, we can use (2.2) to see that the following chain of inequalities holds:

$$\text{PNE-PoA}(\Gamma) \leq \text{MNE-PoA}(\Gamma) \leq \text{CE-PoA}(\Gamma) \leq \text{CCE-PoA}(\Gamma).$$

Often, we are interested in the price of anarchy of a class of games. In particular, we are interested in the worst-case price of anarchy. Let $\mathcal{G}$ be a class of games. For example, the class of all machine scheduling games, any number of jobs, any number of machines, and any processing times of the jobs. The price of anarchy with respect to $X \in \{\mathrm{PNE}, \mathrm{MNE}, \mathrm{CE}, \mathrm{CCE}\}$ of this class of games is

$$X\text{-PoA}(\mathcal{G}) = \sup_{\Gamma \in \mathcal{G}} X\text{-PoA}(\Gamma).$$

A *cost-minimization game* is a game in strategic form together with a social cost function that is the sum of the individual costs, i.e., $SC(\boldsymbol{s}) = \sum_{i \in N} c_i(\boldsymbol{s})$. A powerful technique to prove bounds on the price of anarchy in cost-minimization games is *smoothness* (Roughgarden, 2015).

Definition 2.4.12. A cost-minimization game Γ is (λ, μ)-*smooth*, with $\lambda > 0$, $\mu < 1$, if for every two strategy profiles $\boldsymbol{s}, \boldsymbol{s}^* \in S$ it holds that

$$\sum_{i \in N} c_i(s_i^*, \boldsymbol{s}_{-i}) \leq \lambda \cdot SC(\boldsymbol{s}^*) + \mu \cdot SC(\boldsymbol{s}).$$

If a cost-minimization game is (λ, μ)-smooth, then for a pure Nash equilibrium $\boldsymbol{s}$ and optimal strategy profile $\boldsymbol{s}^*$ we have

$$\begin{aligned} SC(\boldsymbol{s}) &= \sum_{i \in N} c_i(\boldsymbol{s}) \\ &\leq \sum_{i \in N} c_i(s_i^*, \boldsymbol{s}_{-i}) \\ &\leq \lambda \cdot SC(\boldsymbol{s}^*) + \mu \cdot SC(\boldsymbol{s}), \end{aligned} \tag{2.3}$$

where we first make use of the definition of $SC(\boldsymbol{s})$, secondly apply the pure Nash equilibria conditions for every player, and finally, use the assumption that the game is (λ, μ)-smooth. Rewriting this implies

$$\frac{SC(\boldsymbol{s})}{SC(\boldsymbol{s}^*)} \leq \frac{\lambda}{1 - \mu}.$$

Hence, the pure price of anarchy is at most $\frac{\lambda}{1-\mu}$.

The *robust price of anarchy* is the best price of anarchy bound that can be proved via the smoothness technique.

Definition 2.4.13. The *robust price of anarchy* of a cost-minimization game Γ is

$$\text{RPoA}(\Gamma) = \inf \left\{ \frac{\lambda}{1 - \mu} \mid \Gamma \text{ is } (\lambda, \mu)\text{-smooth}, \lambda > 0, \mu < 1 \right\}.$$

We required the inequality in (2.3) to hold for every combination of $\boldsymbol{s}, \boldsymbol{s}^*$, not only for a pure Nash equilibrium and optimal solution. This extra requirement makes it possible to extend the price of anarchy bound up to coarse correlated equilibria, which Roughgarden (2015) did in the proof of the following extension theorem.

Theorem 2.4.14. (Roughgarden, 2015). *For every cost-minimization game* Γ

$$\text{CCE-PoA}(\Gamma) \leq \text{RPoA}(\Gamma).$$

2.5 Mechanism Design

In algorithmic game theory, we are given a game and try to analyze its outcomes. In mechanism design, we start with preferable outcomes and try to design the game such that it ends up in one of these outcomes. It is therefore also sometimes called reverse game theory.

In this thesis, we are mostly interested in mechanisms in which we can use money to create incentives. We have a set of alternatives A and a set of n players $N = [n]$. Each player $i \in N$ has a valuation function $v_i : A \to \mathbb{R}$ which specifies how much the player values the various alternatives. For each player i there is a set $V_i \subseteq \mathbb{R}^A$ that contains all the possible valuation functions for player i. We define $V = V_1 \times \cdots \times V_n$.

Definition 2.5.1. A (direct-revelation)[4] mechanism $\mathcal{M} = (f, \boldsymbol{p})$ consists of a social choice function $f : V \to A$ and payments functions $p_1, \dots, p_n : V \to \mathbb{R}$.

The players reveal some bid $b_i \in V_i$ (which may be different from their actual v_i). Let $\boldsymbol{b} = (b_1, \dots, b_n)$. Then the mechanism chooses some alternative $f(\boldsymbol{b})$ and specifies an amount of money $p_i(\boldsymbol{b})$ that each player has to pay (or receive if $p_i(\boldsymbol{b})$ is negative). The utility that player i obtains is given by

$$u_i(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}).$$

This utility function is called *quasilinear.*

The *social welfare* of a mechanism is the overall valuation obtained by the bidders, i.e.,

$$\mathrm{SW}(\boldsymbol{b}) = \sum_{i \in N} v_i(f(\boldsymbol{b})).$$

Complete and Incomplete Information. We are mainly interested in the *complete information* setting in which each player knows the utility functions and possible valuations of all players, but the private valuation of each player is unknown to the others. In, for example, the *incomplete information* model of Harsanyi (1967) the valuation function v_i of player i is drawn from V_i according to some probability distribution $\pi_i : V_i \to [0, 1]$. Let $\pi = \pi_1 \times \cdots \pi_n$ be the product distribution of $\pi_1, \cdots, \pi_n$. The distribution π is *publicly* known, but the drawn valuations profiles $v_1, \dots, v_n$ are *private* information. The players can use this knowledge to determine their bidding strategy. Some results in this thesis can be extended to this incomplete information setting, and we will mention it when this is the case. But, as mentioned, we are mainly interested in a *prior-free* setting where the prior distribution of the agent's valuation function is *unknown.*

[4] A general mechanism does not need to be direct. There can be multiple rounds of interaction between the players and the mechanism designer. At first glance, one may believe this gives much more flexibility. However, due to the revelation principle (Myerson, 1981), we know that if for a social choice function f there exists a general mechanism such that f is implemented in dominant strategies, i.e., the players make optimal strategic decisions and the mechanism selects the outcome it would have picked knowing the players' private information, then there also exists a truthful (defined below) direct-revelation mechanism that implements f. So we will only consider direct-revelation mechanisms.

Most of the mechanisms we consider are auctions. In particular, *sealed-bid* auctions. In a sealed-bid auction, the players submit their bids privately to the auctioneer such that the other players do not learn about the bid before they submit their own bid. It is possible that after the auction completes, bids are published for verification purposes.

Example 2.5.2. Consider a sealed-bid single-item auction with n players. The set of alternatives for this auction is $A = N$, where outcome $i \in A$ implies that player i receives the item. Each player only values winning and has a value of 0 for not winning. Instead of a valuation function v_i we have a value $v_i \in \mathbb{R}$ denoting the value that player i has for winning the item. The players submit bids $\boldsymbol{b} = (b_1, \dots, b_n)$, and the social choice function f chooses the outcome i^*, where i^* is the index of the player with the highest bid in $\boldsymbol{b}$. The winning player has to pay some value p. Hence, if player i wins and has to pay p, their utility is $v_i - p$, while if player i does not win, they have a utility of 0.

To maximize the social welfare, the item should end up with the player who values it the highest. But we do not *know* the v_i's, we only know what the players disclose to the mechanism, i.e., the b_i's. Yet, we still want a mechanism that cannot be strategically manipulated and makes sure we select the player with the highest v_i. We can use the payments to incentivize the players to do what we want.

Suppose the winning player has to pay what they bid. If the second-highest bid was strictly smaller than the highest bid, the winner had an incentive to report a bid that is smaller than their actual valuation. But, of course, the winning bidder does not know what the second-highest bid will be. A way to incentivize the winning bidder to still bid their actual valuation is to charge the highest losing bid as payment.

This is also known as the Vickrey (second-price) auction and it has the following nice property.

Proposition 2.5.3. *Consider a sealed-bid single-item auction where the payment of the winner is equal to the second-highest bid. For every $\boldsymbol{b} \in V$ and every $v_i \in V_i$ it holds that $u_i(v_i, \boldsymbol{b}_{-i}) \geq u_i(\boldsymbol{b})$.*

This property, where the utility of player i is maximized by revealing their true valuation is called *truthfulness* (or *incentive-compatible*, or *strategyproofness*). More generally:

Definition 2.5.4. A mechanism $\mathcal{M} = (f, \boldsymbol{p})$ is called *truthful* if for each player i with valuation function $v_i \in V_i$, for all $\boldsymbol{b} \in V$ it holds that

$$v_i(f(v_i, \boldsymbol{b}_{-i})) - p_i(v_i, \boldsymbol{b}_{-i}) = u_i(v_i, \boldsymbol{b}_{-i}) \geq u_i(\boldsymbol{b}) = v_i(f(b_i, \boldsymbol{b}_{-i})) - p_i(b_i, \boldsymbol{b}_{-i}).$$

Another desirable property is that it is never better for a player to not participate in the auction at all. We capture this by requiring that a player never has to pay more than their bid.

Definition 2.5.5. A mechanism $\mathcal{M} = (f, p)$ is called *individually rational (IR)* if for all possible reported valuation functions $\boldsymbol{b} \in V$ it holds that $p_i(\boldsymbol{b}) \leq b_i(f(\boldsymbol{b}))$.

Depending on the situation, it makes sense that players do not receive money from the mechanism. If that is true, we call the mechanism *pay-only* or say that it has the *no positive transfers* property.

Definition 2.5.6. A mechanism $\mathcal{M} = (f, p)$ has the *no positive transfers (NPT)* property if for all possible reported valuation functions $\boldsymbol{b} \in V$ it holds that $p_i(\boldsymbol{b}) \geq 0$.

A family of mechanisms that are truthful and maximize social welfare is the family of Vickrey-Clarke-Groves mechanisms:

Definition 2.5.7. A mechanism $M = (f, \mathbf{p})$ is called a *Vickrey-Clarke-Groves (VCG) mechanism* if the following two conditions are satisfied:

1. $f(\mathbf{b}) \in \arg\max_{a \in A} \sum_{i \in N} b_i(a)$;
2. for every player $i \in N$ there is a function $h_i : V_{-i} \to \mathbb{R}$ such that
$$p_i(\mathbf{b}) = h_i(\mathbf{b}_{-i}) - \sum_{j \neq i} b_j(f(\mathbf{b})).$$

VCG mechanisms allow for different instantiations of functions h_i to define the payments of the players. However, if the valuation functions are non-negative and one additionally insists on satisfying both NPT and IR, then there remains a unique payment rule due to Clarke (1971): A VCG mechanism $(f, \mathbf{p})$ implements the *Clarke pivot rule* if for every player $i \in N$ we have that $h_i(\mathbf{b}_{-i}) = \sum_{j \neq i} b_j(a^{-i})$, where $a^{-i} \in \arg\max_{a \in A} \sum_{j \neq i} b_j(a)$ is an alternative that maximizes the social welfare if player i would not be present.

The following is due to (Clarke, 1971; Groves, 1973; Vickrey, 1961).

Proposition 2.5.8. *Every VCG mechanism is truthful. The VCG mechanism that uses the Clarke pivot rule satisfies NPT. Further, if all valuation functions of the players are non-negative, then it also satisfies IR.*

Inefficiency of Mechanisms

Consider the first-price auction. Although it is not truthful, it is used a lot in practice. Because it is used, it is important to understand the inefficiencies that arise.

If the mechanism $\mathcal{M} = (f, p)$ is fixed we can view this is a game where every player i wants to maximize their utility u_i, and their strategy set is $S_i = V_i$.

The definitions of PNE, MNE, CE and CCE carry over from the cost minimization setting except that inequalities are reversed. For example, a bid vector $\boldsymbol{b} \in V$ is a PNE if for all $i \in N$

$$u_i(b_i, \boldsymbol{b}_{-i}) \geq u_i(b'_i, \boldsymbol{b}_{-i}) \quad \text{for all } b'_i \in V_i.$$

We prefer to keep the price of anarchy a number that is at least 1 and thus, for $X \in \{\text{PNE}, \text{MNE}, \text{CE}, \text{CCE}\}$ it is defined as

$$X\text{-PoA}(\mathcal{M}) = \frac{\text{SW}(\boldsymbol{b}^*)}{\min_{\boldsymbol{b} \in X(\mathcal{M})} \text{SW}(\boldsymbol{b})},$$

where $\boldsymbol{b}^* \in V$ is a bidding profile maximizing the social welfare.

Example 2.5.9. Consider a single-item second-price auction with two players. Player 1 has a value of $v_1 = 1$ while player 2 has a value of $v_2 = \epsilon$ (for some small $\epsilon > 0$) for winning the item. The optimal social welfare of 1 is achieved when player 1 wins with, for example, the bidding profile $\boldsymbol{b}^* = (1, \epsilon)$. Consider the bidding profile $\boldsymbol{b} = (\epsilon, 1)$. Player 2 wins and pays ϵ for a utility of $\epsilon - \epsilon = 0$. While player 1 loses and has a utility of 0. The social welfare is ϵ. Both players have no incentive to deviate. Hence,

$$\text{PNE-PoA} = \frac{\text{SW}(\boldsymbol{b}^*)}{\min_{\boldsymbol{b} \in \text{PNE}} \text{SW}(\boldsymbol{b})} \geq \frac{1}{\epsilon}.$$

Letting $\epsilon \to 0$, gives us an instance with PNE-PoA$\to \infty$.

To prevent this pathological example often a *no-overbidding* assumption is made. In a single-item auction, this means that $b_i \leq v_i$ for all $i \in N$. Assuming the no-overbidding assumption the price of anarchy of a single item second-price auction is 1. In Chapter 6 we will see the effect of the no-overbidding assumption on multi-unit auctions.

Chapter 3

Approximate Pricing in Networks

3.1 Introduction

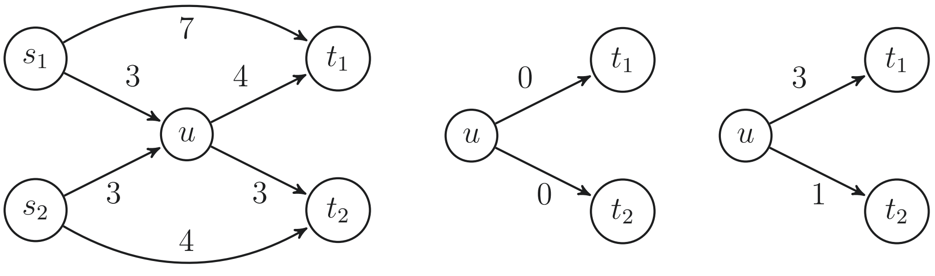
One of the most prominent ways to quantify which nodes are more ‘important’ than others in a network is with *betweenness centrality* (Freeman, 1977). The more frequently a node appears as an intermediate node on the shortest paths between all pairs of nodes in the network, the higher its betweenness centrality. In many applications, it is useful to know which nodes have a high betweenness centrality. For example, in a telecommunications network, the nodes with high betweenness centrality will have more information passing through them, giving them more control. In social networks, the nodes with high betweenness centrality correspond to highly influential people (Das, Samanta, and Pal, 2018). In the international tax treaty network, betweenness centrality can be used to identify conduit countries, i.e., intermediary countries on a route via which corporations send their money (Polak, 2014; Riet and Lejour, 2018).

Depending on what the network represents, it can be beneficial for a node to put effort into increasing its betweenness centrality. For example, if a country is able to increase the amount of money that is sent through it in a worldwide financial network, it can attract more jobs in the financial sector. When a person in a social network improves their betweenness centrality, they become more influential and therefore more appealing to advertisers. One way of increasing the betweenness centrality of a node is by inserting edges (Bergamini et al., 2018). Another one, on which we focus in this chapter, is to change the cost of edges in the network.

The latter approach is well-motivated. For example, when an airport lowers its fees for airlines to use the airport, this can attract more flights and thus more passengers who will spend more on shopping. But, clearly, if the loss in airport fees is higher than the increase in revenue from the shopping, then lowering the prices is not worth it. The same holds for a country lowering taxes in an international tax treaty network: lowering the tax rate increases the money flowing through the country, possibly attracting more jobs, but if the tax income goes down too much, it is not profitable. In these settings, one wants to maximize revenue instead of betweenness centrality.

In this chapter, we take the perspective of a single node in a network. In the network, there are various commodities and each commodity has a certain flow demand that needs to be transported from a source to a destination node. We view this as a (multi-commodity) flow that sends flow from source nodes to their respective destination nodes along shortest paths. If there are multiple shortest paths from a source to a destination node the flow splits uniformly. Suppose a designated node u can change the cost of its outgoing edges. Changing the costs will change (part of) the shortest paths, and this will indirectly redirect (part of) the flows going through the network. For every outgoing edge, the node u earns revenue equal to the cost of that edge multiplied by the amount of flow going through it. The node u either wants to maximize the flow going through it or maximize its revenue.

To get intuition for the context, we take a look at the following example:



(a) Example graph

(b) Optimal costs for maximizing flow

(c) Optimal costs for maximizing revenue

Example 3.1.1. Consider the graph in Figure 3.1(a). Suppose we have two commodities of weight one. One starts in s_1 and ends in t_1 and the other starts in s_2 and ends in t_2.

Currently, both s_1, t_1-paths for the first commodity have equal costs, so the flow splits evenly. For the second commodity, taking the edge that goes directly from s_2 to t_2 is the cheaper option.

Node u can change the cost of the edges (u, t_1) and (u, t_2). It is easy to verify that setting the cost of both edges to 0 maximizes the flow going through u, while setting the cost of (u, t_1) to 3[1] and of (u, t_2) to 1 maximizes the revenue.

[1] For reasons outlined in Section 3.2.2 the costs are assumed to be integers.

In the example above, we could change both outgoing edges, but we might only have the resources to change the cost of one edge. In that case, we have the extra difficulty of choosing on which edge we want to change the cost. Let τ be the number of edges for which we are allowed to change the cost. For both objectives and for different values of τ we get different results. Depending on whether τ is one, equal to the outdegree of u or some number in between, we show that some versions are polynomial-time solvable while others are NP-hard. In the latter case, we resort to approximation algorithms.

We study approximation algorithms because they give us a tool to find near-optimal solutions in polynomial time for problems for which we do not expect an optimal polynomial-time algorithm to exist, such as, for example, NP-hard problems. Approximation algorithms complement hardness results perfectly as they take the same worst-case perspective. They can both serve as a mathematically rigorous basis on which to study heuristic algorithms and give inspiration for heuristics that can be used in practice. For a more elaborate exposition on why one wants to study approximation algorithms, we refer to Williamson and Shmoys (2011).

We do not only prove NP-hardness but also show that some versions are inapproximable for various approximation ratios. The approximation factors of the approximation algorithms we derive (almost) match the inapproximability factors, showing that our algorithms are essentially best possible in terms of approximation guarantees.

3.1.1 Our Contributions

We introduce and study the following *Network Pricing Problem* (**NPP**): We are given a directed graph $G = (V, E)$ with non-negative edge costs $(c_e)_{e \in E}$, k commodities $(s_i, t_i, w_i)_{i \in [k]}$, a designated node $u \in V$ and a natural number τ. Each commodity $i \in [k]$ is represented by a source-target pair $(s_i, t_i) \in V \times V$ and demand (or weight) $w_i > 0$, specifying that w_i units of flow are sent from s_i to t_i along the shortest s_i, t_i-paths (with respect to $(c_e)_{e \in E}$). The weight w_i of each commodity i is split evenly over all the shortest s_i, t_i-paths. Let $d^+(u)$ denote the number of outgoing edges of u. Suppose we can change the costs of τ outgoing edges of u with $1 \leq \tau \leq d^+(u)$, while the costs of all other edges remain fixed; we also say that we *price* (or *tax*) the edges of u. Our goal is to optimally price τ outgoing edges of u such that (i) the total flow passing through u is maximized (**FLOW-NPP**), or (ii) the total revenue (i.e., flow times tax) through u is maximized (**REV-NPP**).

As it turns out, the problems behave rather differently in terms of hardness and approximability, depending on the objective under consideration and the parameter τ. More specifically, our main findings are as follows:

1. We show that **FLOW-NPP** can be solved in polynomial time when a constant

number of edges or almost all edges of u can be priced.

2. In contrast, we prove that **FLOW-NPP** is NP-hard and $(1 - 1/e)$-inapproximable (even for the special case of unit demands) if τ is part of the input. Further, we show that a natural greedy algorithm achieves an approximation guarantee of $(1 - 1/e)$ (which is best possible).

3. We show that **REV-NPP** can be solved in polynomial time if only one edge can be priced. On the other hand, **REV-NPP** becomes NP-hard and $(1 - 1/e)$-inapproximable if τ is part of the input. We also show that the greedy algorithm might perform arbitrarily badly in this case.

4. We prove that already in the unit demand setting, **REV-NPP** is highly inapproximable if all outgoing edges of u can be priced; more specifically, we prove that the problem is both $1/d^{+}(u)^{1/2-\varepsilon}$-inapproximable and $1/\log^{1-\epsilon}(k)$ in this case.

5. In light of this intractability result, we derive algorithms for the following special cases of **REV-NPP**.

   > First, we show that the single-commodity case is polynomial-time solvable. This result also constitutes an important building block for our *uniform pricing algorithms* (i.e., all edges are priced the same).
   >
   > Then, we focus on the unit demand setting and derive a (tight) H_k-approximate uniform pricing algorithm.

6. Finally, we show that our uniform pricing algorithm extends to the general setting and provides a $\max\{1/d^{+}(u), 1/k\}$-approximation algorithm for **REV-NPP**.

Our results for **FLOW-NPP** mostly follow by using standard arguments for submodular function maximization. In contrast, we need to establish several new ideas and exploit structural insights to derive our results for **REV-NPP** (which constitute the main technical contributions of this chapter).

We conclude with some (preliminary) experimental findings on an international tax treaty network based on real data. Our experiments indicate that our uniform pricing algorithm computes tax rates that would significantly increase the current tax revenue of the Netherlands (by a factor 68) and is at least within 51% of the optimal revenue (which is much better than the worst-case approximation guarantee suggests).

3.1.2 Related Work

The problem of increasing the centrality of a node in a network has been widely investigated for different centrality measures. For example, boosting the popularity of web pages by increasing their page rank has been studied intensively (Avrachenkov and Litvak, 2006; Olsen and Viglas, 2014), with a particular focus on “fooling” search engines (e.g., through link farming (Wu and Davison, 2005)). The problem has also been considered for other centrality measures such as closeness centrality (Crescenzi, D’Angelo, et al., 2015, 2016), betweenness centrality (Bergamini et al., 2018), coverage centrality (D’Angelo, Olsen, and Severini, 2019), eccentricity (Demaine and Zadimoghaddam, 2010; Perumal, Basu, and Guan, 2013), average distance (Meyerson and Tagiku, 2009) and some measures related to the number of paths passing through a given node (Ishakian et al., 2012). Below, we give a few representative references only; most of these works focus on edge additions to increase the centrality.

Meyerson and Tagiku (2009) give a constant-factor approximation algorithm for the problem of minimizing the average shortest-path distance between all pairs of nodes by adding shortcut edges. Several algorithms are proposed by Papagelis, Bonchi, and Gionis (2011) and Parotsidis, Pitoura, and Tsaparas (2015) and are experimentally shown to perform well in practice. Bauer et al. (2012) study the problem of minimizing the average number of hops in shortest paths. They prove that the problem cannot be approximated within a logarithmic factor and provide respective approximation algorithms. Bilò, Gualà, and Proietti (2012) and Demaine and Zadimoghaddam (2010) consider the problem of minimizing the diameter of a graph and provide constant factor approximation algorithms.

The problem of maximizing revenue by pricing the edges of a graph has been studied in several works. These problems are known under different names, such as the *network (or highway) pricing problem* (Brotcorne et al., 2011; Labbe, Marcotte, and Savard, 1998), but also as *Stackelberg network pricing games* (Briest, Hoefer, and Krysta, 2012; Roch, Savard, and Marcotte, 2005).

Labbe, Marcotte, and Savard (1998) use a bilevel optimization model for taxing a given subset of the edges in a network to maximize the revenue that the leader receives from the followers. Among other results, they prove that the problem is NP-hard for single-commodity instances, exploiting negative edge costs and lower bound restrictions on the taxes. In subsequent work, Roch, Savard, and Marcotte (2005) improve upon this result and show NP-hardness for non-negative edge costs and no lower bound restrictions. They also provide an approximation algorithm for the single-commodity case.

Briest, Hoefer, and Krysta (2012) consider the following Stackelberg setting: There are several buyers who are interested in buying certain (pre-determined) subgraphs of the network and a seller (network owner) who can price a given subset of the edges. Once the seller fixes the prices, the buyers purchase the cheapest subgraph they are interested in. The goal is to maximize the total revenue obtained from the buyers. The authors show that a uniform price for all edges guarantees the seller a revenue within logarithmic factor of the optimal revenue for unweighted followers. A more specific problem was considered by Briest, Chalermsook, et al. (2010), where each buyer i is interested in purchasing a subgraph that contains a shortest s_i, t_i-path. Other special cases were considered by Gamzu and Segev (2010), Grandoni and Rothvoss (2016), and Guruswami et al. (2005).

In general, there is a vast literature on the problem of pricing multiple items so as to maximize the revenue obtained from (possibly budget-constrained) buyers. There is a close connection between our problem and the problem of determining *envy-free* prices (Guruswami et al., 2005) because envy-freeness naturally corresponds to choosing the cheapest available option. Especially, we exploit known hardness results for the special cases of the *unit-demand pricing problem* and the *single-minded pricing problem* (see (Briest, 2008; Chalermsook, Laekhanukit, and Nanongkai, 2013; Guruswami et al., 2005)) to establish the inapproximability results of our (more restrictive) network pricing problem.

Our problem differs from the ones mentioned above because (i) the seller corresponds to a given node u who can set the prices of its outgoing edges only, and (ii) the revenue that u obtains depends on the proportion of the demand of each commodity routed along shortest paths through u.

3.2 Preliminaries

We formally define the *Network Pricing Problems* considered in this chapter. Suppose we are given a directed graph $G = (V, E)$ with non-negative edge costs $(c_e)_{e \in E}$, $k \geq 1$ commodities $(s_i, t_i, w_i)_{i \in [k]}$, a designated node $u \in V$, and a natural number τ. Each commodity $i \in [k]$ is specified by a source-target pair $(s_i, t_i) \in V \times V$ with $s_i \neq t_i$ and a demand (or weight) $w_i > 0$. The interpretation here is that each commodity $i \in [k]$ sends a total of w_i units of flow from the source node s_i to the target node t_i. The demand w_i is split evenly along all (simple[2]) shortest s_i, t_i-paths with respect to the edge costs $(c_e)_{e \in E}$ (formal definitions are given below). We assume that for each commodity $i \in [k]$, $s_i, t_i \neq u$ and there is at least one s_i, t_i-path that passes through u. This assumption is without loss of generality as otherwise the commodity is irrelevant (as will become clear below) and can be removed.

We introduce some more notation. Let n and m be the number of nodes and edges of G, respectively. Recall that by $\pi(x, y)$ we denote the number of shortest x, y-paths with respect to $(c_e)_{e \in E}$. For ease of notation, for every commodity $i \in [k]$, we use $\pi^i = \pi(s_i, t_i)$ to refer to the number of shortest s_i, t_i-paths. Further, we define π^i_u as the number of shortest s_i, t_i-paths that pass through node $u \in V$,

[2] Recall that a path is said to be *simple* if it does not contain any cycles. Throughout the chapter, whenever we refer to a shortest path we implicitly mean a simple shortest path.

where $s_i, t_i \neq u$. Given an outgoing edge $e = (u, v) \in E$ of u, we denote by π_e^i the number of shortest s_i, t_i-paths that pass through e. Observe that $\pi_u^i = \sum_{e \in \delta^+(u)} \pi_e^i$.

We can now define the flow that passes through the outgoing edges of u: Recall that the demand w_i of each commodity $i \in [k]$ is assumed to be split evenly over all shortest s_i, t_i-paths. Formally, the flow f_e^i of an outgoing edge $e = (u, v)$ of commodity i is defined as $f_e^i = w_i \cdot \pi_e^i / \pi^i$. The total flow passing through node u with respect to commodity i is then

$$f_u^i = \sum_{e \in \delta^+(u)} f_e^i = \sum_{e \in \delta^+(u)} w_i \cdot \frac{\pi_e^i}{\pi^i} = w_i \cdot \frac{\pi_u^i}{\pi^i}.$$

Further, we define $f_e = \sum_{i \in [k]} f_e^i$ as the total flow on edge e. The *total flow of node u* is then defined as

$$f_u = \sum_{e \in \delta^+(u)} f_e = \sum_{i \in [k]} \sum_{e \in \delta^+(u)} f_e^i = \sum_{i \in [k]} w_i \cdot \frac{\pi_u^i}{\pi^i} = \sum_{i \in [k]} f_u^i.$$

Another notion that is of interest in this chapter is the following one: The *total revenue of node u* is defined as

$$r_u = \sum_{e \in \delta^+(u)} f_e \cdot c_e = \sum_{i \in [k]} \sum_{e \in \delta^+(u)} f_e^i \cdot c_e = \sum_{i \in [k]} \sum_{e \in \delta^+(u)} w_i \cdot \frac{\pi_e^i}{\pi^i} \cdot c_e.$$

To get intuition we will show some of the above values by means of an example.

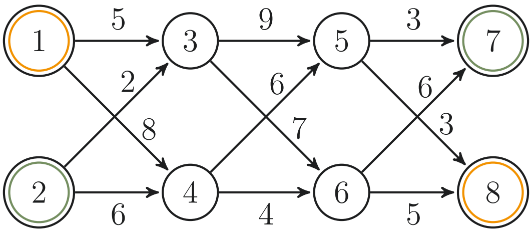


Figure 3.2: Example graph

Example 3.2.1. Consider the graph in Figure 3.2. Suppose we have two commodities $(s_1, t_1, w_1) = (1, 8, 3), (s_2, t_2, w_2) = (2, 7, 2)$.

Then, $\pi^1 = \pi(1, 8) = 4$ as the shortest path has a length of 17 and there are four of them. Two of the four paths go through node 5, so $\pi_5^1 = 2$. Also, $\pi_{\{5,8\}}^1 = 2$ because two paths go through edge $\{5, 8\}$.

As commodity 1 has a weight of 3, we can compute $f_5^1 = 3 \cdot \frac{2}{4} = \frac{3}{2}$.

It can be verified that $f_5^2 = 2 \cdot \frac{1}{1} = 2$. So, the total flow going through node 5 is $f_5 = \frac{3}{2} + 2 = \frac{7}{2}$.

Focusing on an edge, $f_{\{3,5\}}^1 = \frac{3}{4}$ and $f_{\{3,5\}}^2 = 2$. Thus $f_{\{3,5\}} = \frac{11}{8}$.

We can also compute the revenue of node 3. Edge $(3,5)$ has a flow of $\frac{11}{8}$ yielding $9 \cdot \frac{11}{8} = \frac{99}{8}$ while edge $(3,6)$ has a flow of $\frac{3}{4}$ giving $7 \cdot \frac{3}{4} = \frac{21}{4}$. Hence, together this adds up to $r_3 = \frac{99}{8} + \frac{21}{4} = \frac{141}{8}$.

Suppose we can change the costs of $\tau \in [d^+(u)]$ outgoing edges of u; we also say that we can *price* (or *tax*) τ outgoing edges of u. How would we set the edge costs such that the total flow (or revenue) of u is maximized? More precisely, our goal is to determine a set $S \subseteq \delta^+(u)$ with $|S| \leq \tau$ and non-negative costs $\bar{c}_S = (\bar{c}_e)_{e \in S}$ for the edges in S such that f_u (or r_u) with respect to the combined edge costs $(\bar{c}_S, c_{-S})$ is maximized, where we use $c_{-S} = (c_e)_{e \in E \setminus S}$ to refer to the (original) costs of the edges in $E \setminus S$ that remain unchanged. For convenience, we also write $\bar{c}_e = \bar{c}_{\{e\}}$. We also write p_S when we set the cost of all edges in S to $p \in \mathbb{R} \cup \{\infty\}$. We use $f_u(\bar{c}_S)$ and $r_u(\bar{c}_S)$ to refer to the total flow and revenue of u, respectively, with respect to $(\bar{c}_S, c_{-S})$.

3.2.1 Network Pricing Problems

This gives rise to the following two optimization problems:

NETWORK PRICING PROBLEM (NPP)

Given: A directed graph $G = (V, E)$ with non-negative edge costs $(c_e)_{e \in E}$, k commodities $(s_i, t_i, w_i)_{i \in [k]}$, a designated node $u \in V$ and a number $\tau \in [d^+(u)]$.

Goal: Determine a set $S \subseteq \delta^+(u)$ with $|S| \leq \tau$ and edge costs $\bar{c}_S = (\bar{c}_e)_{e \in S}$ such that $f_u(\bar{c}_S)$ is maximized (**FLOW-NPP**), or $r_u(\bar{c}_S)$ is maximized (**REV-NPP**)

Note that if the commodities correspond to all possible node pairs of the graph (not involving u as a source or target node) and $w_i = 1$ for all i, then the flow through u is the betweenness centrality of u scaled by $n(n-1)$. In particular, in this case, **FLOW-NPP** can be interpreted as the problem of maximizing the betweenness centrality of u.

In our discussion below, we distinguish the following three cases:

1. $\tau = 1$: We are allowed to change the cost of only one outgoing edge of u.
2. $1 < \tau < d^+(u)$: We are allowed to change the cost of τ outgoing edges of u.
3. $\tau = d^+(u)$: We are allowed to change the cost of all the outgoing edges of u.

3.2.2 Observations and Assumptions

We continue with some basic observations. A pathological case we want to avoid in **REV-NPP** is that we can charge arbitrarily high costs.

Assumption 3.2.2. *For every commodity $i \in [k]$ there is at least one s_i, t_i-path that does not pass through u.*

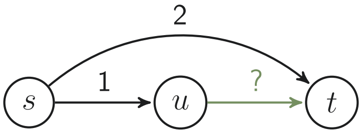


Figure 3.3: Example graph

Throughout the chapter, we assume that the edge costs are non-negative integers (as they may correspond to monetary units, percentages of a fixed precision, etc.).[3] The following example shows that this assumption is needed if one wants to be able to determine edge costs that realize the optimal revenue. Consider the instance depicted in Figure 3.3 and assume that there is a unit demand to be sent from s to t. Suppose we can impose an arbitrary non-negative rational cost $\bar{c}_e \in \mathbb{Q}_{\geq 0}$ on the edge $e = (u, t)$. If we set $\bar{c}_e = 1$, then the revenue of u becomes $\frac{1}{2}$. Otherwise, if we set $\bar{c}_e = 1 - \varepsilon$ for a small rational $\varepsilon > 0$, then the revenue of u is $1 - \varepsilon$. It follows that **REV-NPP** does not admit an optimal solution.

Finally, we need to be able to efficiently compute how the flow splits. If there are zero cost cycles this may become infeasible because the problem of counting the number of simple s, t-paths in a directed graph is $\#P$-complete (Valiant, 1979). We thus make the following assumption:

Assumption 3.2.3. *The edge costs $(c_e)_{e \in E}$ are non-negative integers and the graph does not contain any zero cost cycles, even if all outgoing edges of u are set to zero.*

Making Assumption 3.2.3, it is not hard to see that we can compute all relevant flows (as defined above) in polynomial time by running an adapted version of Dijkstra's shortest path algorithm (Dijkstra, 1959) for each commodity $i \in [k]$: In addition to the distance label that Dijkstra's algorithm maintains for each node $v \in V$, we also keep track of the number of shortest paths that can reach v. Throughout the chapter we use this fact without stating it explicitly.

3.3 Flow Maximization Problem

In this section, we consider the problem **FLOW-NPP**. We settle the three cases **(C1)**–**(C3)** completely for this problem.

[3] All our results continue to hold if the edge costs are of the form $p \cdot \mathbb{Z}_{\geq 0}$ for some real number $p > 0$. In particular, this covers most practically relevant scenarios where one is bound to a finite number of decimals.

3.3.1 Changing the Costs of Few or Almost All Edges

We first prove the following intuitive monotonicity property for the flow f_u through u: If the cost of a single outgoing edge of u decreases, then the flow through u does not decrease. This property will turn out to be crucial below.

Lemma 3.3.1. *Consider an edge $e = (u, v) \in \delta^+(u)$ and assume that the edge cost c_e is decreased to $\bar{c}_e < c_e$. Then $f_u(\bar{c}_e) \geq f_u(c_e)$.*

Proof: Let $c = (c_e)_{e \in E}$ and $\bar{c} = (\bar{c}_e, c_{-e})$ be the original and modified edge costs, respectively. Fix a commodity $i \in [k]$ and consider the flow $f_u^i(c_e) = f_u^i$ passing through u with respect to c. We have

$$f_u^i(c_e) = \frac{w_i \cdot \pi_u^i}{\pi^i} = \frac{w_i \cdot \pi_u^i}{\pi_{\bar{u}}^i + \pi_u^i}, \tag{3.1}$$

where $\pi_{\bar{u}}^i$ denotes the number of shortest s_i, t_i-paths *not* passing through node u. Note that if the cost of e decreases, then for commodity i the cost of the shortest s_i, t_i-path either decreases or stays the same. In the former case, all shortest s_i, t_i-paths must pass through edge e and therefore through node u. This implies that $f_u^i(\bar{c}_e) = w_i \geq f_u^i(c_e)$.

In the latter case, there are two possible scenarios. (i) There is no shortest s_i, t_i-path passing through edge e with respect to $\bar{c}$. Then there is no shortest path passing through e with respect to c either, because for every path P containing e it holds that $c(P) \geq \bar{c}(P)$. Thus, $f_u^i(\bar{c}_e) = f_u^i(c_e)$. (ii) There is at least one shortest s_i, t_i-path passing through edge e with respect to $\bar{c}$. Then each shortest s_i, t_i-path not passing through u remains a shortest path (recall that we assume that the cost of the shortest s_i, t_i-path remains the same). Thus, $\pi_{\bar{u}}^i$ does not change. On the other hand, π_u^i (with respect to $\bar{c}$) increases. Using (3.1) together with the fact that $x/y \leq (x + z)/(y + z)$ for any $x, z \geq 0$, $y > 0$ and $x \leq y$, it follows that $f_u^i(\bar{c}_e) > f_u^i(c_e)$.

We conclude that for each $i \in [k]$ it holds that $f_u^i(\bar{c}_e) \geq f_u^i(c_e)$. By summing over all commodities, we obtain

$$f_u(\bar{c}_e) = \sum_{i \in [k]} f_u^i(\bar{c}_e) \geq \sum_{i \in [k]} f_u^i(c_e) = f_u(c_e).$$

□

Using Lemma 3.3.1, it is clear what we should do if we can price a subset $S \subseteq \delta^+(u)$ of edges: Simply set the cost of each edge $e \in S$ to zero to maximize the flow through u. We summarize this observation in the following corollary.

Corollary 3.3.2. *Suppose we can change the costs of the edges in $S \subseteq \delta^+(u)$. Then setting $\bar{c}_e = 0$ for every $e \in S$ maximizes the flow f_u of u.*

Proof: The proof follows by induction on the size of S and Lemma 3.3.1. □

Note that this takes away the difficulty of determining optimal costs for the edges in S. What remains is how to find the right subset of edges S to be priced. Exploiting this insight, we can prove the following theorem:

Theorem 3.3.3. FLOW-NPP *can be solved optimally in polynomial time if* $\tau = \mathcal{O}(1)$ *or* $\tau = d^+(u) - \mathcal{O}(1)$.

Proof: Let $S \subseteq \delta^+(u)$ be an arbitrary subset of τ outgoing edges of u. By Corollary 3.3.2, we maximize the flow f_u through u by setting $\bar{c}_e = 0$ for every edge $e \in S$. By iterating over all possible subsets of size τ and choosing the one that maximizes $f_u(\bar{c}_S)$, we obtain an optimal solution.

The time needed to compute $f_u(\bar{c}_S)$ for a fixed set $S \subseteq \delta^+(u)$ is at most $\mathcal{O}(mn + n^2 \log n)$ because we have to run Dijkstra's algorithm at most n times. Assume that $\tau = \gamma$ or $\tau = d^+(u) - \gamma$ for some constant γ. The number of such sets is then

$$\binom{d^+(u)}{\tau} < d^+(u)^\gamma \leq n^\gamma.$$

The overall running time is thus polynomial in n and m (γ is a constant). □

3.3.2 Changing the Costs of τ Edges

We consider the cases of **(C2)** which are not captured by Theorem 3.3.3. Then the above approach of enumerating all possible subsets fails to provide an efficient algorithm. In fact, below we show that **FLOW-NPP** is NP-hard to approximate within a factor $1 - 1/e$, even in the unit demand setting (i.e., $w_i = 1$ for all $i \in [k]$).

Inapproximability

Theorem 3.3.4. *Assuming* P $\neq$ NP*, there is no* α*-approximation algorithm with* $\alpha > 1 - 1/e$ *for* **FLOW-NPP** *with* $\mathcal{O}(1) < \tau < d^+(u) - \mathcal{O}(1)$*, even in the unit demand setting.*

We recall the definition of the *Maximum Set Coverage Problem*, which will be used in our reductions.

Problem 3.3.5. (Maximum Set Coverage Problem (**MSC**)). *Given a finite set of elements* $\mathcal{U} = \{e_1, \ldots, e_\nu\}$*, a collection of subsets* $\mathcal{S} = \{S_1, \ldots, S_\mu\}$ *with* $S_j \subseteq \mathcal{U}$ *for every* $j \in [\mu]$*, and an integer* l*, find a subcollection of sets* $\mathcal{S}' \subseteq \mathcal{S}$ *with* $|\mathcal{S}'| \leq l$ *such that the total number of covered elements* $\mathsf{cov}(\mathcal{S}') := |\cup_{S_j \in \mathcal{S}'} S_j|$ *is maximized.*

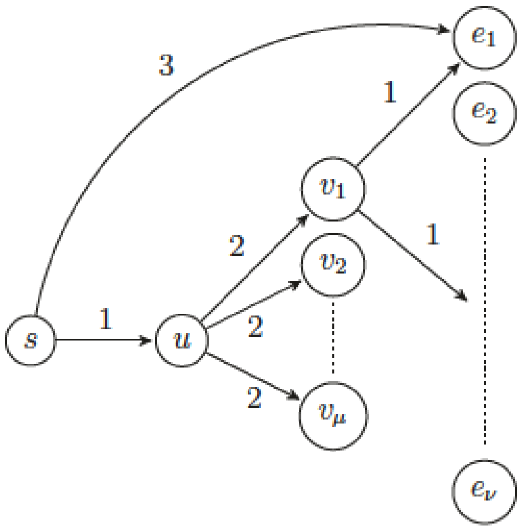


Figure 3.4: Illustration of instance used in the reduction

Proof: Below we derive an L-reduction from **MSC** to **FLOW-NPP** with $a = b = 1$. As a consequence, an α-approximation algorithm for **FLOW-NPP** provides an α-approximation algorithm for **MSC**. Given that it is NP-hard to approximate **MSC** by a factor better than $1 - 1/e$ (Feige, 1998), we conclude that there is no α-approximation algorithm for **FLOW-NPP** with $\alpha > 1 - 1/e$, unless P = NP.

Let $I = (\mathcal{U}, \mathcal{S}, l)$ be an instance of **MSC**. We construct an instance $I' = (G, (c_e)_{e \in E}, (s_i, t_i, w_i)_{i \in [k]}, u, \tau)$ of **FLOW-NPP** as follows: Let the set of vertices of G be $V = \{s, u, v_1, \ldots, v_\mu\} \cup \{e_1, \ldots, e_\nu\}$, where each vertex v_j, $j \in [\mu]$, corresponds to a set $S_j \in \mathcal{S}$ and each vertex e_i, $i \in [\nu]$, corresponds to the respective element in $\mathcal{U}$. The set of edges E and the respective edge costs $(c_e)_{e \in E}$ are defined as follows (see Figure 3.4 for an illustration):

There is an edge (s, u) of cost 1.

For every v_j, $j \in [\mu]$, there is an edge (u, v_j) of cost 2.

For every $e_i \in \mathcal{U}$ and $j \in [\mu]$ such that $e_i \in S_j$, there is an edge (v_j, e_i) of cost 1.

For every $e_i \in \mathcal{U}$, there is an edge (s, e_i) of cost 3.

Finally, we let $\tau = l$ and define a commodity (s, e_i, w_i) with demand $w_i = 1$ for every $i = 1, \ldots, \nu$; in particular, there are $k = \nu$ commodities. Clearly, this reduction can be done in time $\mathcal{O}(\mu\nu)$ and thus Property (i) of Definition 2.3.16 is satisfied.

Next, we show that the optimal solution values for instances I and I' are the same; in particular, this proves Property (ii) of Definition 2.3.16 for $a = 1$. Let $\bar{c}_D$ with $D \subseteq \delta^+(u)$, $|D| \leq \tau$, be an optimal solution for instance I' of **FLOW-NPP**.

Recall that by Corollary 3.3.2 we can assume without loss of generality that $\bar{c}_e = 0$ for all $e \in D$. Note that by construction for every commodity $i \in [k]$ the demand of $w_i = 1$ passes through u if and only if there exists an edge $(u, v_j) \in D$ with $e_i \in S_j$. Define a subcollection $\mathcal{S}'$ for instance I of MSC as $\mathcal{S}' = \{S_j \in \mathcal{S} \,:\, (u, v_j) \in D\}$. Note that $|\mathcal{S}'| \leq \tau = l$ and thus $\mathcal{S}'$ is a feasible solution for I. Further, by the above, the total flow $f_u(\bar{c}_D)$ through u is equal to the number of elements $\mathsf{cov}(\mathcal{S}')$ covered by $\mathcal{S}'$. We conclude that

$$\mathrm{OPT}(I') = f_u(\bar{c}_D) = \mathsf{cov}(\mathcal{S}') \leq \mathrm{OPT}(I). \tag{3.2}$$

Now, let $\mathcal{S}'$, $|\mathcal{S}'| \leq l$, be an optimal solution for instance I of MSC. We define a solution $\bar{c}_D$ for instance I' of $\mathsf{FLOW\text{-}NPP}$ as $\bar{c}_D = 0_D$ with $D = \{(u, v_j) \in E \,:\, S_j \in \mathcal{S}'\}$. Then $\bar{c}_D$ is feasible for I'. By following the same arguments as above, it is not hard to see that

$$\mathrm{OPT}(I) = \mathsf{cov}(\mathcal{S}') = f_u(\bar{c}_D) \leq \mathrm{OPT}(I'). \tag{3.3}$$

Combining (3.2) and (3.3) shows that $\mathrm{OPT}(I) = \mathrm{OPT}(I')$ as claimed.

It remains to show Property (iii) of Definition 2.3.16. Consider an arbitrary solution $\bar{c}_D$ with $D \subseteq \delta^+(u)$, $|D| \leq \tau$, for instance I' of $\mathsf{FLOW\text{-}NPP}$. Let the total flow through u be $Z' = f_u(\bar{c}_D)$. Note that the flow through u does not change if we remove from D all edges e with $\bar{c}_e \geq 2$. We can therefore assume without loss of generality that $\bar{c}_e \in \{0, 1\}$ for all $e \in D$. Define $\mathcal{S}' = \{S_j \in \mathcal{S} \,:\, (u, v_j) \in D\}$ and let $Z = \mathsf{cov}(\mathcal{S}')$. Clearly, $\mathcal{S}'$ is a feasible solution for instance I of MSC and can be derived in polynomial time. Note that Property (iii) holds with $b = 1$ if we can show that $Z \geq Z'$ because $\mathrm{OPT}(I) = \mathrm{OPT}(I')$ (as shown above).

Fix some e_i with $i \in [k]$. First, we consider the contribution of commodity i to Z'. Define $R_i = \{(u, v_j) \in E \,:\, e_i \in S_j\}$ as the set of *relevant edges* of e_i. We distinguish three cases: Case 1: none of the edges $(u, v_j) \in R_i$ belongs to D. Then i contributes zero to Z' (because the shortest s, e_i-path has cost 3 and does not pass through u). Case 2: $\bar{c}_e = 0$ for some edge $e = (u, v_j) \in R_i \cap D$. Then i contributes the entire demand $w_i = 1$ to Z' (because all shortest s, e_i-paths have cost 2 and pass through u). Case 3: $\bar{c}_e = 1$ for all edges $e = (u, v_j) \in R_i \cap D$. Then i contributes at least half of the demand $w_i = 1$ to Z' (because there is at least one shortest s, e_i-path of cost 3 passing through u).

Next, consider the contribution of element e_i to the value Z of the MSC instance I. In Case 1, e_i contributes 0 to Z. On the other hand, e_i contributes 1 to Z in Cases 2 and 3 (because e_i is covered by $\mathcal{S}'$). We conclude that $Z \geq Z'$. □

Greedy Algorithm

Next, we derive a $(1 - 1/e)$-approximation algorithm for $\mathsf{FLOW\text{-}NPP}$, which is best possible by Theorem 3.3.4. Our algorithm exploits a well-known result

Algorithm 3.1: Greedy algorithm for **FLOW-NPP**

```
$S = \emptyset$
for $i = 1, \ldots, \tau$ do
    Let $e_{\max} \in \arg\max\{f_u(0_{S \cup \{e\}}) \,:\, e \in \delta^+(u) \setminus S\}$
    $S = S \cup \{e_{\max}\}$
return $S$
```

by Nemhauser, Wolsey, and Fisher (1978) for the maximization of submodular functions.

The *Submodular Function Maximization Problem* is defined as follows: Given a finite set N, a non-negative, monotone, and submodular function $z : 2^N \to \mathbb{R}$ and an integer l, find a set $S \subseteq N$ such that $|S| \leq l$ and $z(S)$ is maximum. Nemhauser, Wolsey, and Fisher (1978) proved that the following natural greedy algorithm exhibits an approximation ratio of $1 - 1/e$ for this problem: Start with the empty set $S = \emptyset$ and repeatedly add an element of maximal *marginal gain* to S, i.e., an element $j \in N \setminus S$ that maximizes $z(S \cup \{j\}) - z(S)$, until S contains l elements.

Applied to our setting, the greedy algorithm iteratively selects τ edges, where in each iteration it sets to zero the cost of an edge $e = (u, v)$ that gives the largest marginal increase in the flow passing through node u. A more detailed description of the algorithm is given in Algorithm 3.1.

Theorem 3.3.6. *The greedy algorithm provides a $(1 - 1/e)$-approximation for* **FLOW-NPP**.

The proof of Theorem 3.3.6 follows directly from Lemma 3.3.7 below, which shows that $f_u(0_S)$ (if considered as a set function) is non-negative, monotone, and submodular, and using the result of Nemhauser et al. (Nemhauser, Wolsey, and Fisher, 1978).

Lemma 3.3.7. *Define $z(S) = f_u(0_S)$ for every $S \subseteq \delta^+(u)$. The function z is non-negative, monotone, and submodular.*

Proof: Clearly, $z(S) \geq 0$ by definition. Also, note that for each $S \subset \delta^+(u)$ and for each edge $e \in \delta^+(u) \setminus S$, $z(S \cup \{e\}) = f_u(0_{S \cup \{e\}}) \geq f_u(0_S) = z(S)$, which follows directly from Lemma 3.3.1. Thus, z is monotone.

We now show that z is submodular, i.e., for every $S \subseteq T \subseteq \delta^+(u)$ and for each edge $e \in \delta^+(u)$ such that $e \notin T$, we need to show that

$$\begin{aligned} z(S \cup \{e\}) - z(S) &= f_u(0_{S \cup \{e\}}) - f_u(0_S) \\ &\geq f_u(0_{T \cup \{e\}}) - f_u(0_T) = z(T \cup \{e\}) - z(T). \end{aligned} \tag{3.4}$$

Below we prove the following inequality for each commodity $i \in [k]$:

$$f_u^i(0_{S \cup \{e\}}) - f_u^i(0_S) \geq f_u^i(0_{T \cup \{e\}}) - f_u^i(0_T). \tag{3.5}$$

Inequality (3.4) then follows because $\sum_{i\in[k]} f_u^i = f_u$.

Fix some commodity $i \in [k]$. We denote by $\delta_A(\cdot,\cdot)$ the shortest path distances in G with respect to the edge costs $(0_A, c_{-A})$ (i.e., after setting the costs of the edges in A to zero). Note that for each edge $e = (u,v) \in \delta^+(u)$ we have

$$\delta_S(s_i,u) = \delta_T(s_i,u). \tag{3.6}$$

If $\delta_T(v,t_i) < \delta_S(v,t_i)$ then all shortest paths (in δ_T) from v to t_i pass through u. So in this case $\delta_T(v,t_i) = \delta_T(v,u) + \delta_T(u,t_i)$. But $\delta_T(v,u) > 0$ because after setting c_e to zero, no zero cycles arise (by assumption). So $\delta_T(v,t_i) > \delta_T(u,t_i)$, and setting c_e to zero creates no extra shortest s_i,t_i-paths passing through u, which implies that $f_u(0_{T\cup\{e\}}) = f_u(0_T)$, so (3.5) is satisfied. Therefore, we assume from now on that

$$\delta_S(v,t_i) = \delta_T(v,t_i). \tag{3.7}$$

If $\delta_S(s_i,t_i) > \delta_S(s_i,u) + \delta_S(v,t_i)$, then all shortest s_i,t_i-paths will pass through u after setting c_e to zero. This means that $f_u^i(0_{T\cup\{e\}}) = f_u^i(0_{S\cup\{e\}}) = w_i$. We conclude that (3.5) holds because $f_u^i(0_T) \geq f_u^i(0_S)$ by Lemma 3.3.1.

So we may assume $\delta_S(s_i,t_i) \leq \delta_S(s_i,u) + \delta_S(v,t_i)$, and hence

$$\delta_T(s_i,t_i) \leq \delta_S(s_i,t_i) \leq \delta_S(s_i,u) + \delta_S(v,t_i) = \delta_T(s_i,u) + \delta_T(v,t_i), \tag{3.8}$$

where the inequality follows since $T \supseteq S$, and the last equality follows from (3.6) and (3.7).

If the inequality $\delta_T(s_i,t_i) \leq \delta_T(s_i,u) + \delta_T(v,t_i)$ in (3.8) is strict, then setting $c_e = 0$ after setting the cost of edges in T to zero has no effect on the flow passing through the node, i.e., $f_u^i(0_{T\cup\{e\}}) - f_u^i(0_T) = 0$, and (3.5) is satisfied.

If equality holds throughout in (3.8), then $\delta_S(s_i,t_i) = \delta_S(s_i,u) + \delta_S(v,t_i)$ and $\delta_T(s_i,t_i) = \delta_T(s_i,u) + \delta_T(v,t_i)$, so the number of shortest paths through u increases by π_e^i by setting $c_e = 0$, both after setting the cost of edges in S and T to zero. Therefore

$$f_u^i(0_{S\cup\{e\}}) = \frac{f_u^i(0_S)\cdot\pi^i(S) + w_i\pi_e^i}{\pi_e^i + \pi^i(S)},$$

which gives

$$f_u^i(0_{S\cup\{e\}}) - f_u^i(0_S) = \frac{w_i\pi_e^i - f_u^i(0_S)\cdot\pi_e^i}{\pi_e^i + \pi^i(S)} = \frac{\pi_e^i(w_i - f_u^i(0_S))}{\pi_e^i + \pi^i(S)}. \tag{3.9}$$

Similarly, we obtain

$$f_u^i(0_{T\cup\{e\}}) - f_u^i(0_T) = \frac{\pi_e^i(w_i - f_u^i(0_T))}{\pi_e^i + \pi^i(T)}. \tag{3.10}$$

Now, by noting that $w_i - f_u^i(0_T) \leq w_i - f_u^i(0_S)$ and that $\pi^i(T) \geq \pi^i(S)$ (since $\delta_T(s_i,t_i) = \delta_S(s_i,t_i)$), (3.9) and (3.10) imply that (3.5) holds.

□

3.4 Revenue Maximization Problem

We turn to the problem **REV-NPP**. While case **(C1)** can be solved optimally the problem becomes much more challenging for cases **(C2)** and **(C3)**. In particular, as we show below obtaining constant factor approximation algorithms is infeasible, even if we can price all outgoing edges of u (case **(C3)**).

3.4.1 Changing the Cost of One Edge

We consider the case of **REV-NPP(C1)**, i.e., if we can change the cost of one outgoing edge only. We prove the following theorem.

Theorem 3.4.1. **REV-NPP(C1)** *can be solved optimally in polynomial time.*

We first show that we can efficiently compute the optimal cost for the edge $e = (u, v)$ if the other costs stay fixed; see Lemma 3.4.2 below.

Fix an outgoing edge $e = (u, v) \in \delta^+(u)$ of u. Define the *threshold price* of commodity $i \in [k]$ as

$$\theta_i := \delta_{E\setminus\{e\}}(s_i, t_i) - \delta(s_i, u) - \delta(v, t_i). \tag{3.11}$$

Intuitively, θ_i is the price that we can impose on edge e to equalize the costs of the shortest s_i, t_i-paths passing through e with those that are not passing through e.

The following structural result on the optimal costs is crucial.

Lemma 3.4.2. *Fix an outgoing edge $e = (u, v) \in \delta^+(u)$ of u. We can then determine the cost $\bar{c}_e$ of e maximizing the revenue $r_u(\bar{c}_e)$ of u in polynomial time.*

Proof: Let $\bar{c}_e^*$ be some optimal cost that maximizes $r_u(\bar{c}_e^*)$. We first claim that there exists some optimal cost $\bar{c}_e$ with $\bar{c}_e \in T$, where

$$T = \left(\bigcup_{i\in[k]} \{\theta_i - 1, \theta_i\} \right) \cup \{\infty\}.$$

If $\bar{c}_e^* > \max\{\theta_i : i \in [k]\}$ there is no flow passing through e. We obtain the same by setting $\bar{c}_e = \infty \in T$ and thus $r_u(\bar{c}_e) = r_u(\bar{c}_e^*)$, which is optimal. Suppose now that $\bar{c}_e^* \leq \max\{\theta_i : i \in [k]\}$ and $\bar{c}_e^* \notin T$. Let $L = \{i \in [k] : \theta_i < \bar{c}_e^*\}$ and $U = \{i \in [k] : \theta_i - 1 > \bar{c}_e^*\}$. Note that $L \cup U = [k]$. For the commodities in L there is no flow passing through e, while for the commodities in U the entire flow passes through e. By setting $\bar{c}_e = \min\{\theta_i - 1 : i \in U\}$ the flows do not change while $\bar{c}_e > \bar{c}_e^*$. Because $U \neq \emptyset$ we have $r_u(\bar{c}_e) > r_u(\bar{c}_e^*)$, contradicting the optimality of $\bar{c}_e^*$. Hence there is an optimal cost $\bar{c}_e$ in T.

Determining T takes at most $3k$ shortest path calculations. If all costs are fixed, we can compute the revenue by k shortest path calculations. Exploiting that $|T| \leq 2k + 1$, we can thus simply try all values in T and choose $\bar{c}_e$ as the cost

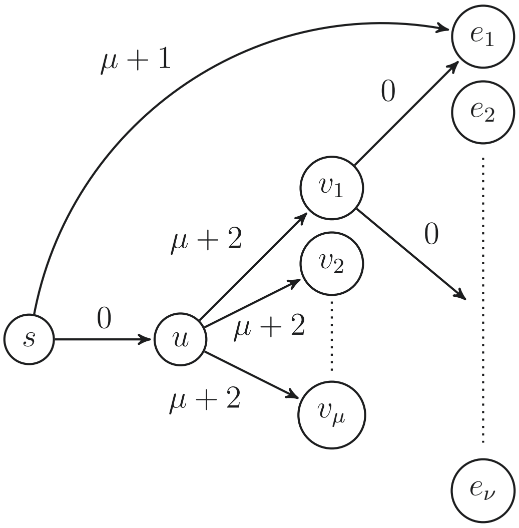


Figure 3.5: Illustration of instance used in the reduction

that gives the largest revenue. □

Proof of Theorem 3.4.1: For every outgoing edge of u, i.e., for all $u \in \delta^+(u)$, we use Lemma 3.4.2 to compute the optimal cost and revenue for that edge in polynomial time. Taking the maximum solves the Theorem. □

3.4.2 Changing the Costs of τ Edges

We turn to case **(C2)** of **REV-NPP**. As we show, this problem is hard to approximate:

Theorem 3.4.3. *Assuming* P $\neq$ NP*, there is no α-approximation algorithm with $\alpha > 1 - 1/e$ for* **REV-NPP(C2)** *with $1 < \tau < d^+(u)$, even in the unit demand setting.*

Proof: Below we derive an L-reduction from **MSC** to **REV-NPP** with $a = \mu$ and $b = \frac{1}{\mu}$. As a consequence, an α-approximation algorithm for **REV-NPP** provides an α-approximation algorithm for **MSC**. Given that it is NP-hard to approximate **MSC** by a factor better than $1 - 1/e$ (Feige, 1998), we conclude that there is no α-approximation algorithm for **REV-NPP** with $\alpha > 1 - 1/e$, unless P = NP.

Let $I = (\mathcal{U}, \mathcal{S}, l)$ be an instance of **MSC**. We construct an instance $I' = (G, (c_e)_{e\in E}, (s_i, t_i, w_i)_{i\in[k]}, u, \tau)$ of **REV-NPP** as follows: Let the set of vertices of G be $V = \{s, u, v_1, \ldots, v_\mu\} \cup \{e_1, \ldots, e_\nu\}$, where each vertex v_j, $j \in [\mu]$, corresponds to a set $S_j \in \mathcal{S}$ and each vertex e_i, $i \in [\nu]$, corresponds to an element in $\mathcal{U}$. The set of edges E and the respective edge costs $(c_e)_{e\in E}$ are defined as follows (see Figure 3.5 for an illustration):

There is an edge (s, u) of cost 0.

For every v_j, $j \in [\mu]$, there is an edge (u, v_j) of cost $\mu + 2$.

For every $e_i \in \mathcal{U}$ and $j \in [\mu]$ such that $e_i \in S_j$, there is an edge (v_j, e_i) of cost 0.

For every $e_i \in \mathcal{U}$, there is an edge (s, e_i) of cost $\mu + 1$.

Finally, we let $\tau = l$ and define a commodity (s, e_i, w_i) with demand $w_i = 1$ for every $i = 1, \ldots, \nu$; in particular, there are $k = \nu$ commodities. Clearly, this reduction can be done in polynomial time $\mathcal{O}(\mu\nu)$, which proves Property (i) of Definition 2.3.16.

Next, we show that the optimal solution values for instances I and I' differ by a factor μ. Let $\bar{c}_D$ with $D \subseteq \delta^+(u)$, $|D| \leq \tau$, be an optimal solution for instance I' of **REV-NPP**.

We first argue that we can assume that $\bar{c}_e \geq \mu$ for all $e \in D$. Suppose that $\bar{c}_e = \mu + 1$ for some $e \in D$. If there is no flow passing through e, i.e., $f_e = 0$, then setting $\bar{c}_e = \mu + 2$ will neither change the flow nor decrease the revenue. If there is some flow passing through e for some commodity $i \in [k]$, i.e., $f_e^i > 0$, then using $\bar{c}_e = \mu$ instead strictly increases the revenue for this commodity. To see this, note that the revenue gained by commodity i through edges of cost $\mu + 1$ is at most $\frac{\tau}{\tau+1}(\mu + 1)$. By setting $\bar{c}_e$ to μ we obtain a revenue of μ which is an increase because $\frac{\tau}{\tau+1}(\mu+1) < \frac{\mu}{\mu+1}(\mu+1) = \mu$. Note that this argument holds for every commodity i independently. We can thus conclude that $\bar{c}_e = \mu$ or $\bar{c}_e \geq \mu + 2$.

Define a subcollection $\mathcal{S}'$ for instance I of **MSC** as

$$\mathcal{S}' = \{S_j \in \mathcal{S} \,:\, (u, v_j) \in D,\ \bar{c}_{(u,v_j)} = \mu\}.$$

Note that $|\mathcal{S}'| \leq \tau = l$ and thus $\mathcal{S}'$ is a feasible solution for I. Further, each element in $\mathsf{cov}(\mathcal{S}')$ corresponds to a commodity i for which the entire demand $w_i = 1$ is sent through u and we thus obtain a revenue of μ. We conclude that the optimal revenue is

$$\mathrm{OPT}(I') = \mu \cdot \mathsf{cov}(\mathcal{S}') \leq \mu \cdot \mathrm{OPT}(I). \tag{3.12}$$

This proves Property (ii) of Definition 2.3.16 for $a = \mu$.

It remains to show Property (iii) of Definition 2.3.16 with $b = \frac{1}{\mu}$. First, let $\mathcal{S}'$, $|\mathcal{S}'| \leq l$, be an optimal solution for I. We define a solution $\bar{c}_D$ for I' as $\bar{c}_D = \mu_D$ with $D = \{(u, v_j) \in E \,:\, S_j \in \mathcal{S}'\}$. Then $\bar{c}_D$ is feasible for I' and it is immediate that

$$\mathrm{OPT}(I) = \mathsf{cov}(\mathcal{S}') \leq \frac{1}{\mu} \cdot \mathrm{OPT}(I'). \tag{3.13}$$

Combining (3.12) and (3.13) proves $\mathrm{OPT}(I') = \mu \cdot \mathrm{OPT}(I)$. Using this equality all we need to show to fulfill Property (iii) is that given a solution with revenue Z' for I' we can convert it in polynomial time into a solution for I with value $Z \geq \frac{Z'}{\mu}$.

Consider an arbitrary solution $\bar{c}_D$ with $D \subseteq \delta^+(u)$, $|D| \leq \tau$, for instance I' of **REV-NPP**. Let the total revenue be Z'. Define $\mathcal{S}' = \{S_j \in \mathcal{S} : (u, v_j) \in D,\ \bar{c}_{(u,v_j)} \leq \mu\}$ and let $Z = \mathsf{cov}(\mathcal{S}')$. The revenue that we obtain from each commodity is at most μ and thus $Z \geq \frac{Z'}{\mu}$, proving Property (iii) for $b = \frac{1}{\mu}$. □

One could hope that a greedy approach similar to the one used for **FLOW-NPP(C2)** would work here as well, but unfortunately, this is not the case. In fact, the objective function is not submodular and the natural greedy algorithm can perform arbitrarily bad.

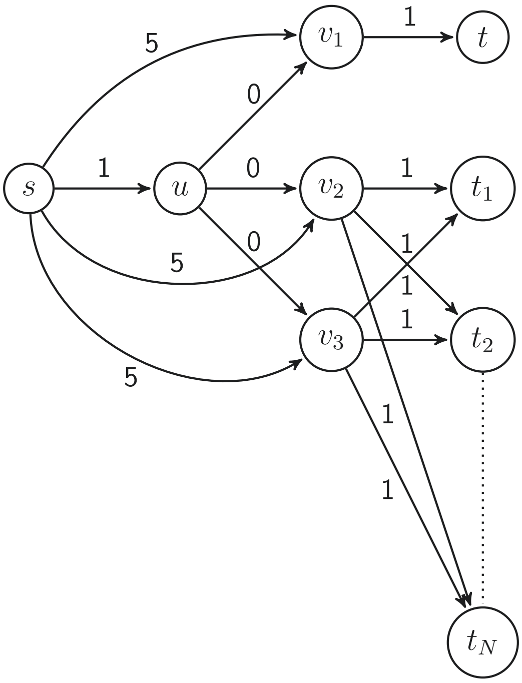
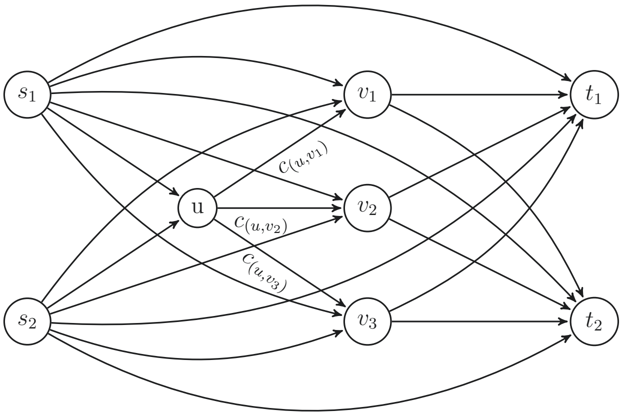


Figure 3.6: Instance showing that the greedy algorithm performs badly

Example 3.4.4. Consider the instance depicted in Figure 3.6 and suppose $\tau = 2$. Furthermore there are $N + 1$ commodities $(s, t, 1)$ and $(s, t_i, 1)$ for $i \in [N]$. The revenue that u receives in the initial configuration is zero. The greedy approach will first set edge $\bar{c}_{(u,v_1)}$ to 2. This leads to a total increase of revenue of 2. After that, the greedy approach will either change $\bar{c}_{(u,v_2)}$ or $\bar{c}_{(u,v_3)}$ yielding no additional revenue, so the total received revenue is 2. But setting both $\bar{c}_{(u,v_2)}$ and $\bar{c}_{(u,v_3)}$ to 2 will give a total revenue of $2N$. Therefore the gap between the optimal solution and the greedy solution can be arbitrarily large.

The above example might suggest that this situation does not occur if the costs of all outgoing edges of u would be ∞ to begin with. However, even then the objective function is *not* submodular as the following theorem shows.



Edge e	c_e	Edge e	c_e
(s_1, u)	1	(s_2, u)	1
(s_1, v_1)	7	(s_2, v_1)	5
(s_1, v_2)	6	(s_2, v_2)	4
(s_1, v_3)	5	(s_2, v_3)	8
(s_1, t_1)	7	(s_2, t_1)	5
(s_1, t_2)	7	(s_2, t_2)	8
(v_1, t_1)	1	(v_1, t_2)	8
(v_2, t_1)	6	(v_2, t_2)	4
(v_3, t_1)	4	(v_2, t_2)	4

Figure 3.7: Counterexample to submodularity

Theorem 3.4.5. *The objective function of* **REV-NPP** *when starting with* $c_{\delta^+(u)} = \infty_{\delta^+(u)}$ *is not submodular.*

Proof: We show that there exists a pair of sets $S \subseteq T \subseteq \delta^+(u)$ and an element $e \in \delta^+(u) \setminus T$ such that $r_u(\bar{c}_{S\cup\{e\}}) - r_u(\bar{c}_S) < r_u(\bar{c}_{T\cup\{e\}}) - r_u(\bar{c}_T)$.

Consider the graph in Figure 3.7. Our goal is to maximize the revenue of node u by changing the cost of the edges in $\delta^+(u)$. Let $S = \{(u, v_3)\}$, $T = \{(u, v_3), (u, v_1)\}$ and $e = (u, v_2)$. For the left hand side of the inequality, we notice that $r_u(\bar{c}_{S\cup\{e\}}) = 13$ which is obtained by setting $\bar{c}_{(u,v_3)} = 3$ and $\bar{c}_{(u,v_2)} = 2$. This is the optimal solution if we are allowed to change the cost of these two edges. $r_u(\bar{c}_S)$, instead, is equal to 8 and it is obtained by setting $\bar{c}_{(u,v_3)} = 2$ which is optimal because if we set it to 3 or to a higher value we lose the flow for commodity (s_2, t_1) (it will go through edge (s_2, t_1) of cost 5).

For the right hand side, by setting $\bar{c}_{(u,v_1)} = 3, \bar{c}_{(u,v_3)} = 3$ and $\bar{c}_{(u,v_2)} = 2$ we have $r_u(\bar{c}_{T\cup\{e\}}) = 23.5$, while for $r_u(\bar{c}_T)$ we obtain 17.5 with $\bar{c}_{(u,v_1)} = 3, \bar{c}_{(u,v_3)} = 2$. Therefore, $13 - 8 < 23.5 - 17.5$ proving that our objective function is not submodular. □

3.4.3 Changing the Costs of All Edges

In this section, we consider the case **(C3)** of **REV-NPP**, where we can change the costs of all outgoing edges of u. We first present our inapproximability results and then turn to our approximation algorithms.

Inapproximability

The main result that we prove in this section is that, under certain computational hardness assumptions, **REV-NPP(C3)** is hard to approximate within a factor

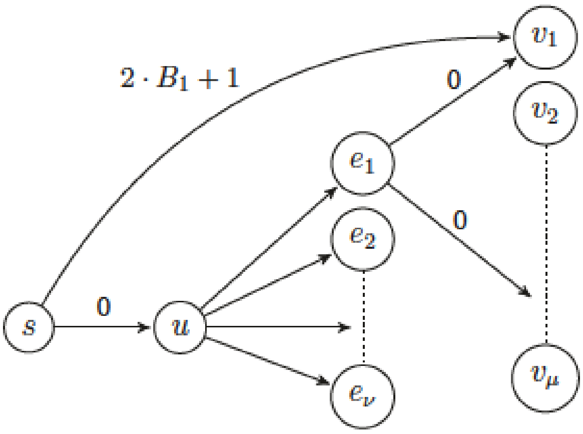


Figure 3.8: Illustration of the instance used in the proof of Theorem 3.4.6

of $\Omega(1/\log^{1-\varepsilon}(k))$ and $\Omega(1/d^+(u)^{1/2-\varepsilon})$, already in the unit demand setting. In particular, note that we obtain two different asymptotic lower bounds on the attainable approximation factors in terms of (i) the number of commodities k, and (ii) the outdegree $d^+(u)$ of u, respectively.

Theorem 3.4.6. *The following hardness results hold for* **REV-NPP(C3)**, *even in the unit demand setting.*

> *Assuming that* NP $\not\subseteq$ *ZPTIME*$(2^{\log^{1/\epsilon}(k)})$, **REV-NPP(C3)** *is* $\Omega(1/\log^{1-\epsilon}(k))$*-inapproximable for any* $\epsilon > 0$.
>
> *Assuming the Exponential Time Hypothesis,* **REV-NPP(C3)** *is* $\Omega\left(1/d^+(u)^{1/2-\epsilon}\right)$*-inapproximable for any* $\epsilon > 0$.

Our reduction is based on the following *Unit-Demand Min-Buying Pricing Problem* ($\mathsf{UDP}_{\min}$) (Chalermsook, Laekhanukit, and Nanongkai, 2013):

Problem 3.4.7. (Unit-Demand Min-Buying Pricing ($\mathsf{UDP}_{\min}$)). *We are given a set of ν items $\mathcal{I} = \{e_1, \ldots, e_\nu\}$ and a set of μ buyers $\mathcal{B} = \{b_1, \ldots, b_\mu\}$. Every buyer $b_i \in \mathcal{B}$ has some budget $B_i \in \mathbb{Z}_{\geq 0}$ and a set $S_i \subseteq \mathcal{I}$ of items she is interested in. Given prices $p : \mathcal{I} \to \mathbb{Z}_{\geq 0}$ for the items, buyer b_i will buy an item $e \in S_i$ whose price $p(e)$ is minimum, but only if $p(e) \leq B_i$. from* $\arg\min_{e \in S_i} p(e)$ *but only if* $\min_{e \in S_i} p(e) \leq B_i$. *The goal is to find prices that maximize the total revenue, i.e.,*

$$\sum_{b_i \in \mathcal{B}} \min\{p(e) \,:\, e \in S_i \;\; \text{and} \;\; p(e) \leq B_i\},$$

where we define the minimum of an empty set to be zero.

Proof of Theorem 3.4.6: We describe a polynomial-time reduction that transforms a given instance I of $\mathsf{UDP}_{\min}$ to an instance I' of **REV-NPP** such that every solution of I' can be converted to a solution of I, thereby losing at most a factor 2 in objective value. As a consequence, the existence of an α-approximation algorithm for **REV-NPP** implies a $\frac{1}{2}\alpha$-approximation algorithm for $\mathsf{UDP}_{\min}$. Chalermsook, Laekhanukit, and Nanongkai (2013) showed that $\mathsf{UDP}_{\min}$ is $\Omega(1/\log^{1-\varepsilon}(\mu))$-inapproximable and $\Omega(1/\nu^{1/2-\varepsilon})$-inapproximable under the respective hardness assumptions stated above. As in our reduction, we have $k = \mu$ and $d^+(u) = \nu$, the stated inapproximability results for **REV-NPP** follow.

Let $I = (\mathcal{I}, \mathcal{B}, (S_i)_{b_i \in \mathcal{B}}, (B_i)_{b_i \in \mathcal{B}})$ be an instance of $\mathsf{UDP}_{\min}$. We construct an instance $I' = (G, (c_e)_{e \in E}, (s_i, t_i, w_i)_{i \in [k]}, u, \tau)$ of **REV-NPP** as follows: Let $k = \mu$, $d^+(u) = \nu$ and let the set of vertices of G be $V = \{s, u, v_1, \ldots, v_\mu\} \cup \{e_1, \ldots, e_\nu\}$, where each vertex v_j, $j \in [\mu]$, corresponds to the set S_j, $j \in [k]$ in I and $e_j \in \mathcal{I}$ corresponds to its counterpart in I. The set of edges E and the respective edge costs $(c_e)_{e \in E}$ are defined as follows (see Figure 3.8 for an illustration):

There is an edge (s, u) of cost 0.

For every $e_i \in \mathcal{I}$, there is an edge (u, e_i) that needs to be priced.

For every $e_i \in \mathcal{I}$ and $j \in [k]$ such that $e_i \in S_j$, there is an edge (e_i, v_j) of cost 0.

For every S_j, $j \in [k]$, there is an edge (s, v_j) of cost $2 \cdot B_j + 1$.

Finally, we have k commodities (s, v_j, w_j) with demand $w_j = 1$ for every $j \in [k]$. Clearly, this reduction can be done in polynomial time.

First note that $\mathrm{OPT}(I') \geq 2\mathrm{OPT}(I)$ since taking the optimal prices p in I and using the prices $\bar{c}_{(u,e_i)} = 2p(e_i)$ for all $i \in [N]$ in I' will give a revenue of $2\mathrm{OPT}(I)$ for I'. If buyer b_i bought item e_j for price $p(e_j)$ in I, then the flow of commodity j will go through (u, e_j), or is split over edges all with cost $2p(e_j)$. in I' contributing $2p(e_j)$ to the revenue.

Consider an optimal solution $\bar{c}$ for I' with revenue Z'. We will convert this into a solution p for I with revenue Z such that $Z \geq Z'/4$. Note that we may assume that $B_j \geq 1$ for all $j \in [k]$: if $B_j = 0$ the buyer cannot contribute anything to the revenue so we can ignore them. Then it holds that $Z' \geq \sum_{i \in [k]} 2 \min_{j \in [k]}\{B_j\} \geq 2k$, which is the revenue we would get by setting all prices to $2\min_{j \in [k]}\{B_j\}$. Now, let $\bar{c}'$ be a modified $\bar{c}$ given by

$$\bar{c}'_e = \begin{cases} \bar{c}_e - 1 & \text{if } e \in \delta^+(u) \text{ and } \bar{c}_e \text{ is odd,} \\ \bar{c}_e & \text{otherwise.} \end{cases}$$

By this modification, we lose at most k revenue. Thus we have $Z' - k \geq Z'/2$ revenue remaining. Observe that all prices are even and thus $f^i_u \in \{0, 1\}$ for all $i \in [k]$. Using prices $p(e_i) = \bar{c}_{(u,e_i)}/2$ for $i \in [\nu]$ in I yields a revenue Z of at least $Z'/4$.

To conclude, if $Z' \geq \alpha \cdot \mathrm{OPT}(I')$ then $4Z \geq Z' \geq \alpha \cdot \mathrm{OPT}(I') \geq 2\alpha \cdot \mathrm{OPT}(I)$ implying $Z \geq \alpha/2 \cdot \mathrm{OPT}(I)$ which proves the theorem. □

Approximation algorithms

We next present our approximation algorithms. We first consider some special cases and turn to the general case of **REV-NPP(C3)** at the end of this section.

Special case: single commodity. We consider the problem of **REV-NPP(C3)** for a single commodity only, i.e., $k = 1$. In this case, we can assume without loss of generality that $w_1 = 1$. Our goal is thus to determine $\bar{c}_{\delta^+(u)} = (\bar{c}_e)_{e\in\delta^+(u)}$ to maximize the revenue

$$r_u(\bar{c}_{\delta^+(u)}) = \sum_{e\in\delta^+(u)} f_e^1 \cdot \bar{c}_e = w_1 \sum_{e\in\delta^+(u)} \frac{\pi_e^1}{\pi_1} \cdot \bar{c}_e = \sum_{e\in\delta^+(u)} \frac{\pi_e^1}{\pi_1} \cdot \bar{c}_e.$$

For every edge $e = (u, v) \in \delta^+(u)$, define the value

$$h(v) := \delta_{E\setminus\delta^+(u)}(s_1, t_1) - \delta(s_1, u) - \delta(v, t_1).$$

Intuitively, $h(v)$ is the largest uniform price that we can impose on all outgoing edges of u such that the edge $e = (u, v)$ is still part of a shortest s_1, t_1-path. We define θ as the maximum over all these uniform prices:

$$\theta = \max_{(u,v)\in\delta^+(u)} h(v). \tag{3.14}$$

The next theorem basically proves that imposing a uniform price of either $\theta - 1$ or θ is optimal.

Theorem 3.4.8. REV-NPP(C3) *with a single commodity only (i.e., $k = 1$) can be solved optimally in polynomial time.*

Proof: Let θ be as defined in (3.14). If $\theta \leq 0$, then no revenue can be obtained and we stop. If $\theta > 0$, we compute the revenue obtained by setting all costs uniformly to either $\theta - 1$ or θ:

$$\begin{aligned} r_u((\theta-1)_{\delta^+(u)}) &= \theta - 1 \\ r_u(\theta_{\delta^+(u)}) &= \theta \cdot \tfrac{1}{\pi^1} \cdot \sum_{\substack{e=(u,v)\in\delta^+(u):\\ h(v)=\theta}} \pi_e^1. \end{aligned}$$

We argue that the maximum of the two is the optimal revenue.

Let $\bar{c}$ be a cost achieving optimal revenue. When $\bar{c}_e > \theta$ for some $e \in \delta^+(u)$ it holds that $f_e = 0$, so we can assume that there is an optimum where $\bar{c}_e \leq \theta$

Algorithm 3.2: Uniform Price Algorithm

```
$C = \emptyset$
foreach commodity $i \in [k]$ do
    Compute the uniform price $\theta_i$ as defined in (3.14) when considering commodity $i$ only.
    Let $\theta_i^* \in \{\theta_i - 1, \theta_i\}$ be the price that achieves higher revenue for commodity $i$ when all edges are priced uniformly.
    $C = C \cup \{\theta_i^*\}$
return $\arg\max\{r_u(p_{\delta^+(u)}) \ : \ p \in C\}$
```

for all $e \in \delta^+(u)$. If there is an optimum for which $\bar{c}_e \leq \theta - 1$ for all $e \in \delta^+(u)$ then the maximum revenue we can get is $\theta - 1$, which is actually attained by setting all $\bar{c}_e$ to $\theta - 1$. Suppose the optimum is larger than $\theta - 1$. Then there must be some $e' \in \delta^+(u)$ with $\bar{c}_{e'} = \theta$ and strictly positive flow. If there is some other edge for which $\bar{c}_e < \theta$ that gets flow then this contradicts e' getting flow. Thus e cannot have flow in which case we could also set $\bar{c}_e = \theta$. Thus there is an optimum solution where $\bar{c}_e = \theta$ for all $e \in \delta^+(u)$. So, the maximum of $r_u(\theta_{\delta^+(u)})$ and $r_u((\theta-1)_{\delta^+(u)})$ achieves the optimal revenue as claimed.

Finally, observe that the values of $h(v)$ and $r_u(\theta_{\delta^+(u)})$ and $r_u((\theta-1)_{\delta^+(u)})$ can all be computed in polynomial time. □

Uniform demands. We exploit the fact that for a single commodity we are able to find optimal uniform costs in polynomial time to derive an approximation algorithm for **REV-NPP(C3)** in the uniform demand setting (i.e., if all w_i are the same).

If there is more than one commodity we can run the procedure described in the proof of Theorem 3.4.8 for every commodity $i \in [k]$ separately (as if it were the only commodity in the network) to obtain a set of possible prices. We prove below that among these prices there is one uniform price that guarantees a good revenue. The resulting *uniform price algorithm* is summarized in Algorithm 3.2.

Theorem 3.4.9. *Algorithm 3.2 is a $1/H_k$-approximation algorithm for* **REV-NPP(C3)** *when all demands are uniform.*

Proof: We can assume without loss of generality that all demands are 1. Let θ_i^* be the optimal price for commodity $i \in [k]$ as determined in the proof of Theorem 3.4.8 and let f_u^i be the flow of commodity i going through u when using prices $\bar{c}_{\delta^+(u)} = (\theta_i^*)_{\delta^+(u)}$.

Assume that the commodities are ordered such that $\theta_1^* \geq \theta_2^* \geq \ldots \geq \theta_k^*$ and if $i < j$ and $\theta_i^* = \theta_j^*$ then $f_u^i \geq f_u^j$. So, first we order on θ_i^* and if the θ_i^* are equal then we order on f_u^i. Let σ be the number of unique values among the θ_i^*. Let $i_1 = 1$ and define i_j for $2 \leq j \leq \sigma$ recursively as the index of the first entry that is strictly smaller than $\theta_{i_{j-1}}^*$. For convenience let $i_{\sigma+1} = k+1$.

Let p be the uniform price output by Algorithm 3.2. The algorithm tries prices θ_i^* and because we have unit demands and by the ordering of the commodities we know that for $i \in \{i_j, \ldots, i_{j+1}-1\}$, it holds that

$$p \geq \theta_i^* \cdot \left((i_j - 1) + \sum_{\ell=i_j}^{i_{j+1}-1} f_u^\ell \right), \quad \text{which implies} \quad \theta_i^* \leq \frac{p}{(i_j-1) + \sum_{\ell=i_j}^{i_{j+1}-1} f_u^\ell}. \tag{3.15}$$

Let OPT be the maximum attainable revenue. If we single out the income from one commodity we cannot expect to do better than when we just consider that commodity. Hence,

$$\begin{aligned} \text{OPT} &\leq \sum_{i=1}^{k} \theta_i^* f_u^i = \sum_{j=1}^{\sigma} \sum_{i=i_j}^{i_{j+1}-1} \theta_i^* f_u^i \leq \sum_{j=1}^{\sigma} \sum_{i=i_j}^{i_{j+1}-1} \frac{p \cdot f_u^i}{(i_j-1) + \sum_{\ell=i_j}^{i_{j+1}-1} f_u^\ell} \qquad (3.16) \\ &\leq \sum_{j=1}^{\sigma} \sum_{i=i_j}^{i_{j+1}-1} \frac{p}{(i_j-1) + (i - (i_j - 1))} = \sum_{j=1}^{\sigma} \sum_{i=i_j}^{i_{j+1}-1} \frac{p}{i} = \sum_{i=1}^{k} \frac{p}{i} = H_k \cdot p. \end{aligned}$$

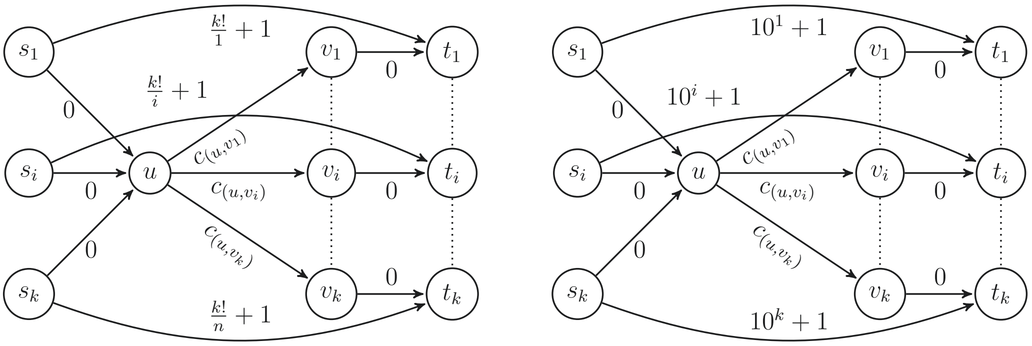
The second inequality follows from (3.15). For the third inequality, we make use of the fact that $f_u^{i*} \leq 1$ and that we sorted the commodities in such a way that if $i < j$ and $\theta_i^* = \theta_j^*$ we have $f_u^{i*} \geq f_u^{j*}$. Thus there are at least $i - (i_j - 1)$ terms for which the θ^*-value is equal but the f-value is at least as large. □

To verify that we could not have done a better job in the analysis of Theorem 3.4.9.

Theorem 3.4.10. *The analysis of the approximation ratio in Theorem 3.4.9 is tight.*

Proof: We give an instance that shows that the analysis of Theorem 3.4.9 is tight. Consider the graph G in Figure 3.9(a). There are k commodities $(s_i, t_i, 1)$ for $i \in [k]$.

For every commodity $(s_i, t_i, 1)$, $i \in [k]$, there is one path through u and one alternative path. The latter having a cost of $\frac{k!}{i} + 1$. The optimal costs are $\bar{c}_{(u,v_i)} = \frac{k!}{i}$, then all flow goes through u for the highest possible revenue $r_u = \sum_{i \in [k]} \frac{k!}{i} = k! \cdot H_k$. For uniform costs we clearly want to choose a cost among $\{\frac{k!}{i} : i \in [k]\}$ since any lower value can be raised to a value inside this set to get a strict improvement. We notice that, for every cost in this set, the revenue is $i \cdot \frac{k!}{i} = k!$. Therefore, the approximation ratio of this example is $\frac{k! \cdot H_k}{k!} = H_k$, and it is tight. □



(a) Example with $w_i = 1$ (b) Example with $w_i = 10^{k-i}$

Figure 3.9: Examples showing that the approximation ratios in Theorem 3.4.9 and Theorem 3.4.11 are tight

General demands. Finally, we turn to **REV-NPP(C3)** with general demands. As it turns out, a simple adaptation of the uniform price algorithm (Algorithm 3.2) achieves approximation ratios that are (asymptotically) best possible: We modify Algorithm 3.2 by replacing line 5 with $C = C \cup \{\theta_i - 1, \theta_i\}$.

Theorem 3.4.11. *The modified version of Algorithm 3.2 is a* $\max\{1/k, 1/d^+(u)\}$*-approximation algorithm for* **REV-NPP(C3)**.

Proof: We first prove that the modified algorithm is a $1/k$-approximation algorithm for **REV-NPP(C3)**. Note that it is sufficient to only consider the prices that are used in the original version of Algorithm 3.2. We follow the same reasoning as in Theorem 3.4.9.

Let θ_i^* and f_u^{i*}, $i \in [k]$, be as in the proof of Theorem 3.4.9. We note that $p \geq \theta_i^* \sum_{\ell=1}^{i} f_u^{\ell*}$, which implies that $\theta_i^* \leq p / \sum_{\ell=1}^{i} f_u^{\ell*}$. Thus,

$$\mathrm{OPT} \leq \sum_{i=1}^{k} \theta_i^* \cdot f_u^{i*} \leq \sum_{i=1}^{k} \frac{p \cdot f_u^{i*}}{\sum_{\ell=1}^{i} f_u^{\ell*}} \leq \sum_{i=1}^{k} p = k \cdot p. \tag{3.17}$$

Next, we show that the modified version of Algorithm 3.2 is also a $1/d^+(u)$-approximation algorithm. Let $\bar{c}^*$ be the optimal prices achieving a revenue of OPT. If we consider the revenue that is contributed by each $e \in \delta^+(u)$, there is at least one $e^* \in \delta^+(u)$ that contributed at least $\mathrm{OPT}/d^+(u)$, i.e., $f_{e^*} \cdot \bar{c}^*_{e^*} \geq \mathrm{OPT}/d^+(u)$. Consider $\bar{c}_e = \bar{c}^*_{e^*}$ for all $e \in \delta^+(u)$. The flow f_{e^*} will not go down because of $e \in \delta^+(u)$ such that $\bar{c}^*_e < \bar{c}^*_{e^*}$. Some of f_{e^*} may go to $e \in \delta^+(u)$ such that $\bar{c}^*_e \geq \bar{c}^*_{e^*}$ but if this happens we will still earn at least $\bar{c}^*_{e^*}$ on it. Hence the revenue for $\bar{c}_e$ is at least $f_{e^*} \cdot \bar{c}^*_{e^*} \geq \mathrm{OPT}/d^+(u)$.

Fix $\bar{c}_e = \bar{c}^*_{e^*}$ for all $e \in \delta^+(u)$. Let $\mathcal{F} = \{i \in [k] : f_u^i > 0\}$, i.e., all commodities that have some positive flow going through u, and so we earn some revenue on them. Let θ_i be the uniform price of commodity i as defined in (3.14). Note that

$\bar{c}_e^* \leq \min\{\theta_i : i \in \mathcal{F}\}$. If $\bar{c}_e^* \geq \min\{\theta_i - 1 : i \in \mathcal{F}\}$ then we are done because then the approximation algorithm will try a price that yields at least $\text{OPT}/d^+(u)$ revenue. Suppose $\bar{c}_e^* < \min\{\theta_i - 1 : i \in \mathcal{F}\}$. Then $f_u^i/w_i = 1$ for all $i \in \mathcal{F}$ and when raising $\bar{c}_e$ to $\min\{\theta_i - 1 : i \in \mathcal{F}\}$ for all $e \in d^+(u)$ the flows for commodities $i \in \mathcal{F}$ will not change while the revenue increases. Hence the approximation algorithm tries a price which yields a revenue of at least $\text{OPT}/d^+(u)$. We conclude that Algorithm 3.2 is a $1/d^+(u)$-approximation algorithm. □

Theorem 3.4.12. *The analysis of the approximation ratio in Theorem 3.4.11 is tight.*

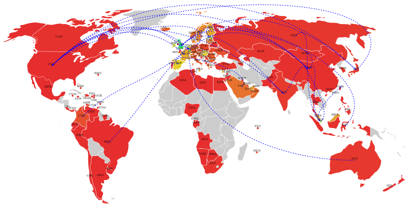
Proof: We give an instance that shows that the analysis of Theorem 3.4.11 is tight. Consider the graph G in Figure 3.9(b). There are k commodities $(s_i, t_i, 1)$ for $i \in [k]$, and $w_i = 10^{k-i}$.

In this case, the optimum is clearly attained by setting $\bar{c}_{(u,v_i)} = 10^i$ for all $i \in [k]$ resulting in $r_u = \sum_{i \in [k]} 10^i \cdot 10^{k-i} = k \cdot 10^k$. If we have a uniform cost, then this will clearly be a value in $\cup_{i \in [k]}\{10^i, 10^i + 1\}$. For a uniform cost of the form 10^i the revenue is $r_u = 10^i \cdot \sum_{j \geq i} 10^{k-j} = \sum_{j \geq i} 10^{k+i-j} = \mathcal{O}(10^k)$ and for a uniform cost of the form 10^i+1 the revenue is $r_u = \frac{1}{2}(10^i+1) \cdot 10^{k-i} + (10^i+1) \cdot \sum_{j>i} 10^{k-j} = \mathcal{O}(10^k)$. Hence the uniform costs are off by a factor of $\mathcal{O}(k)$. □

3.5 Conclusion

A motivating scenario for this research was figuring out how a country should change its tax rates in order to maximize its revenue. Computing the optimum is an intractable problem, but we can use our results to compute an optimal uniform tax. We perform a small experiment on real-world data to test the performance of the uniform price algorithm. We use data from (Riet and Lejour, 2014, 2018), which also provides estimates of the volumes that are sent from one country to another (based on the sizes of their economies). The data contains 108 countries (nodes), 8777 tax treaties (edges), and 11342 commodities. A visualization of the data can be found in Figure 3.10(a). The colored countries are part of the data set, and the colors red, orange, yellow, and green indicate a low to high betweenness centrality. The blue lines are the 20 highest-weight commodities. In this scenario, we need to find "money-transfer" paths such that the total tax paid by the companies is as low as possible.[4] We run our experiments with "The Netherlands" as node u. The results are summarized in Figure 3.10. If the Netherlands would change its outgoing tax rate to 6.7% for all treaties, it would

[4] The tax rates are percentages, while our machinery works with additive shortest paths. To overcome this issue, we use a standard trick and apply $-\log(\cdot)$ on all rates.



(a) World map[5]

	Revenue	
Original	0.0246140	×68
Uniform Pricing	1.6748451	×0.51
Optimum (UB)	3.2994390	
Optimal Uniform Tax	6.70%	

(b) Results

Figure 3.10: Outcome of experiments

potentially increase its revenue by a factor 68. Further, the optimal uniform tax revenue is even within 51% of the optimum (upper bound as in (3.17)) and thus much better than suggested by Theorem 3.4.11.

We settle most cases of **FLOW-NPP** and **REV-NPP** in this chapter, but a case that is not completely settled is **REV-NPP(C2)**. Although we show that it is inapproximable within a factor $1 - 1/e$, case **(C3)** seems to suggest that it may even be harder.

The experiment shows that the uniform price algorithm can give much better solutions than one would expect when considering the approximation factor. Why this is the case is a good starting point for further research. The graph structure could play a role, or the commodities could have certain characteristics that make the problem easier. As mentioned in the introduction, approximation algorithms can serve as a basis for heuristics. One suggestion for a heuristic is to first apply a uniform price algorithm and then locally improve the solution. Then, one starts with a decent solution and can iteratively improve.

An interesting way to look at our problem is from a game theory perspective. Now that we know what one node will do (approximately), what will happen if the nodes correspond to strategic players? Will they settle into a stable scenario where everybody gets some revenue, or will it end in a "price war" where the revenue of each player becomes zero?

[5] Made by Sven Polak.

Chapter 4

Shortest Paths and Centrality in Uncertain Networks

4.1 Introduction

Computing the shortest path between two nodes in a network is a well-studied fundamental graph problem with numerous applications, including route planning, network routing protocols, computer games, and pathfinding in social networks. In the preliminaries, we have seen that Dijkstra's algorithm can be used to solve the shortest path problem. However, there are many other algorithms like, for example, Bellman-Ford (Bellman, 1958; Ford Jr, 1956), which can compute a shortest path in a graph with negative edge weights (but no negative cycles), the A^* search algorithm that uses heuristics to speed up Dijkstra's algorithm (Hart, Nilsson, and Raphael, 1968), an algorithm improving the running time if the cost on the edges is bounded (Ahuja et al., 1990), and there is a whole line of work on algorithms that use preprocessing to speed up computations (Delling et al., 2009).

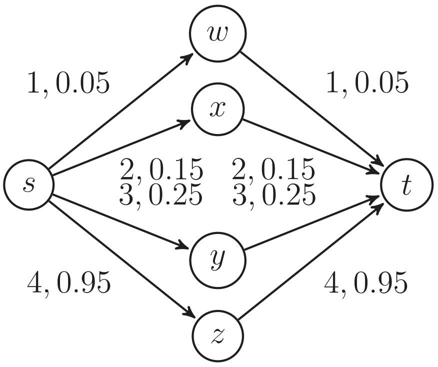
In a deterministic network, all edges are always available, but in many situations, the availability of edges in a network is uncertain. For example, in a wireless network, a connection can fail between two access points in the network, and in a road network, an accident can cause a road to be closed off for a while. In terms of the underlying graph, this means that the edge corresponding to the failing connection or the road with an accident is unavailable at that moment in time.

A way to model an uncertain network is by a graph in which each edge has a probability of existence. Given two nodes s and t in an uncertain network, we can ask the question: what is the shortest path between these two nodes? If we compute a shortest path when all edges are available, it might turn out that, in a realization of the graph, some edges in the path are unavailable. In this chapter, we introduce the notion of *Most Probable Shortest Path* (MPSP). The

MPSP between two nodes is the path that has the highest probability of being the shortest path. Here the probability is taken over all possible realizations of the graph. A formal definition will be given in Section 4.2.

Computing MPSPs is useful in many applications. Road networks can be modeled as uncertain graphs because of unexpected traffic jams (Hua and Pei, 2010). Instead of taking the distance-wise shortest route, a driver might want to take a longer route if it means there is a smaller chance of having issues along the way. MPSPs are also useful in routing over wireless sensor networks, where links between sensor nodes have a probability of failure: In this context, many applications not only require the shortest route, but also one with a high precision (Ghosh et al., 2007; Khan, Bonchi, Gionis, et al., 2014), such as being shortest with high probability. Brain networks are often represented as weighted uncertain graphs, where nodes are the brain regions of interest (ROIs), edges indicate potential co-activation between ROIs, edge distance represents the physical distance between ROIs, and edge probability indicates the strength of the co-activation signal (Craddock et al., 2013). Finding MPSPs between different ROIs of the brain could differentiate healthy brains from those with diseases, such as autism (Di Martino et al., 2010; García Domínguez et al., 2013). In our experiments, we present two case studies of MPSPs on sensor networks and brain networks.

To get some intuition for the problem, we take a look at an example.



Path P	Length	$\mathbb{P}(\mathtt{Sh}_s^t(P))$
$P_1 : (s, w, t)$	2	0.0025
$P_2 : (s, x, t)$	4	0.0224
$P_3 : (s, y, t)$	6	0.0609
$P_4 : (s, z, t)$	8	0.8250

Figure 4.1: Example of paths in an uncertain graph: $\mathbb{P}(\mathtt{Sh}_s^t(P))$ denotes the probability that path P is the shortest path from s to t.

Example 4.1.1. Each edge in the uncertain graph in Figure 4.1 is annotated with its length and its probability of existence (which is independent of the existence of the other edges). The table contains for all s,t-paths its length, and its probability of being the shortest path. The probability that a path is the shortest path is equal to the probability that it exists multiplied by the probability that none of the shorter paths exist. For source node s and target node t, the path $P_1 = (s, w, t)$ is the shortest (when all edges are available). However, this path only has a very small probability of existence of 0.0025. The longest path $P_4 = (s, z, t)$ has a probability of existence of $0.95^2 = 0.9025$, and because the probability that none of the three shorter paths exists is approximately 0.914 path P_4 has a probability of about 0.825 of being the shortest path.

In the previous chapter, we have already seen that it can be useful to know which nodes are the most central nodes in a network. The same arguments carry over to uncertain networks. Different centrality measures give different rankings of the nodes, and it depends on the situation which one is more useful. For uncertain graphs, examples of centrality measures that have been studied are expected betweenness centrality (Pfeiffer and Neville, 2011), where the centrality of a node is a weighted average of the centralities in the different realizations of the graph, and a centrality based on a notion of possible shortest paths (Wang and Lin, 2019), where they try to incorporate an approximation of the probabilities that paths are shortest paths in the centrality definition. We define a new centrality notion based on our most probable shortest paths that gives another ranking of the nodes that is fast to compute.

A common approach to finding shortest paths in uncertain networks is by means of a *filtering and verification* framework. A filtering approach used by Zou, Peng, and Zhao (2011) enumerates paths between the two given nodes in increasing order of length when all edges are available until a termination criterion is satisfied. Among the candidate paths generated, a sampling method is applied to approximately measure each candidate path's probability of being the shortest path. However, it can happen that the MPSP is not one of the shortest few paths when all edges are available (as also illustrated in Example 4.1). In fact, in general, there can be an exponential number of paths shorter than the MPSP, and these would all have to be enumerated before the MPSP is included in the candidate set. Thus, we ask the question:

> *Can we quickly include the MPSP in the candidate set without enumerating all paths shorter than the MPSP?*

To address this, we combine Monte Carlo sampling with Dijkstra's algorithm (referred to as DIJKSTRA+MC). From the source node, we run Dijkstra's algorithm, and an edge in the current possible world is sampled only upon request from Dijkstra's algorithm. That is, when a node is reached via Dijkstra's algorithm, its outgoing edges are sampled according to their probabilities, and only the sampled edges are considered for choosing the next node. As formally proved in Section 4.3.3, our method only needs a small number of DIJKSTRA+MC runs to include the MPSP in the candidate set with high probability. For intuition about our strategy, we again take a look at the graph from Figure 4.1.

Example 4.1.2. In Figure 4.1, there are four paths from s to t. The path P_4 (the longest path) is the MPSP. Enumerating paths in increasing order of length, requires us to enumerate all paths to include path P_4. On the other hand, a run of DIJKSTRA+MC has a probability of 0.825 of returning path P_4, hence with only two runs of DIJKSTRA+MC there is a probability of $1-(1-0.825)^2 = 0.969$ of returning path P_4. The probability that path P_4 is not included even decreases exponentially in the number of runs. By introducing a small probability of failure the number of iterations can be halved in this instance.

After we have generated a candidate set, we use Luby-Karp sampling (Karp, Luby, and Madras, 1989) to estimate for every path in the candidate set the probability that that path is the shortest path. As input, this algorithm takes a path and a set of shorter paths, and, by smartly reducing the universe from which it samples, it does not need many iterations for good approximations. In the setting where the set of shorter paths is actually the set of all shorter paths, this technique has been used previously (Zou, Peng, and Zhao, 2011). We find that even when the set of shorter paths is not exhaustive but reasonably large, it still works very well.

The sampling-based approach used here is different from the approach used in the previous chapter. In the previous chapter, we showed that our algorithms efficiently compute feasible solutions that provide a certain approximation guarantee. We are not able to do this for the algorithm we propose in this chapter. Because we show that the problem of computing the probability that a given path is the shortest path is #**P**-hard, it is highly unlikely that a polynomial-time algorithm exists. For deterministic filtering algorithms, it is often not too difficult to come up with a worst-case instance in which they have to enumerate a large number of paths. To overcome the issue of having to enumerate a large number of paths, we use a (randomized) Monte Carlo algorithm. By using a Monte Carlo algorithm, we introduce a small probability of failure and, in return, get a fast algorithm. Repeatedly running the algorithm makes the probability of error arbitrarily small. Because the algorithm is randomized and because we do not have good bounds on both the length and the probability of the optimal solution, we cannot prove approximation factors. However, we are able to prove that, with high probability, our algorithm returns the correct path. Also, we conducted extensive experiments showing that our method is fast in practice.

4.1.1 Our Contributions

In this chapter, we first study the fundamental problem of computing shortest-path queries in uncertain networks, and then we build on top of it a measure of betweenness centrality. The notion of shortest path in an uncertain graph should consider not only the length of a path but also the probability of existence of all edges on the path. Specifically, given an uncertain graph $\mathcal{G}$, a source node s, and a target node t our goal is to find the path P from s to t with the highest probability of being the shortest path, i.e., the probability with which P exists and no path shorter than P exists. We refer to such a path as the *Most Probable Shortest Path* (MPSP) from s to t.

1. We show that computing the MPSP is a hard problem. More specifically, we show that computing the probability that a given path is the shortest path is #P-hard. Next to this, we make observations that give intuition about why computing an MPSP is harder than computing a normal shortest path. A subpath of an MPSP does not also need to be an MPSP and the concatenation of two MPSPs is not guaranteed to be an MPSP.

2. We develop an efficient sampling-based algorithm to compute the MPSP. We provide probabilistic guarantees that the algorithm returns the correct path.

3. Next to computing an MPSP, we show how we can modify the algorithm to solve the following generalizations of the problem:

    Finding the top-k most probable shortest paths.

    Finding the MPSPs for multiple destinations from a single source (single-source multi-target) and its counterpart finding the MPSPs from multiple sources to a single destination (multi-source single-target).

    Finding the MPSP in uncertain multi-graphs, graphs where there can be multiple edges between the same nodes.

4. Using the notion of MPSP, we define a betweenness centrality measure and develop efficient sampling strategies to compute the top-k central nodes. Again, with probabilistic guarantees on returning the correct set.

5. Finally, we conduct extensive experiments.

    We show scalability over large-scale datasets and performance improvements against state-of-the-art methods (Cheng, Yuan, Wang, et al., 2014; Zou, Peng, and Zhao, 2011).

    We do case studies on sensor and brain networks showing the usefulness of MPSPs.

4.1.2 Related Work

Uncertain networks, i.e., graphs where each edge is associated with a probability of existence, have received a great deal of attention thanks to their expressivity and applicability in many real-world contexts. Following the bulk of the literature on uncertain graphs (Ball, 1986; Jin et al., 2011; Khan, Bonchi, Gullo, et al., 2018; Khan, Ye, and Chen, 2018; Potamias et al., 2010; Valiant, 1979; Yuan, Chen, and Wang, 2010; Zou, Peng, and Zhao, 2011) we adopt the well-established *possible* *world* semantics and assume that each edge has a probability of existence that is independent of the other edges. Researchers have studied k-nearest neighbors queries (Li et al., 2018; Potamias et al., 2010) where one, given a node, searches for the k nodes closest to that node given some metric, reachability queries (Ke, Khan, and Quan, 2019) where one tries to determine whether two nodes are connected via a path, and clustering (Han et al., 2019) where one wants to partition the nodes in k subsets such that the connection probability within clusters is maximized, to mention a few. Uncertainty in a network might arise due to noisy measurements (Aggarwal and Yu, 2009), edge prediction models (Liben-Nowell and Kleinberg, 2003), and explicit manipulation of edges, e.g., for privacy purposes (Boldi et al., 2012).

Shortest-path queries (Bonchi et al., 2014; Eppstein, 1998; Johnson, 1977) are one of the fundamental graph primitives with a plethora of applications, e.g., in traffic routing or finding functional pathways in biological networks. A critical application of shortest paths is the computation of *betweenness centrality* (Brandes, 2001; Freeman, 1977; Mahmoody, Tsourakakis, and Upfal, 2016; Riondato and Kornaropoulos, 2016), a measure of the importance of a node in a network based on its effectiveness in connecting pairs of other nodes via shortest paths.

Several variants of shortest-path queries over uncertain graphs have been studied in the literature. There is a line of work (Cheng, Yuan, Chen, et al., 2015; Cheng, Yuan, Wang, et al., 2014; Yuan, Chen, and Wang, 2010) that investigates *threshold-based shortest-path queries* in uncertain graphs, i.e., the problem of finding all paths having shortest-path probability larger than a predefined threshold. In particular, Cheng, Yuan, Chen, et al. (2015) and Cheng, Yuan, Wang, et al. (2014) consider a different uncertain graph model with correlation. The work closest to ours is probably by Zou, Peng, and Zhao (2011), which considers MPSP queries as we do, but it does not provide any hardness result nor any accuracy guarantee.

In (Zou, Peng, and Zhao, 2011), similarly to (Cheng, Yuan, Chen, et al., 2015; Yuan, Chen, and Wang, 2010), a *filtering and verification* framework is used. The sampling method for the verification part is based on the Luby-Karp algorithm (Karp, Luby, and Madras, 1989). To improve upon the filtering step in this framework we combine Monte Carlo sampling with Dijkstra's algorithm (DIJKSTRA+MC). The idea of DIJKSTRA+MC has been extensively used in probabilistic reachability queries (Jin et al., 2011; Ke, Khan, and Quan, 2019; Khan, Bonchi, Gionis, et al., 2014) and for the influence maximization problem (Borgs et al., 2014; Tang, Xiao, and Shi, 2014) where one tries to find a set of nodes that maximizes the expected set of nodes reached according to some probabilistic cascading rule.

The work by Cheng, Yuan, Wang, et al. (2014), discussed before, also employs a form of DIJKSTRA+MC followed by the Horvitz-Thompson unequal probability estimator, to compute the probability of being the shortest path, without any guarantees. While we employ DIJKSTRA+MC for effective and faster candidate generation, we then apply the Luby-Karp sampling to find the MPSP in this candidate set. Unlike Cheng, Yuan, Wang, et al. (2014), we provide probabilistic guarantees for our method, and we also experimentally demonstrate the superiority of our approach over the one of Cheng, Yuan, Wang, et al. (2014).

4.2 Preliminaries

Let $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$ be a probabilistic (or uncertain) directed graph, where $c_e \in \mathbb{R}_{\geq 0}$ is a non-negative edge length, and $p_e \in (0, 1]$ assigns a probability of existence to each edge $e \in E$. We adopt the well-established *possible world* semantics (Ball, 1986; Jin et al., 2011; Khan, Bonchi, Gullo, et al., 2018; Khan, Ye, and Chen, 2018; Potamias et al., 2010; Valiant, 1979; Yuan, Chen, and Wang, 2010; Zou, Peng, and Zhao, 2011) and assume that edge probabilities are independent of each other: the uncertain graph $\mathcal{G}$ is interpreted as a probability distribution over the $2^{|E|}$ deterministic graphs (possible worlds) $G = (V, E_G, (c_e)_{e \in E}) \in \mathcal{G}$ (when writing $\in \mathcal{G}$ we interpret $\mathcal{G}$ as the set of all possible realizations of $\mathcal{G}$) obtained by sampling each edge $e \in E$ independently at random with probability p_e. That is, the probability of observing the possible world $G = (V, E_G, (c_e)_{e \in E})$ with $E_G \subseteq E$ is:

$$\mathbb{P}(G) = \prod_{e \in E_G} p_e \prod_{e \in E \setminus E_G} (1 - p_e). \tag{4.1}$$

In an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, let $\mathcal{P}(\mathcal{G}, s, t)$ denote the set of all paths from s to t when all edges in E are present. Given a path $P = (e_1, e_2, \ldots, e_n)$, the event that P exists (resp. does not exist) is denoted by $\mathbf{X}(P)$ (resp. $\overline{\mathbf{X}}(P)$), and observe the relation $\mathbb{P}\big(\mathbf{X}(P)\big) = \prod_{i=1}^{n} p_{e_i} = 1 - \mathbb{P}\big(\overline{\mathbf{X}}(P)\big)$. Given two paths P, Q with possibly overlapping edges we write $P \setminus Q$ for the set of edges that are present in P but not in Q. The definition of $\mathbf{X}$ then extends in the obvious way by $\mathbb{P}(\mathbf{X}(P \setminus Q)) = \prod_{e \in P \setminus Q} p_e$. We also denote by $\mathtt{Sh}_s^t(P)$ the event that P happens to be a shortest path from s to t, whose probability is:

$$\mathbb{P}(\mathtt{Sh}_s^t(P)) = \sum_{G \in \mathcal{G}} \mathbb{P}(G) \cdot \mathbb{1}_{[P \in \mathsf{SP}(G,s,t)]}, \tag{4.2}$$

where $\mathbb{1}_{[.]}$ is the indicator function.

The main problem studied in this chapter requires finding the path that has the maximum probability of being a shortest path.

Problem 4.2.1. (Most Probable Shortest Path (MPSP)). *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$ and two nodes $s, t \in V$, find a most probable shortest path (MPSP) from s to t. Formally:*

$$MPSP(\mathcal{G}, s, t) \in \arg \max_{P \in \mathcal{P}(\mathcal{G}, s, t)} \mathbb{P}(\mathit{Sh}_s^t(P)). \tag{4.3}$$

In this chapter, we denote by $\mathcal{M}(\mathcal{G}, s, t)$ the set of MPSPs from s to t, and by $\mathbb{M}(\mathcal{G})$ the set of all MPSPs between all pairs of nodes, i.e,.

$$\mathbb{M}(\mathcal{G}) = \bigcup_{(s,t)\in V\times V} \mathcal{M}(\mathcal{G}, s, t).$$

4.2.1 Hardness of the Problem

One reason that makes Problem 4.2.1 challenging is that even computing the probability of being the shortest path between two given nodes, for a given path, is hard.

Theorem 4.2.2. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$ and a path $P \in \mathcal{P}(\mathcal{G}, s, t)$, the problem of computing the probability of P being a shortest path from s to t in $\mathcal{G}$ is $\#\mathbf{P}$-hard.*

Proof: We prove the $\#\mathbf{P}$-hardness by polynomial-time reduction from the s,t-*connectedness* problem, which is known to be $\#\mathbf{P}$-hard (Valiant, 1979). Given a (deterministic) graph $G = (V, E)$ and two nodes s and t, the goal of the s,t-connectedness problem is to find the number of subgraphs of G in which there is a path from s to t.

Consider an arbitrary instance of the s,t-connectedness problem with inputs $G = (V_1, E_1)$ and two nodes $s, t \in V_1$. Let $n = |V_1|$. The deterministic graph G is converted to an uncertain graph $\mathcal{G} = (V_1 \cup V_2, E_1 \cup E_2, (c_e)_{e\in E_1\cup E_2}, (p_e)_{e\in E_1\cup E_2})$, where $V_2 = \{v_1, v_2, \dots, v_n\}$ is a set of n new nodes and $E_2 = \{(s, v_1), (v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n), (v_n, t)\}$. In other words, G is augmented with a new path $P = ((s, v_1), (v_1, v_2), \dots, (v_n, t))$ from s to t. We define $c_e = 1\ \forall e \in E_1 \cup E_2$ and

$$p_e = \begin{cases} \frac{1}{2} & \text{if } e \in E_1 \\ 1 & \text{if } e \in E_2. \end{cases}$$

We make three observations.

(i) Every possible world $G' \in \mathcal{G}$ for which $\mathbb{P}(G') > 0$ contains the path P and has $\mathbb{P}(G') = \left(\frac{1}{2}\right)^{|E_1|}$.

(ii) There is a bijection between the set of subgraphs of G and the set of possible worlds of $\mathcal{G}$ that exist with positive probability. A subgraph $G'' = (V'', E'')$ of G can be mapped to the possible world $G' = (V'' \cup V_2, E'' \cup E_2, (c_e)_{e\in E''\cup E_2})$ of $\mathcal{G}$. This mapping is clearly one-to-one since $V'' \cap V_2 = \phi$ and $E'' \cap E_2 = \phi$ by definition. To see why it is onto, note that any possible world of $\mathcal{G}$, that exists with positive probability, must contain all edges in E_2, since $p_e = 1$ $\forall e \in E_2$. Hence, given a possible world $G' = (V', E', (c_e)_{e\in E'})$ of $\mathcal{G}$, there exists a subgraph $G'' = (V' \setminus V_2, E' \setminus E_2, (c_e)_{e\in E'\setminus E_2})$ which is the pre-image of G' under the mapping.

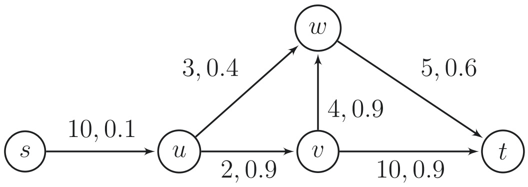
(iii) For a subgraph $G'' = (V'', E'')$ of G and its corresponding possible world $G' = (V'' \cup V_2, E'' \cup E_2, (c_e)_{e \in E'' \cup E_2})$ of $\mathcal{G}$, P is the shortest path from s to t in G' if and only if s and t are disconnected in G''. The *'if'* part is trivial. The *'only if'* part follows since $c(P) = n + 1$ and $c(P') \leq n - 1$, where P' denotes any path from s to t in G''.

Putting together the above observations, we obtain the following:

$$\begin{aligned}
\mathbb{P}(\mathtt{Sh}_s^t(P)) &= \sum_{G' \in \mathcal{G}} \mathbb{P}(G') \cdot \mathbb{1}_{[P \in \mathsf{SP}(G',s,t)]} \\
&= 1 - \sum_{G' \in \mathcal{G}} \mathbb{P}(G') \cdot \mathbb{1}_{[P \notin \mathsf{SP}(G',s,t)]} \\
&= 1 - \left(\frac{1}{2}\right)^{|E_1|} \sum_{G' \in \mathcal{G} \wedge \mathbb{P}(G')>0} \mathbb{1}_{[P \notin \mathsf{SP}(G',s,t)]}.
\end{aligned}$$

From observation (iii), the summation term in the last line is exactly the number of subgraphs of G in which the nodes s and t are connected. Thus, a solution to our problem on $\mathcal{G}$ provides a solution to the s,t-connectedness problem on G. This reduction involves $\mathcal{O}(n)$ node and edge additions to G, and hence takes time polynomial in the size of G. □

In addition to #**P**-hardness, there are some other properties of MPSPs that make our problem hard. *Many of the classical properties of shortest paths over deterministic graphs no longer hold for MPSPs in uncertain graphs.* Firstly, the concatenation of two MPSPs does not have to be an MPSP. Secondly, the subpath of an MPSP is also not necessarily an MPSP. We demonstrate these properties next, using the uncertain graph $\mathcal{G}$ in Figure 4.2.



Path P	$c(P)$	$\mathbb{P}(\mathtt{Sh}_s^t(P))$
$P_1 : (s, u, w, t)$	18	0.024
$P_2 : (s, u, v, w, t)$	21	0.029
$P_3 : (s, u, v, t)$	22	0.035

Figure 4.2: An example to demonstrate properties of MPSP

Observation 4.2.3. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, an MPSP $P \in \mathbb{M}(\mathcal{G})$, and a subpath Q of P, it is possible that $Q \notin \mathbb{M}(\mathcal{G})$.*

Consider the path $P_3 = (s, u, v, t) \in \mathcal{M}(\mathcal{G}, s, t)$ and its subpath (v, t). The probabilities of being the shortest path from v to t are $\mathbb{P}(\mathtt{Sh}_v^t(v, w, t)) = 0.540$ (if it exists it is the shortest paths) and $\mathbb{P}(\mathtt{Sh}_v^t(v, t)) = 0.414$ (the probability that this edge exists while the alternative path (v, w, t) does not) , so that (v, t) is not even the MPSP from v to t. The observation follows.

Recall that given two paths $P = (e_1, \ldots, (u, v))$ and $Q = ((v, w), \ldots, e_n)$, the *concatenation* of P and Q, denoted by $P \cdot Q$, is defined as the path $P \cdot Q = (e_1, \ldots, (u, v), (v, w), \ldots, e_n)$. Note that the concatenation of two paths P and Q is defined only when the target node of P is the same as the source node of Q. The next observation states that the concatenation of two MPSPs is not necessarily an MPSP.

Observation 4.2.4. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$ and two MPSPs $P, Q \in \mathbb{M}(\mathcal{G})$, such that the target node of P is the same as the source node of Q, it is possible that $P \cdot Q \notin \mathbb{M}(\mathcal{G})$.*

Notice that since $P = (s, u, v)$ is the only path from s to v, it is clear that $\mathcal{M}(\mathcal{G}, s, v) = \{(s, u, v)\}$. Also, as shown in Observation 4.2.3, $Q = (v, w, t) \in \mathcal{M}(\mathcal{G}, v, t)$. However, $P \cdot Q = (s, u, v, w, t) = P_2 \notin \mathcal{M}(\mathcal{G}, s, t)$ and hence $P \cdot Q \notin \mathbb{M}(\mathcal{G})$.

4.2.2 Benchmark: Filtering and Verification

Before we outline our new approach we start by shortly discussing the current state-of-the-art algorithm by Zou, Peng, and Zhao (2011). In our experiments (Section 4.5) we use their approach as a benchmark. The algorithm consists of two steps: generating a set of candidate paths containing the MPSP, and using Luby-Karp sampling to find the MPSP in this set.

For step 1, given a source s and a target t, Yen's algorithm[1] (Yen, 1971) is used to progressively generate s, t-paths $P_1, P_2, P_3, \ldots$ in non-decreasing order of length, when all edges are available until some stopping criterion is met. On a high level, the stopping criterion is as follows. For any i, using paths $P_1, \ldots, P_i$, a lower bound $LB(P_i)$ (possibly consisting of multiple combined lower bounds) and an upper bound $UB(P_i)$ on the probability that the path P_i is the shortest path, i.e., on $\mathtt{Sh}_s^t(P_i)$, is computed. The upper bound is monotonically decreasing in i. If some path P_i is the shortest path with probability at most ϵ, all paths P_j generated after path i $(j > i)$ will have a probability of being the shortest path that is smaller than ϵ. Hence, if the probability for path i is too low, then it is also too low for paths P_j with $j > i$. More formally, if $UB(P_i) < \epsilon$ for some $\epsilon > 0$, $UB(P_j) < \epsilon$ for all $j > i$. For including paths in the candidate set, the algorithm continues to generate paths until $UB(P_{i+1}) < \max_{j \in [i]}\{LB(P_j)\}$ for some i or

[1] Yen's algorithm generates paths in non-decreasing order of length. It starts by determining a shortest path (e.g., by using Dijkstra's algorithm). Then it alternates between two phases. Phase 1: it generates new paths by explicitly excluding already found paths and it adds them to a pool of candidates. Phase 2: the shortest path from the pool is the next shortest path, and it goes back to phase 1

until all possible paths have been found. If some path is the shortest path with *at least* some probability, then we can ignore all paths that are the shortest path with *at most* that probability. This gives the candidate set $\{P_1, \ldots, P_i\}$.

Step 2 consists of running the Luby-Karp algorithm (Karp, Luby, and Madras, 1989) to approximate the probability that each path in the candidate set is the MPSP. It returns the path with the highest such probability. The pseudocode of the Luby-Karp algorithm is given in Algorithm 6 and more details in Subsection 4.3.1.

Two shortcomings of step 1 have an influence on the performance of this method. First, the number of candidates generated can be very large, even exponential in the input size. The lower bound on the probability of being the shortest path is actually the maximum of two different lower bounds in (Zou, Peng, and Zhao, 2011). For both lower bounds LB it holds that $LB(P_j) \leq \mathbb{P}(\mathbf{X}(P_j))$. The upper bound on the probability of path P_i being the shortest path is computed as $UB(P_i) = 1 - \sum_{j=1}^{i-1} LB(P_j)$. If the probability of existence of the MPSP is low, then the probability of existence of the other shorter paths will generally also be low. Hence, the upper bound will decrease very slowly, and it can take a lot of time before the candidate generation terminates.

The second shortcoming is the computational cost of the candidate generation. Assume that we generate k paths before the candidate generation terminates. Yen's algorithm has time complexity $\mathcal{O}(k|V|(|E| + |V| \log |V|))$. As mentioned in the first shortcoming, the number of candidates k can become very large, and even if it is small, we have the $|V||E|$ factor. Empirically (Section 4.5) we find that the candidate generation does not finish within reasonable time for our larger datasets.

4.3 Proposed Solution

We propose a two-phase algorithm to find the MPSP between two nodes in an uncertain graph. In the first phase, we compute paths that are candidates for being the MPSP (via DIJKSTRA+MC), and in the second phase, we approximate the probability of each candidate path being the shortest path (via Luby-Karp algorithm). Our method is described in Section 4.3.1 and theoretical guarantees on the quality of the returned path are provided in Section 4.3.3.

The idea of DIJKSTRA+MC is simple, yet effective and efficient for candidate generation as we argued in Example 4.1.2 (Section 4.1). Our novel algorithmic contributions include pairing up DIJKSTRA+MC with the Luby-Karp algorithm for ultimately finding the MPSP approximately, with probabilistic guarantees. Empirical results attest that our algorithm has better accuracy and scalability over the benchmark (Zou, Peng, and Zhao, 2011) (Section 4.2.2), and over more advanced sampling approaches, e.g., Horvitz-Thompson unequal probability estimator (Section 4.5.4). Among other novel algorithmic contributions, we extend

Algorithm 4 Approximating the MPSP from s to t

```
Input: Uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, source $s$, target $t$, positive
    integers $m$ and $N$
Output: An (approximate) MPSP from $s$ to $t$
 1: $CP \leftarrow \phi$                                              /* Phase 1 */
 2: for $i = 1$ to $m$ do
 3:     $P \leftarrow$ Algorithm 5$(\mathcal{G}, s, t)$                      /* DIJKSTRA+MC */
 4:     if $P \neq P_\infty$ then
 5:         $CP \leftarrow CP \cup \{P\}$
 6:     end if
 7: end for
 8: $CP.sort$(in non-decreasing order of length)                  /* Phase 2 */
 9: for $i = 1$ to $|CP|$ do
10:     $\hat{p}(CP[i]) \leftarrow$ Algorithm 6 ($\mathcal{G}$, $s$, $t$, $CP[i]$, $CP[1 : (i-1)]$, $N$)
11:                                                                   /* Luby Karp */
12: end for
13: return $\arg\max_{P\in CP}[\hat{p}(P)]$
```

our method to find the top-k MPSPs for $k > 1$ (Section 4.3.2), single-source and single-target MPSP queries (Section 4.3.4), and to compute the MPSPs in uncertain multi-graphs (Section 4.3.4). Our final technical contribution is to define a novel MPSP-BTW as a concrete application (Section 4.4); we then develop efficient sampling strategies to compute the top-k central nodes, with theoretical quality guarantees.

4.3.1 Two-Phase Algorithm

Algorithm 4 contains pseudocode for our algorithm which consists of two phases.

Phase 1: DIJKSTRA+MC

Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$ and a pair of nodes $(s, t) \in V \times V$, the first phase involves computing paths that are candidates for being the MPSP from s to t. This is done by performing m independent runs of Dijkstra's algorithm on $\mathcal{G}$, where m is a hyperparameter (lines 2 to 7 of Algorithm 4). Dijkstra's algorithm on an uncertain graph is similar to the classic algorithm on deterministic graphs except that when the algorithm reaches a node in the uncertain graph, its outgoing edges are sampled according to their respective probabilities (Algorithm 5). At any stage, only the sampled edges are considered

Algorithm 5 Candidate Generation with DIJKSTRA+MC

```
Input: Uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, source $s$, target $t$
Output: A path from $s$ to $t$
u ← s
visited ← {s}
P[v] ← $P_\infty$ ∀v ∈ V                    /* to store the shortest path from s to v */
while $u \neq t$ and $\mathbf{P}[u] \neq P_\infty$ do
    for all $e = (u, v) \in E$ with $v \notin visited$ do
        /* If we find a shorter path to v, store it with probability $p_e$ */
        if $c(\mathbf{P}[u]) + c_e < c(\mathbf{P}[v])$ then
            With probability $p_e$, $\mathbf{P}[v] \leftarrow \mathbf{P}[u] \cdot (e)$
        end if
    end for
    $u \leftarrow \arg\min_{v \in V \setminus visited} c(\mathbf{P}[v])$          /* The closest unvisited node */
    $visited \leftarrow visited \cup \{u\}$
end while
return P[t]
```

for choosing the next node. This is equivalent to running Dijkstra's algorithm[2] on a possible world $G \in \mathcal{G}$. If t is reachable from s in the sampled possible world, then the Dijkstra's algorithm on $\mathcal{G}$ results in an s,t-path which is added to the set of candidate paths denoted by CP. Otherwise, if t is not reachable, then P_∞ is returned.

Phase 2: Probability Approximation

In the second phase, the Luby-Karp algorithm (Algorithm 6) is employed to compute an approximation of the probability of each candidate path being the shortest s,t-paths in $\mathcal{G}$. Intuitively, given a path P and some other shorter paths $P_1, \ldots, P_n$ from s to t, along with a hyperparameter N, the algorithm first estimates the probability $\hat{p}$ of existence of any of the paths shorter than P by generating N suitable possible worlds via Monte Carlo sampling, and then it returns the value $(1 - \hat{p}) \cdot \mathbb{P}\big(\mathbf{X}(P)\big)$ as an estimate of $\mathbb{P}(\mathtt{Sh}_s^t(P))$.

We elaborate in more detail on why Algorithm 6 returns a good estimate. Fix a path P and assume $P_1, \ldots, P_n$ are all the paths shorter than P. Note that

$$\begin{aligned}\mathbb{P}(\mathtt{Sh}_s^t(P)) &= \mathbb{P}(\mathbf{X}(P)) \cdot \mathbb{P}(\cap_{i \in [n]} \overline{\mathbf{X}}(P_i) \mid \mathbf{X}(P)) \\ &= \mathbb{P}(\mathbf{X}(P)) \cdot (1 - \mathbb{P}(\cup_{i \in [n]} \mathbf{X}(P_i) \mid \mathbf{X}(P)).\end{aligned}$$

[2] In Algorithm 5, instead of DIJKSTRA+MC, one may employ YEN+MC. Yen's algorithm (Yen, 1971) reports the top-l ($l \geq 1$) shortest s,t-paths in each possible world, thereby generating more candidate paths, and possibly improving the quality of the returned MPSPs. However, we empirically found that increasing the value of l results in a very small increase in quality but a very large increase in running time. Thus we choose the default value $l = 1$, which is the same as running DIJKSTRA+MC.

Algorithm 6 Luby-Karp: Estimate $\mathbb{P}(\mathtt{Sh}_s^t(P))$ for a path P from s to t

```
Input: Uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, source $s$, target $t$, $s,t$-paths
    $P$ and $\{P_1, \dots, P_n\}$ shorter than $P$, positive integer $N$
Output: An estimate of $\mathbb{P}(\mathtt{Sh}_s^t(P))$
$C \leftarrow 0$, $S \leftarrow \sum_{i=1}^n \mathbb{P}\big(\mathbf{X}(P_i \setminus P)\big)$
for $r = 1$ to $N$ do
        Sample $i \in [1, n]$ with probability $\frac{\mathbb{P}\left(\mathbf{X}(P_i \setminus P)\right)}{S}$
        Sample $G = (V, E_G, (c_e)_{e\in E}) \in \mathcal{G}$ such that $(P_i \cup P) \subseteq E_G$
        if $\forall(j < i)[(P_j \setminus P) \nsubseteq E_G]$ then
                $C \leftarrow C + 1$
        end if
end for
$\hat{p} \leftarrow \frac{C}{N} \cdot S$
return $(1 - \hat{p}) \cdot \mathbb{P}\big(\mathbf{X}(P)\big)$
```

The variable $\hat{p}$ will be an estimate of $\mathbb{P}(\cup_{i\in[n]}\mathbf{X}(P_i) \mid \mathbf{X}(P))$. A naive way to estimate the probability would be to sample worlds $G \in \mathcal{G}$ in which P exists and count how often at least one of the P_i exists. However, if the probability we want to estimate is (exponentially) small this requires an (exponentially) large number of iterations for accurate estimates (Karp, Luby, and Madras, 1989). Instead, we use a biased estimator on a smaller universe in which we know that P and at least one of the P_i exist. Doing this requires fewer iterations for a good estimate.

Let $\hat{\mathcal{G}}$ be the set of possible realizations of $\mathcal{G}$ in which both P and at least one P_i for $i \in [n]$ exist. Consider the sets

$$T = \{(G, P_i) \mid G \in \hat{\mathcal{G}}, i \in [n]\},$$
$$T' = \{(G, P_i) \mid G \in \hat{\mathcal{G}}, i \text{ is the smallest index for which } P_i \text{ exists in } G\}.$$

We are interested in the quantity

$$\sum_{(G,P_i)\in T'} \mathbb{P}(G \mid \mathbf{X}(P)) = \sum_{G\in\hat{\mathcal{G}}} \mathbb{P}(G \mid \mathbf{X}(P)) = \mathbb{P}(\cup_{i\in[n]}, \mathbf{X}(P_i) \mid \mathbf{X}(P))$$

where the first equality follows because there can only be one P_i with the smallest index in G. Each table cell in Table 4.3 corresponds to an element in T. We have put an x in cell (G_j, P_i) if P_i exists in G_j and an x' if P_i is the path with the smallest index that exists in G_j.

Let $S = \sum_{i\in[n]} \mathbb{P}(\mathbf{X}(P_i \setminus P))$ be a normalizing factor. Lines three and four in Algorithm 6 consist of the following sample steps

1. Sample i with probability $\frac{\mathbb{P}(\mathbf{X}(P_i \setminus P))}{S}$.

2. Sample $G = (V, E_G) \in G$ such that $P_i \cup P \subseteq E_G$.

	P_1	P_2	$\dots$	P_n
G_1		x'	$\dots$	x
G_2	x'	x	$\dots$	
$\vdots$				
G_k			$\dots$	x'

Figure 4.3: Table for illustrating the set T

Element (G_j, P_i) is sampled with probability

$$\begin{aligned}\mathbb{P}(\text{sample } (G_j, P_i)) &= \mathbb{P}(\text{sample } P_i) \cdot \mathbb{P}(\text{ sample } G_j \mid \mathbf{X}(P_i) \cap \mathbf{X}(P)) \\ &= \frac{\mathbb{P}(\mathbf{X}(P_i \setminus P))}{S} \cdot \mathbb{P}(\text{ sample } G_j \mid \mathbf{X}(P_i) \cap \mathbf{X}(P)) \\ &= \frac{\mathbb{P}(G_j \mid \mathbf{X}(P)) \cdot \mathbb{P}(\mathbf{X}(P_i) \mid \text{ sampled } G_j, \mathbf{X}(P))}{S} \\ &= \frac{\mathbb{P}(G_j \mid \mathbf{X}(P))}{S}.\end{aligned}$$

The third probability follows from Bayes' Theorem and the last equality because we know that P_i exists in G_j.

Because there is exactly one element of T' per row in Table 4.3 the probability that we sample an element of T' is

$$\begin{aligned}\sum_{(G,P_i)\in T'} \mathbb{P}(\text{sample } (G, P_i)) &= \sum_{(G,P_i)\in T'} \frac{\mathbb{P}(G \mid \mathbf{X}(P))}{S} \\ &= \frac{1}{S} \sum_{G\in\hat{\mathcal{G}}} \mathbb{P}(G \mid \mathbf{X}(P)) \\ &= \frac{\mathbb{P}(\cup_{i\in[n]} \mathbf{X}(P_i) \mid \mathbf{X}(P))}{S}.\end{aligned}$$

Lines five to seven in Algorithm 6 verify whether the sampled element is in T'. If C_r is the random variable that is one if iteration r raises C by one and 0 otherwise then $C = \sum_{r=1}^{N} C_i$. The discussion above has shown that $\mathbb{E}[C_i] = \frac{\mathbb{P}(\cup_{i\in[n]}\mathbf{X}(P_i) \mid \mathbf{X}(P))}{S}$. Hence, by linearity of expectation

$$\mathbb{E}[\hat{p}] = \mathbb{E}\left[S \cdot \frac{C}{N}\right] = S \cdot \frac{N \cdot \mathbb{P}(\cup_{i\in[n]}\mathbf{X}(P_i) \mid \mathbf{X}(P))}{N \cdot S} = \mathbb{P}(\cup_{i\in[n]}\mathbf{X}(P_i) \mid \mathbf{X}(P)).$$

We discuss in Section 4.3.3 how many iterations are needed for a good approximation.

Notice that in order to approximate the probability of a path P being the shortest path in $\mathcal{G}$, the Luby-Karp algorithm, as described in (Zou, Peng, and Zhao, 2011), requires as input all the paths that are shorter than P. Although the set of candidate paths computed after m runs of Algorithm 5 does not necessarily include all such paths, we shall show in Section 4.3.3 that we can still provide good approximation guarantees.

Time Complexity

In Phase 1, we perform m Dijkstra's runs on the uncertain graph $\mathcal{G}$, which has time complexity $\mathcal{O}\big(m(|E| + |V| \log |V|)\big)$. However, due to sampling of edges, Dijkstra is run on a smaller graph than the original uncertain graph, thus practically it is even more efficient. In Phase 2, first we need to sort (at most) m distinct candidate paths. This step requires $\mathcal{O}(m \log m)$ time. Then, we run Algorithm 6 for each candidate path. For sampling the possible world in line four of Algorithm 6, observe that we can assume that all edges in $P_i \cup P$ are included and for the remaining edges in the graph (actually only the edges that are in one of $P_1, \ldots, P_n$) we flip a coin. Because we do m DIJKSTRA+MC runs we know that $n \leq m$. If $|P|$ is an upper bound on the number of edges in a path we can do the bookkeeping for lines five to seven in time $\mathcal{O}(m|P|)$. With N the number of Monte Carlo runs the Luby-Karp algorithm runs in time $\mathcal{O}(N(|E| + m|P|))$. Therefore, the overall time complexity of our method is: $\mathcal{O}\left(m\big(N(|E| + m|P|) + |V| \log |V| + \log m\big)\right)$.

Space Complexity

Both DIJKSTRA+MC and Luby-Karp have low memory footprints, and do not have much additional overhead other than storing the graph, which is $\mathcal{O}(|E| + |V|)$ via adjacency list. Additionally, DIJKSTRA+MC generates at most m candidate paths, which require at most $\mathcal{O}(m|E|)$ storage, but practically it is less since a path generally has fewer than $|E|$ edges. Thus, the space complexity of our method is $\mathcal{O}(m|E| + |V|)$.

4.3.2 Extension to Top-k MPSPs

The method presented in Section 4.3.1 can be easily extended to compute the top-k MPSPs where $k > 1$. We notice that if the number of candidate paths is smaller than or equal to k, we return all the candidate paths. Otherwise, we modify Algorithm 4 so that it stores every candidate path P and the estimate of $\mathbb{P}(\mathtt{Sh}_s^t(P))$ in decreasing order of the probabilities, and then it returns the top-k elements.

We provide theoretical guarantees that *with high probability, the true top-k shortest paths are the ones returned by our algorithm.*

4.3.3 Accuracy Guarantees

As a first step, notice that an s, t-path P is returned after one run of Algorithm 5 if and only if Algorithm 5 samples a possible world of $\mathcal{G}$ in which P is a shortest path from s to t. Thus, the probability of the former is equal to that of the latter, which, by definition, is equal to $\mathbb{P}(\mathtt{Sh}_s^t(P))$. Extending this to m runs of Algorithm 5, denoting by CP the set of all (candidate) paths returned, for any given path P, we have $\mathbb{P}(P \in CP) = 1 - (1 - \mathbb{P}(\mathtt{Sh}_s^t(P)))^m$.

Further extending to k paths, the probability of any given set $\{P_1, \dots, P_k\}$ of k s,t-paths being included in CP is, by the union bound,

$$\begin{aligned} \mathbb{P}(\{P_1, \dots, P_k\} \subseteq CP) &= \mathbb{P}(\wedge_{i=1}^k (P_i \in CP)) \\ &= 1 - \mathbb{P}(\vee_{i=1}^k (P_i \notin CP)) \\ &\geq 1 - \sum_{P_i \in \{P_1, \dots, P_k\}} \mathbb{P}(P_i \notin CP) \\ &= 1 - \sum_{P_i \in \{P_1, \dots, P_k\}} (1 - \mathtt{Sh}_s^t(P_i))^m. \end{aligned} \tag{4.4}$$

A key observation is that, for an MPSP P, $\mathbb{P}(P \in CP)$ is very high for a reasonably large value of $\mathbb{P}(\mathtt{Sh}_s^t(P))$, even for small m. For example, consider the MPSP P_4 in the graph in Figure 4.1 for which $\mathbb{P}(\mathtt{Sh}_s^t(P_4)) = 0.825$. Setting $m = 20$ yields $\mathbb{P}(P_4 \in CP) > 0.999$. Also, in our experiments, the path P returned by our method for most of the synthetic networks and the road networks for the smaller hop queries satisfies $\mathbb{P}(\mathtt{Sh}_s^t(P)) > 0.06$, and hence $\mathbb{P}(P \in CP) > 0.7$ with $m = 20$. One can always (exponentially) boost the probabilities by raising m.

Before proceeding, we define some useful notations that we use throughout the remainder of the section. Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, a source node s, a target node t, a set of s,t-paths CP, and any path $P \in CP$, we use the following notation:

$\mathbf{A}(P)$ is the set of all paths in $\mathcal{G}$ that are shorter than P.

$\mathbf{C}(P)$ is the set of all paths in CP shorter than P, i.e., $CP \cap \mathbf{A}(P)$.

$\mathbf{M}(P) = \mathbf{A}(P) \setminus \mathbf{C}(P)$.

$\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)$ denotes the probability that P exists and no path in $\mathbf{C}(P)$ exists, i.e., $\mathbb{P}\big(\mathbf{X}(P)\big)\big[1 - \mathbb{P}\big(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\big)\big]^3$ where $Q \setminus P$ is the set of all edges in Q that are not in P. Clearly, $\mathbf{p_{ne}}\big(P, \mathbf{A}(P)\big) = \mathbb{P}(\mathtt{Sh}_s^t(P))$.

$\mathbf{p_m}\big(P, \mathbf{C}(P)\big)$ is the sum over all paths Q shorter than P and missing from CP of the probability that Q is the shortest s,t-path and that P exists, i.e., $\sum_{Q \in \mathbf{M}(P)} \mathbb{P}(\mathtt{Sh}_s^t(Q) \wedge \mathbf{X}(P))$.

$\hat{p}\big(P, \mathbf{C}(P)\big)$ is the output of Alg. 6 ($\mathcal{G}$, s, t, P, $\mathbf{C}(P)$, N).

[3] Observe that because the edges are sampled independently it holds that $\mathbb{P}\big(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q \setminus P)\big) = \mathbb{P}\big(\bigcup_{Q \in \mathbf{C}(P)} \mathbf{X}(Q) \mid \mathbf{X}(P)\big)$. In this section, we regularly use this idea that excluding the edges in P is in some sense the same as conditioning on the existence of P

Even if the true top-k MPSPs are included in CP, the probability of them being the paths finally returned depends on the quality of the approximation computed in Algorithm 6 for every single path in CP. Fortunately, there is a guarantee on this quality (Karp, Luby, and Madras, 1989; Zou, Peng, and Zhao, 2011).

Theorem 4.3.1. (Karp, Luby, and Madras, 1989; Zou, Peng, and Zhao, 2011). *Consider an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, a source node s and a target node t, a set of s,t-paths CP, and a path $P \in CP$. $\hat{p}\big(P, \mathbf{C}(P)\big)$ is an accurate estimate of $\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)$ with high probability. More formally, for all $\epsilon \in [0, 2]$,*

$$\mathbb{P}\Big(\big|\hat{p}\big(P, \mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big)\big| \geq \epsilon \Big) \leq 2 \exp\left(- \frac{N\epsilon^2}{4|\mathbf{C}(P)|} \right). \tag{4.5}$$

However, as mentioned in Section 4.3.1, the quality of approximating $\mathbb{P}(\texttt{Sh}_s^t(P))$ could be hampered because the set CP computed after m runs of Algorithm 5 may not include all paths shorter than the path in question. We shall show that, even then, the approximation made by Algorithm 6 is very accurate with high probability. To this end, we first provide a lower and an upper bound on the difference in probability of being the shortest path resulting from missing out on some shorter paths.

Theorem 4.3.2. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, a source node s and a target node t, let CP denote a set of paths from s to t. Consider a path $P \in CP$. Then*

$$0 \leq \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathit{Sh}_s^t(P)) \leq \mathbf{p_m}\big(P, \mathbf{C}(P)\big). \tag{4.6}$$

Proof: We have, by definition, the following:

$$\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) = \mathbb{P}\big(\mathbf{X}(P)\big) \left[1 - \mathbb{P}\left(\bigcup_{Q\in\mathbf{C}(P)} \mathbf{X}(Q \setminus P) \right) \right],$$

$$\mathbb{P}(\texttt{Sh}_s^t(P)) = \mathbb{P}\big(\mathbf{X}(P)\big) \left[1 - \mathbb{P}\left(\bigcup_{Q\in\mathbf{A}(P)} \mathbf{X}(Q \setminus P) \right) \right].$$

Let us define:

$$D_A = \mathbb{P}\left(\bigcup_{Q\in\mathbf{A}(P)} \mathbf{X}(Q \setminus P) \right), \quad D_C = \mathbb{P}\left(\bigcup_{Q\in\mathbf{C}(P)} \mathbf{X}(Q \setminus P) \right).$$

This means that

$$\begin{aligned}
&\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P)) \\
&= \mathbb{P}\big(\mathbf{X}(P)\big)\left[\mathbb{P}\left(\bigcup_{Q\in\mathbf{A}(P)} \mathbf{X}(Q\setminus P)\right) - \mathbb{P}\left(\bigcup_{Q\in\mathbf{C}(P)} \mathbf{X}(Q\setminus P)\right)\right] \\
&= \mathbb{P}\big(\mathbf{X}(P)\big)\cdot\big(D_A - D_C\big). \qquad (4.7)
\end{aligned}$$

By definition, $\mathbf{C}(P) \subseteq \mathbf{A}(P)$. Thus it holds that $D_A - D_C \geq 0$. Now, observe that any path $Q \in \mathbf{A}(P)$ is shorter than P. Since $\mathbf{A}(P)$ contains all paths in $\mathcal{G}$ that are shorter than P, the set of all paths in $\mathbf{A}(P)$ shorter than Q is exactly equal to that of all paths in $\mathcal{G}$ that are shorter than Q, which is, by definition, equal to $\mathbf{A}(Q)$. Hence,

$$\begin{aligned}
D_A &= \mathbb{P}\left(\bigcup_{Q\in\mathbf{A}(P)} \mathbf{X}(Q\setminus P)\right) \\
&= \sum_{Q\in\mathbf{A}(P)}\left[\mathbb{P}\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - \mathbb{P}\left(\bigcup_{R\in\mathbf{A}(Q)} \mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right]. \qquad (4.8)
\end{aligned}$$

By similar reasoning, the set of all paths in $\mathbf{C}(P)$ shorter than Q is exactly equal to $\mathbf{C}(Q)$. Hence,

$$\begin{aligned}
D_C &= \mathbb{P}\left(\bigcup_{Q\in\mathbf{C}(P)} \mathbf{X}(Q\setminus P)\right) \\
&= \sum_{Q\in\mathbf{C}(P)}\left[\mathbb{P}\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - \mathbb{P}\left(\bigcup_{R\in\mathbf{C}(Q)} \mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right] \\
&\geq \sum_{Q\in\mathbf{C}(P)}\left[\mathbb{P}\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - \mathbb{P}\left(\bigcup_{R\in\mathbf{A}(Q)} \mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right], \qquad (4.9)
\end{aligned}$$

where (4.9) follows because $\mathbf{C}(Q) \subseteq \mathbf{A}(Q)$ by definition. Note that (4.8) and (4.9) are summations of the same term across all paths Q in $\mathbf{A}(P)$ and $\mathbf{C}(P)$ respectively. Since $\mathbf{C}(P) \subseteq \mathbf{A}(P)$ and $\mathbf{A}(P)\setminus\mathbf{C}(P) = \mathbf{M}(P)$ by definition,

$$D_A - D_C \leq \sum_{Q\in\mathbf{M}(P)}\left[\mathbb{P}\big(\mathbf{X}(Q\setminus P)\big)\left\{1 - \mathbb{P}\left(\bigcup_{R\in\mathbf{A}(Q)} \mathbf{X}\big((R\setminus P)\setminus Q\big)\right)\right\}\right]. \qquad (4.10)$$

Plugging (4.10) into (4.7), and using $D_A - D_C \geq 0$ we have

$$\begin{aligned}
0 &\leq \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P)) \\
&= \mathbb{P}\big(\mathbf{X}(P)\big) \cdot \big(D_A - D_C\big) \\
&\leq \mathbb{P}\big(\mathbf{X}(P)\big) \sum_{Q \in \mathbf{M}(P)} \left[\mathbb{P}\big(\mathbf{X}(Q \setminus P)\big) \left\{1 - \mathbb{P}\left(\bigcup_{R \in \mathbf{A}(Q)} \mathbf{X}\big((R \setminus P) \setminus Q\big)\right)\right\}\right] \\
&= \sum_{Q \in \mathbf{M}(P)} \mathbb{P}(\mathtt{Sh}_s^t(Q) \wedge \mathbf{X}(P)) \\
&= \mathbf{p_m}\big(P, \mathbf{C}(P)\big),
\end{aligned}$$

where the second to last equality follows because the expression in the sum is the probability that Q exists and no path shorter than Q exists and the last equality follows by definition. This completes the proof. □

Note that from (4.4), we can say that for every s,t-path missing from CP (not returned in any run of Algorithm 5), it is highly likely that the probability of that path being the shortest s,t-path is extremely small. Thus, if P is the path returned by Algorithm 4, the sum of the shortest path probabilities of all paths shorter than P and missing from CP is also very small, and hence $\mathbf{p_m}\big(P, \mathbf{C}(P)\big)$, which also includes the condition that P exists, is even smaller.

Using Theorems 4.3.1 and 4.3.2, we can provide a quality guarantee for Algorithm 6 on a single path even with some shorter paths missing.

Theorem 4.3.3. *Consider an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, a source node s and a target node t, a set CP of s,t-paths, and a path $P \in CP$. Then, $\hat{p}\big(P, \mathbf{C}(P)\big)$ is an accurate estimate of $\mathbb{P}(\mathit{Sh}_s^t(P))$ with high probability. More formally, for all $\epsilon \in [0,1]$,*

$$\mathbb{P}\Big(\hat{p}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathit{Sh}_s^t(P)) - \mathbf{p_m}\big(P, \mathbf{C}(P)\big) \geq \epsilon\Big) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right), \quad (4.11)$$

$$\mathbb{P}\Big(\hat{p}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathit{Sh}_s^t(P)) \leq -\epsilon\Big) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right). \quad (4.12)$$

Proof: Note that $\mathbf{p_m}\big(P, \mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) + \mathbb{P}(\mathtt{Sh}_s^t(P))$ is at least 0 from Theorem 4.3.2; and is also at most 1 since $\mathbf{p_m}\big(P, \mathbf{C}(P)\big) \leq 1$, $\mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P)) \geq 0$. Thus $\epsilon + \mathbf{p_m}\big(P, \mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P, \mathbf{C}(P)\big) + \mathbb{P}(\mathtt{Sh}_s^t(P)) \in [0,2]$.

Applying Theorem 4.3.1 gives

$$\begin{aligned}
&\mathbb{P}\Big(\hat{p}\big(P,\mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P)) - \mathbf{p_m}\big(P,\mathbf{C}(P)\big) \geq \epsilon\Big) \\
&= \mathbb{P}\Big(\hat{p}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) \geq \epsilon + \mathbf{p_m}\big(P,\mathbf{C}(P)\big) \\
&- \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + \mathbb{P}(\mathtt{Sh}_s^t(P))\Big) \\
&\leq \exp\left(-\frac{N\Big(\epsilon + \mathbf{p_m}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + \mathbb{P}(\mathtt{Sh}_s^t(P))\Big)^2}{4|\mathbf{C}(P)|}\right) \\
&\leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right).
\end{aligned}$$

By a similar logic on $\mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P))$, Theorem 4.3.1 gives

$$\begin{aligned}
&\mathbb{P}\left(\hat{p}\big(P,\mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P)) \leq -\epsilon\right) \\
&= \mathbb{P}\Big(\hat{p}\big(P,\mathbf{C}(P)\big) - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) \leq -\epsilon - \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) + \mathbb{P}(\mathtt{Sh}_s^t(P))\Big) \\
&\leq \exp\left(-\frac{N\Big(\epsilon + \mathbf{p_{ne}}\big(P,\mathbf{C}(P)\big) - \mathbb{P}(\mathtt{Sh}_s^t(P))\Big)^2}{4|\mathbf{C}(P)|}\right) \leq \exp\left(-\frac{N\epsilon^2}{4|\mathbf{C}(P)|}\right),
\end{aligned}$$

which proves the theorem. □

We can now show that if the gap between the probability of the kth and the $(k+1)$st most probable shortest path is sufficiently big we return the top k paths with high probability.

Theorem 4.3.4. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, a source node s, a target node t, an integer k, let $P_1, \ldots, P_{k+1}$ denote the true top $k+1$ MPSPs (in order) from s to t. Then, $P_1, \ldots, P_k$ are indeed the paths returned by our method with high probability. Formally, let*

$$mid = \frac{1}{2}[\mathbb{P}(\mathtt{Sh}_s^t(P_k)) + \mathbb{P}(\mathtt{Sh}_s^t(P_{k+1})) + \mathbf{p_m}(P_{k+1}, \mathbf{C}(P_{k+1}))].$$

Further, let CP be the set of candidate paths returned by Algorithm 5 and define for each $P \in CP$:

$$d_P = \begin{cases} \mathbb{P}(\mathtt{Sh}_s^t(P)) - mid & \text{if } P \in \{P_1, \ldots, P_k\}, \\ mid - \mathbb{P}(\mathtt{Sh}_s^t(P)) - \mathbf{p_m}(P, \mathbf{C}(P)) & \text{otherwise,} \end{cases}$$

and assume that $d_P \geq 0$. This assumption is reasonable since, as noted earlier, $\mathbf{p_m}(P, \mathbf{C}(P))$ is very small. Then the following holds:

$$\mathbb{P}(\textit{Alg. 4 returns } P_1, \ldots, P_k) \geq \mathbb{P}(\{P_1, \ldots, P_k\} \subseteq CP) \prod_{P \in CP} \left[1 - \exp\left(-\frac{N d_P^2}{4|\mathbf{C}(P)|}\right)\right]$$

Proof: We compute

$$\begin{aligned}
&\mathbb{P}(\text{Alg. 4 returns } P_1, \ldots, P_k) \\
&\geq \mathbb{P}(\{P_1, \ldots, P_k\} \subseteq CP) \cdot \\
&\qquad \prod_{P \in \{P_1, \ldots, P_k\}} \mathbb{P}(\hat{p}(P, \mathbf{C}(P)) > mid) \cdot \prod_{P \in CP \setminus \{P_1, \ldots, P_k\}} \mathbb{P}(\hat{p}(P, \mathbf{C}(P)) < mid) \\
&= \mathbb{P}(\{P_1, \ldots, P_k\} \subseteq CP) \cdot \prod_{P \in \{P_1, \ldots, P_k\}} [1 - \mathbb{P}(\hat{p}(P, \mathbf{C}(P)) \leq mid)] \cdot \\
&\qquad \prod_{P \in CP \setminus \{P_1, \ldots, P_k\}} [1 - \mathbb{P}(\hat{p}(P, \mathbf{C}(P)) \geq mid)] \\
&= \mathbb{P}(\{P_1, \ldots, P_k\} \subseteq CP) \cdot \prod_{P \in \{P_1, \ldots, P_k\}} \left[1 - \mathbb{P}\left(\hat{p}(P, \mathbf{C}(P)) - \mathbb{P}(\mathtt{Sh}_s^t(P)) \leq -d_P\right)\right] \cdot \\
&\qquad \prod_{P \in CP \setminus \{P_1, \ldots, P_k\}} \left[1 - \mathbb{P}\left(\hat{p}(P, \mathbf{C}(P)) - \mathbb{P}(\mathtt{Sh}_s^t(P)) - \mathbf{p_m}(P, \mathbf{C}(P)) \geq d_P\right)\right] \\
&\geq \mathbb{P}(\{P_1, \ldots, P_k\} \subseteq CP) \prod_{P \in CP} \left[1 - \exp\left(-\frac{N d_P^2}{4|\mathbf{C}(P)|}\right)\right].
\end{aligned}$$

In the first inequality we make use of the fact that the scenario where Algorithm 6 returns a probability that is greater than mid for every path in $\{P_1, \ldots, P_k\}$ and lower than mid for every path in $CP \setminus \{P_1, \ldots, P_k\}$ is just a subset of the scenarios in which the algorithm returns $\{P_1, \ldots, P_k\}$. Because for every path the Monte Carlo rounds for estimating $\hat{p}(P, \mathbf{C})$ are independent we can split the probability in a product. In the last equality, we make use of the definition of d_P and in the last inequality we invoke Theorem 4.3.3. □

4.3.4 Extensions

Single-Source and Single-Target MPSPs

Our approach for generating the MPSP from a single source to a single target can be easily extended to compute MPSPs from a single source to all other nodes in the graph. Phase 1 continues running DIJKSTRA+MC on the entire graph until all edges are sampled, or no new target nodes can be reached. Phase 2 runs separately for each individual target (i.e., each source-target pair). A similar strategy can be applied for computing MPSPs to a single target from all other nodes: we need to use the same method on the graph with the edges reversed. Since Phase 1 is not run separately for each source-target pair, this helps to reduce the running time of this phase from $|V|$ times that of a single source-target pair to a smaller value. This is demonstrated empirically in Section 4.5.6.

Extension to Uncertain Multi-Graphs

An uncertain multi-graph is a quadruple $(V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, where V is a set of nodes and $E \subseteq V \times V$ is a set of directed edges with lengths (c_e) and probabilities of existence (p_e), such that every pair of nodes can be connected by zero, one, or more edges, called *parallel edges*, with a distinct combination of length and probability of existence. This more general data model can be used, e.g., for a probability distribution of travel times on a segment of a road network, depending on the traffic conditions.

Given a pair of nodes $(s, t) \in V \times V$, a (simple) path in an uncertain multi-graph is an ordered sequence of edges $(e_1, e_2, \ldots, e_n)$ where $e_i = (u_i, u_{i+1}, w_i, p_i) \in E$, $u_1 = s$, $u_{n+1} = t$ and $u_i \neq u_j$ for $i \neq j$. Our algorithm, described in Section 4.3.1, can be easily adapted to find MPSPs in uncertain multi-graphs. The main difference lies in the generation of the candidate paths. In Phase 1, when we reach a node in the uncertain graph, its outgoing edges are sampled with their respective probabilities, and only one sampled edge from the current node to each adjacent node (having the minimum length among all sampled edges from the current node to that adjacent node) is considered for updating the paths in line 8 of Algorithm 5.

4.4 MPSP-Betweenness Centrality

We define MPSP betweenness centrality (MPSP-BTW) in uncertain graphs and design an efficient sampling strategy to approximate the centrality of every node quickly, with theoretical guarantees.

In the preliminaries we have seen that in a deterministic directed graph $G = (V, E, (c_e)_{e\in E})$, the *betweenness centrality* of a node $v \in V$ is defined as

$$b_G(v) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(s,t)\in V\times V \\ s\neq v\neq t, \pi(s,t)\neq 0}} \frac{\pi_v(s,t)}{\pi(s,t)}, \tag{4.13}$$

where $\pi(s, t)$ denotes the number of shortest paths from s to t, and $\pi_v(s, t)$ the number of such paths P that contain v as an internal node.

In our work, we naturally extend this definition to betweenness centrality in an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$ for most probable shortest paths by replacing $\pi(s, t)$ with $|\mathcal{M}(\mathcal{G}, s, t)|$ (recall that $\mathcal{M}(\mathcal{G}, s, t)$ is the set of MPSPs from s to t) and $\pi_v(s, t)$ with $|\mathcal{M}_v(\mathcal{G}, s, t)|$, where $\mathcal{M}_v(\mathcal{G}, s, t)$ consists of the paths $P \in \mathcal{M}(\mathcal{G}, s, t)$ that have v as an internal node.

Definition 4.4.1. (MPSP-BTW). In an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e \in E}, (p_e)_{e \in E})$, we define the betweenness centrality of a node $v \in V$ based on most probable shortest paths as

$$b_{\mathcal{G}}(v) = \frac{1}{|V|(|V|-1)} \sum_{\substack{(s,t) \in V \times V \\ s \neq v \neq t, \mathcal{M}(\mathcal{G},s,t) \neq \emptyset}} \frac{|\mathcal{M}_v(\mathcal{G}, s, t)|}{|\mathcal{M}(\mathcal{G}, s, t)|}. \tag{4.14}$$

A different definition of betweenness centrality for uncertain graphs is given in (Pfeiffer and Neville, 2011; Wang and Lin, 2019) and it is referred to as *expected betweenness centrality* (EXP-BTW). The EXP-BTW of a node is the weighted average of its betweenness over all possible worlds:

$$\mathbb{E}_{G \sim \mathcal{G}}[b_G(v)] = \sum_{G \in \mathcal{G}} \mathbb{P}(G) b_G(v). \tag{4.15}$$

Either of these notions can be meaningful, depending on the application. For instance, the notion of EXP-BTW is worth studying when the application concerns the broadcasting of a message from one node to another, in which the message can be propagated over different possible paths. On the other hand, the notion of MPSP-BTW gives a more accurate picture when the application concerns routing or route recommendation, in which the path(s) need to be fixed beforehand and we can only use a single path to go from the origin to the destination.

In our experiments in Section 4.5.7, we see that these different notions of betweenness yield slightly different rankings when ordering the nodes based on their betweenness values. Moreover, by exploiting the results in Section 4.4, we are able to compute the MPSP-BTW much faster than the EXP-BTW.

Efficient s, t-pairs Sampling

The naive method of computing the MPSP-BTW of a node by considering all the s, t-pairs and then computing the MPSPs is infeasible for large uncertain graphs. Moreover, designing an efficient algorithm for this task is challenging in our setting. As observed in Section 4.2.1, in uncertain graphs, a sub-path of an MPSP is not necessarily an MPSP. Therefore, we cannot decompose a shortest path into two smaller shortest sub-paths or concatenate two shortest sub-paths to get a larger shortest path. For these reasons, we can neither apply optimization techniques such as those exploited in Brandes' algorithm (Brandes, 2001), nor apply techniques based on node sampling where a small set of nodes is sampled and their contributions to the betweenness centralities are accumulated to estimate the betweenness of other nodes (Bader et al., 2007; Chehreghani, 2014; Geisberger, Sanders, and Schultes, 2008).

Therefore, we design a novel algorithm based on an efficient s, t-path sampling strategy instead of node sampling. In the following, for simplicity, we assume that

Algorithm 7 Approximating MPSP-BTW

```
Input: Uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$, number of samples $r$, positive integers $m$ and $N$.
Output: $\hat{b}_{\mathcal{G}} : V \to \mathbb{R}$.
$\hat{b}_{\mathcal{G}}(v) \leftarrow 0\ \forall v \in V$.
for $i = 1$ to $r$ do
    Sample distinct nodes $s$ and $t$
    $P \leftarrow$ Algorithm 4($\mathcal{G}$, $s$, $t$, $m$, $N$)      /* Approximating the MPSP */
    for all $v \in Int(P)$ do
        $\hat{b}_{\mathcal{G}}(v) \leftarrow \hat{b}_{\mathcal{G}}(v) + \frac{1}{r}$
    end for
end for
return $\hat{b}_{\mathcal{G}}$
```

there is only one MPSP for every pair of nodes. Especially in complex networks, this is a reasonable assumption. Thanks to this assumption, choosing an MPSP uniformly at random is equivalent to finding the unique MPSP between them using Algorithm 4. However, if there are multiple MPSPs for a pair of nodes, we can identify all of them using our top-k approach in Section 4.3.2, and then select one among them uniformly at random.

Our proposed method, whose pseudocode is shown in Algorithm 7, samples r s,t-pairs, for each of which it computes the MPSP P and then increments the betweenness centrality of every internal node of P by $\frac{1}{r}$. The main question that now arises is: *How many samples are needed to produce an accurate estimate of the betweenness centrality of every node with high probability?* In the remainder of this section, we provide an answer to this question. Specifically, given $\epsilon, \delta > 0$, we find a lower bound on the number of samples r so that, with probability at least $1 - \delta$, the difference between the approximate and the exact centrality of every node is at most ϵ.

Following the ideas in (Mahmoody, Tsourakakis, and Upfal, 2016), we can obtain the following lower bound on the required number of samples.

Theorem 4.4.2. *Given an uncertain graph $\mathcal{G} = (V, E, (c_e)_{e\in E}, (p_e)_{e\in E})$ and $\epsilon, \delta > 0$, assuming that Algorithm 4 returns the correct MPSP and that there is a unique MPSP between every pair of nodes, the output of Algorithm 7 when using $r \geq \frac{1}{2\epsilon^2} \ln \frac{2|V|}{\delta}$ samples satisfies*

$$\mathbb{P}(|\hat{b}_{\mathcal{G}}(v) - b_{\mathcal{G}}(v)| < \epsilon\ \forall v \in V) > 1 - \delta.$$

Proof: For a fixed node $v \in V$, let $X_i(v)$ for $i = 1, \ldots, r$ be a random variable which is 1 if node v is an internal node of the MPSP of sample i and 0 otherwise. We will show that $\mathbb{E}[X_i(v)] = b_{\mathcal{G}}(v)$ for all i. The probability of sampling $s \neq t \in V$ is $\frac{1}{|V|(|V|-1)}$ and as we assumed there is a unique MPSP between every pair of nodes it holds that $|\mathcal{M}_v(\mathcal{G}, s, t)| \in \{0, 1\}$ depending on whether v lies on the MPSP from s to t or not. So,

$$\begin{aligned}\mathbb{E}[X_i(v)] &= \sum_{s \neq t} \frac{1}{|V|(|V|-1)} |\mathcal{M}_v(\mathcal{G}, s, t)| \\ &= \frac{1}{|V|(|V|-1)} \sum_{s \neq t : |\mathcal{M}(\mathcal{G},s,t)| \neq 0} \frac{|\mathcal{M}_v(\mathcal{G}, s, t)|}{|\mathcal{M}(\mathcal{G}, s, t)|} \\ &= b_{\mathcal{G}}(v).\end{aligned}$$

Notice that $\hat{b}_{\mathcal{G}}(v) = \frac{1}{r} \sum_{i=1}^{r} X_i(v)$ for all $v \in V$ and that the $X_i(v)$ for $i = 1, \ldots, r$ are independent and identically distributed. By linearity of expectation,

$$\mathbb{E}[\hat{b}_{\mathcal{G}}(v)] = \mathbb{E}\left[\frac{1}{r} \sum_{i=1}^{r} X_i(v)\right] = \frac{1}{r} \sum_{i=1}^{r} \mathbb{E}[X_i(v)] = \frac{1}{r} r \cdot b_{\mathcal{G}}(v) = b_{\mathcal{G}}(v).$$

Using the union bound over all nodes $v \in V$ and then Hoeffding's inequality[4], we obtain

$$\begin{aligned}\mathbb{P}(\exists v \in V : |\hat{b}_{\mathcal{G}}(v) - b_{\mathcal{G}}(v)| \geq \epsilon) &\leq \sum_{v \in V} \mathbb{P}(|\hat{b}_{\mathcal{G}}(v) - b_{\mathcal{G}}(v)| \geq \epsilon) \\ &\leq |V| \cdot 2 \exp\left(-2r\epsilon^2\right).\end{aligned}$$

Plugging in $r \geq \frac{1}{2\epsilon^2} \ln \frac{2|V|}{\delta}$ yields the result. □

Computational Complexity

The space and time complexities of Algorithm 7 are dominated by those of Algorithm 1 (line 4 of Algorithm 7). Hence, it follows from Section 4.3.1 that the complexities are: $\mathcal{O}(m|E| + |V|)$ and $\mathcal{O}(rm(|E| + |V| \log |V| + \log m))$, respectively.

Parallel Implementation

In Algorithm 7, the computations performed on the r sampled s, t-pairs are independent of each other. Hence, these computations can be implemented in parallel, e.g., via multiple threads. We experimentally demonstrate the effect of the number of threads on the running time of our algorithm in Section 4.5.7.

[4] If $X_1, X_2, \ldots, X_n$ are independent and identically distributed random variables each taking values in $[0, 1]$ and $\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$, it holds that $\mathbb{P}[|\bar{X} - \mathbb{E}[X]| \geq \epsilon] < 2e^{-2n\epsilon^2}$ (Hoeffding, 1963).

4.5 Experimental Results

We assess the efficiency and effectiveness of our proposal and compare it against previous work (Cheng, Yuan, Wang, et al., 2014; Zou, Peng, and Zhao, 2011) on synthetic networks (Section 4.5.2) and on road networks (Section 4.5.3). We also analyze the effect of each phase of our method on the performance (Section 4.5.4), parameter sensitivity analysis (Section 4.5.5), and single-source and single-target queries (Section 4.5.6). Finally, we present use cases on sensor networks (Section 4.5.7) and brain networks (Section 4.5.7), and application to computing network centrality (Section 4.5.7).

4.5.1 Experimental Setup

Experiments are conducted on a single core (except when testing our parallel implementation) of a server with a 3.7 GHz Xeon processor, and 256 GB RAM. Our C++ code is available at (Saha et al., 2020).

Queries

For each uncertain graph, we generate four categories of source-target pairs as queries. The first three categories constitute randomly chosen node pairs that are 2, 4, and 6 hops away. The last category comprises pairs of randomly chosen connected nodes. Under each category, we consider 100 different s, t-pairs. The result for each category is an average over these 100 queries.

Parameters

DIJKSTRA+MC-runs in Phase 1 (m): A small m is sufficient for our purpose (Section 4.3.3). We vary $m \in \{5, 10, 20, 50, 100\}$, with the default value 20.

# Monte Carlo-samples in Phase 2 (N): We vary $N \in \{10^1, 10^2, 10^3, 10^4, 10^5\}$, with the default value 10^3.

Top-k MPSPs: We vary $k \in \{1, 5, 10\}$, with the default value 1.

4.5.2 Results on Synthetic Networks

We generate synthetic, uncertain (directed) graphs according to two classic models. (i) The *Erdős-Rényi* (ER) model (Erdős and Rényi, 1959) generates a random graph with $|V|$ nodes and $|E|$ directed edges chosen uniformly at random from $|V|(|V|-1)$ possible edges; (ii) The *Barabási-Albert* (BA) model (Barabási and Albert, 1999) generates a graph with $|V|$ nodes and $|E|$ edges satisfying a power law (in)degree distribution. Starting with a single node and no edge, a new node is added in every time step along with $|E|/|V|$ edges directed from the new node to an existing node, such that the probability of choosing an existing node i (as target), with its current in-degree d_i, is proportional to d_i.

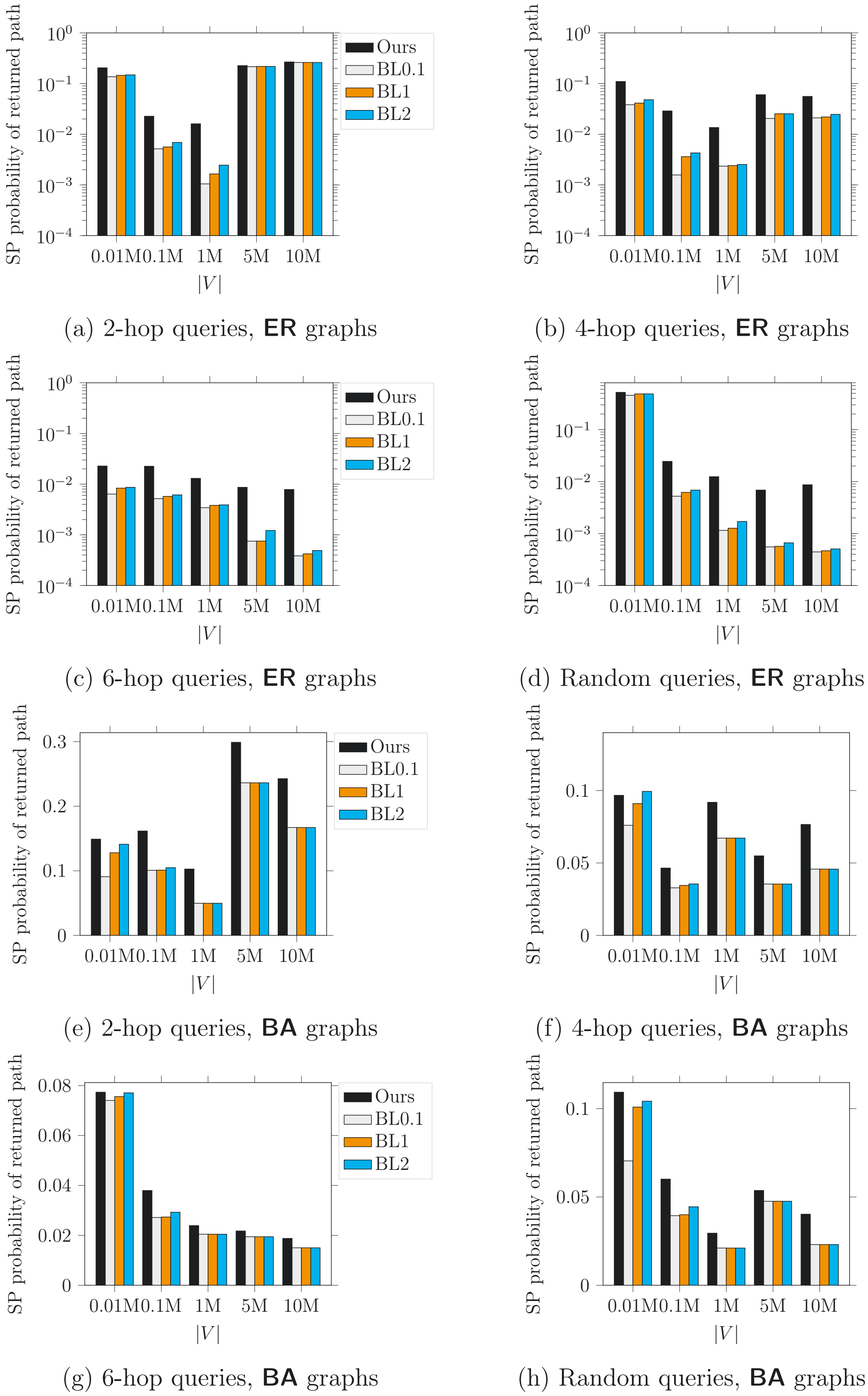
For both models, we vary $|V|$ in {0.01M, 0.1M, 1M, 5M, 10M}, and for every value of $|V|$, we vary the value of $|E|/|V|$ in {2, 6, 10}. In each synthetic graph, the probability of every edge is a uniform random number in the interval $(0, 1]$, and the length of every edge is a uniform random integer in the interval $(0, 1000]$.

Figure 4.4 reports quality comparison (expressed as the probability of the returned path being a shortest path) against the benchmark method (Zou, Peng, and Zhao, 2011). Given that the candidate generation phase (Section 4.2.2) of (Zou, Peng, and Zhao, 2011) does not finish in one hour for our synthetic datasets, to make the comparison feasible, we place an upper limit on the candidate generation time of (Zou, Peng, and Zhao, 2011). Notice that increasing this time limit leads to more candidate paths, hence the possibility of higher-quality returned paths is also increased. However, once an MPSP is included in the candidate set, increasing the time threshold further would not lead to better-quality solutions.

Following this observation, if T denotes the candidate generation time of our method for a given query, we compare the effectiveness of our algorithm against three variants of the benchmark, when we terminate the benchmarks candidate generation at time cT , with $c \in \{0.1, 1, 2\}$. We denote these three sets of benchmarks as $BL0.1$, $BL1$, $BL2$, as shown in Figure 4.4. Intuitively, $BL2$ could result in higher-quality returned paths compared to those via $BL0.1$ and $BL1$, however at the cost of higher running time, i.e., about 2-times higher running time than $BL1$ and 20-times more than $BL0.1$. $BL2$ is also about 2-times more time-consuming than ours.

Quality results in Figure 4.4 show that *in most cases our method outperforms all variants of the benchmark*. For 6-hop and random queries over larger ER graphs, the probability of being a shortest path of our solution is up to one order of magnitude better than those returned by the benchmarks (Figures 4.4(c) and 4.4(d))). Moreover, $BL2$ often results in a slightly higher probability of being the shortest path of the returned path compared to the other two variants of the benchmark: $BL0.1$ and $BL1$.

We show the efficiency of our method in Figure 4.5 for different query categories. Since the time limit of the benchmark is set by us, we do not compare its running time with that of ours. We observe that our running times are less sensitive to different query categories. However, the running times in **ER** graphs are some orders of magnitude larger than those in **BA** graphs. This can be attributed based on how these graphs are constructed: each node in **BA** graphs has an out-degree at most 10. On the other hand, in **ER** graphs, there are several nodes with out-degrees more than 15-20. This implies that Dijkstra's algorithm visits higher out-degree nodes a lot more in **ER** graphs, requiring longer running times.



(a) 2-hop queries, **ER** graphs

(b) 4-hop queries, **ER** graphs

(c) 6-hop queries, **ER** graphs

(d) Random queries, **ER** graphs

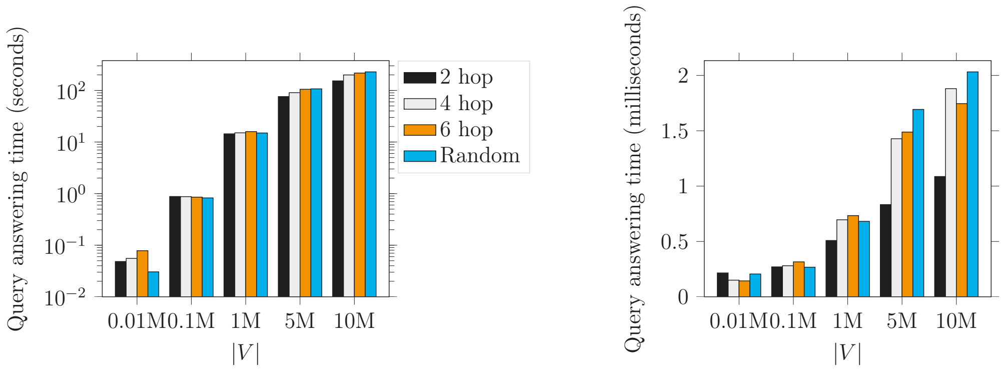
(e) 2-hop queries, **BA** graphs

(f) 4-hop queries, **BA** graphs

(g) 6-hop queries, **BA** graphs

(h) Random queries, **BA** graphs

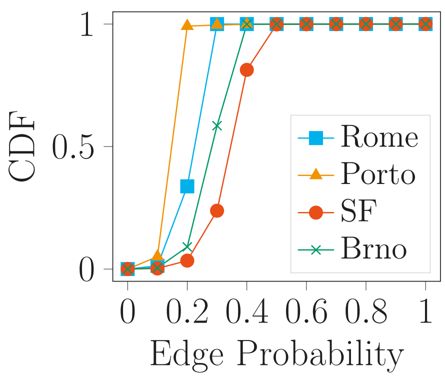
Figure 4.4: Quality comparison (expressed as the probability $\mathbb{P}(\mathtt{Sh}_s^t(P))$ of the returned path P of being a shortest path on synthetic graphs with $|E|/|V| = 10$



(a) **ER** graphs

(b) **BA** graphs

Figure 4.5: Running time on synthetic graphs with $|E|/|V| = 10$



City	$\|V\|$	$\|E\|$	Origin
Brno	1.9M	4.0M	[a]
Porto	1.8M	3.7M	[b]
Rome	4.0M	8.0M	[c]
SF	3.0M	6.2M	[d]

[a] (Ptošek, Rapant, and Martinovič, 2018)

[b] (Moreira-Matias et al., 2013)

[c] (Amici et al., 2014)

[d] (Piorkowski, Sarafijanovic-Djukic, and Grossglauser, 2009)

Figure 4.6: Properties of road networks

4.5.3 Results on Road Networks

We construct uncertain (directed) graphs from four real-world road networks obtained via OpenStreetMap (OpenStreetMap contributors, 2020), along with recorded taxi trajectory data for each network (see the table in Figure 4.6). The nodes denote locations, while the edges denote road segments. The length of an edge is measured as its spatial length. We *map-match* every trajectory to the corresponding map using the open-source software OSRM (Luxen and Vetter, 2011), obtaining the road segments involved in each trajectory along with the speed on each segment. However, there are road segments in each network that are not traversed by any trajectory. We synthetically assign a speed to each such segment following Costa et al. (2015), by sampling from a normal distribution with mean equal to the speed limit on that segment and standard deviation equal to a quarter of the mean. Since commuters are more likely to prefer those roads on which they can travel at a higher speed, we assign the probability of an edge (road segment) proportional to its average speed across all trajectories. The number of nodes, edges, and distribution of the edge probabilities in the resultant graphs are shown in Figure 4.6.

In our experiments on road networks, varying time thresholds for the benchmark does not result in any quality difference. This is because the road networks are sparse, and the MPSP is often the shortest path in the deterministic version of the network. Hence, we terminate the benchmarks candidate generation as soon as only the first s, t-path is obtained, which is essentially the shortest s, t-path considering the deterministic version of the network. We refer to this variant of the benchmark as *BL-1st-Path*.

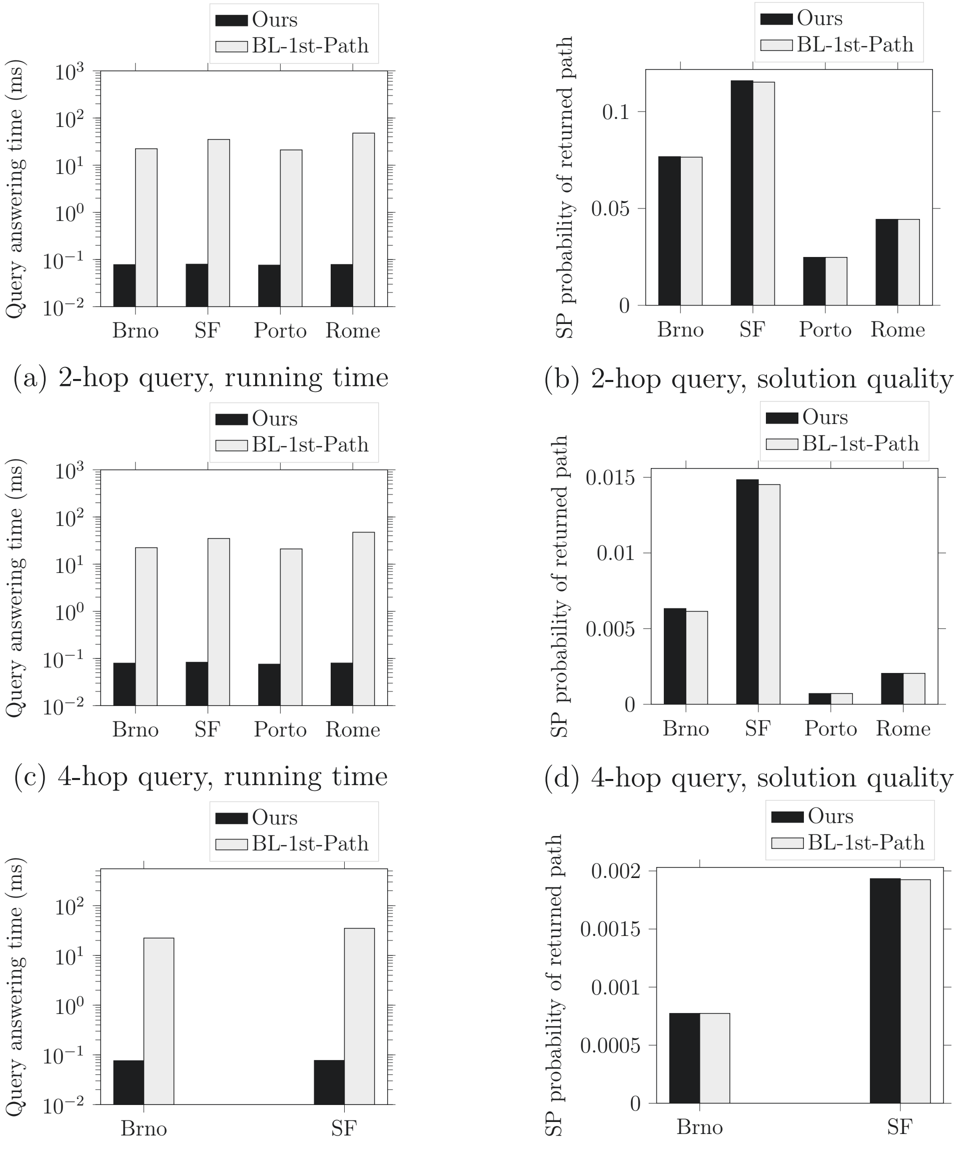
Figures 4.7(b), 4.7(d), and 4.7(f) compare the quality of our method with respect to the benchmark. Both methods return similar results in terms of quality. As stated earlier, the returned path (by both methods) for almost every query is also the shortest path in the certain version of the graph. Notice that the entries in the 6-hop query category are vacant for the *Porto* and *Rome* road networks. This implies that for these graphs, running DIJKSTRA+MC in the 6-hop category resulted in an empty path. This can be attributed to the fact that the edge probabilities of these graphs are smaller compared to those of the other graphs, which is evident from Figure 4.6. In general, due to sparseness of road networks and relatively smaller edge probabilities, MPSP queries are more meaningful here for nearby s, t-pairs (e.g., find the MPSP to the nearest gas station, or restaurant).

On the other hand, our method takes up to 2-3 orders of magnitude less time than the benchmark, as shown in Figures 4.7(a), 4.7(c), and 4.7(e). This is because the benchmark approach essentially uses Dijkstra's algorithm on the certain version of the graph to retrieve the shortest path, which has to visit every node closer to the source than the target. In contrast, DIJKSTRA+MC in our candidate generation may end up not visiting many nodes since the corresponding edges are not sampled.

4.5.4 Effect of Each Phase on the Performance

Our method (Section 4.3.1) consists of two phases: DIJKSTRA+MC for efficient candidate paths generation (Phase 1), followed by the Luby-Karp algorithm to select the MPSP among them (Phase 2). We analyze the benefits of both phases by comparing the two-phased method against a method based only on Phase 1, followed by a selection by majority, i.e., the path that has been sampled most times by DIJKSTRA+MC is returned as the candidate MPSP.

Table 4.1 shows that the two-phased method *never produces worse-quality results, and can return better MPSPs for up to 59% of the queries.* A possible explanation for this result is the following. Assume that there are two s, t-paths P_1 and P_2 such that the probability of P_1 of being the shortest path is slightly higher than that of P_2. Then, it could happen that P_2 is sampled a larger number of times (i.e., with a higher frequency) than P_1, due to the randomness of the



(a) 2-hop query, running time

(b) 2-hop query, solution quality

(c) 4-hop query, running time

(d) 4-hop query, solution quality

(e) 6-hop query, running time

(f) 6-hop query, solution quality

Figure 4.7: Results on road networks

Query type	% of queries our method finds better MPSPs	
	vs. Phase 1 + Majority	vs. Phase 1 + HT-estimator
2-hop	36%	12%
4-hop	59%	5%
Random	11%	6%

Table 4.1: Percentage of queries our method finds better MPSPs compared to (a) only Phase 1 of our method (DIJKSTRA+MC) followed by selection via majority, and (b) Phase 1 of our method followed by HT-estimator Cheng, Yuan, Wang, et al., 2014. ER graph with $|V| = 10^4, |E| = 10^5$.

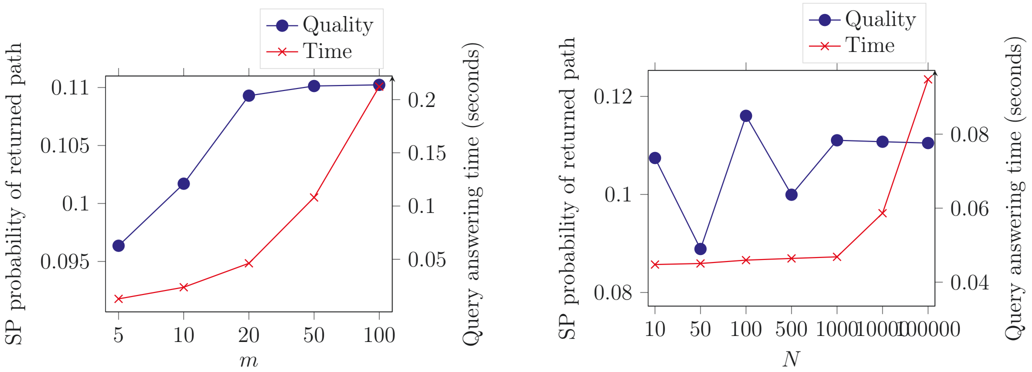
DIJKSTRA+MC sampling. Then, according to majority, the estimate for P_2 is higher than that of P_1. However, the Luby-Karp algorithm in our second phase does not care about the sampling frequency at all; it only needs to know if P_1 and P_2 are present in the sampled candidate set (at least once), thereby reporting the correct MPSP. These results demonstrate the usefulness of Phase 2.

We also compare with the case in which Phase 1 is augmented with an unequal probability estimator, e.g., Horvitz-Thompson (HT) inspired by Cheng, Yuan, Wang, et al. (2014). Recall (Section 4.1.2) that Cheng, Yuan, Wang, et al. (2014) deal with a different problem (i.e., threshold-based shortest-path queries) and adopts a different uncertain data model. However, their heuristic approach can be adapted for our purposes. Although the HT-estimator is useful in reducing the variance of DIJKSTRA+MC sampling, the Luby-Karp algorithm in our second phase still outperforms it, for the reason stated above. In particular, *our method never produces worse results and it produces better MPSPs for up to 12% of the queries.*

4.5.5 Parameter Sensitivity Analysis

Impact of m and N

We vary the number $m \in \{5, 10, 20, 50, 100\}$ of DIJKSTRA+MC runs (Phase 1), and the number of Monte Carlo samples $N \in \{10^1, 10^2, 10^3, 10^4, 10^5\}$ for the Luby-Karp algorithm (Phase 2). The results are shown in Figure 4.8. We show the results of 4-hop queries on the **ER** graph with $|V| = 10^4$ and $|E| = 10^5$, but other graph sizes showed similar results. For DIJKSTRA+MC, we observe that increasing m until its default value ($m = 20$) steadily increases the probability of the returned paths being the shortest path. This indicates that we need about $m = 20$ runs of DIJKSTRA+MC to include the MPSP in the candidate set. For the Luby-Karp algorithm, on the other hand, increasing N until its default value ($N = 10^3$) shows a fluctuation of the probabilities of being the shortest path that are returned, implying that the sampling method has not converged yet. The returned probabilities of being the shortest path stabilize around these default parameter values. We further notice that increasing these parameter values beyond their default values of $m = 20$ (resp. $N = 10^3$) returns probabilities of being the shortest path of paths (resp. Luby-Karp algorithm) having nearly the same value, but the running time is significantly increased. This justifies the selection of our default parameter values.



(a) Variation of m ($N = 10^3$) (b) Variation of N ($m = 20$)

Figure 4.8: Running times and quality of returned paths; **ER** graph with $|V| = 10^4, |E| = 10^5$; 4-hop queries.

Top-k MPSPs

We find the top-k MPSPs with $k \in \{1, 5, 10\}$. The results for $k = 1$ have already been shown in Figures 4.4 and 4.5. For $k \in \{5, 10\}$, the running times are nearly the same as that with $k = 1$; thus, we only show the probability of being the shortest path of our solution (averaged over the k paths returned for each query) in Figure 4.9. Notice that our algorithm returns better top-k paths compared to the benchmark.

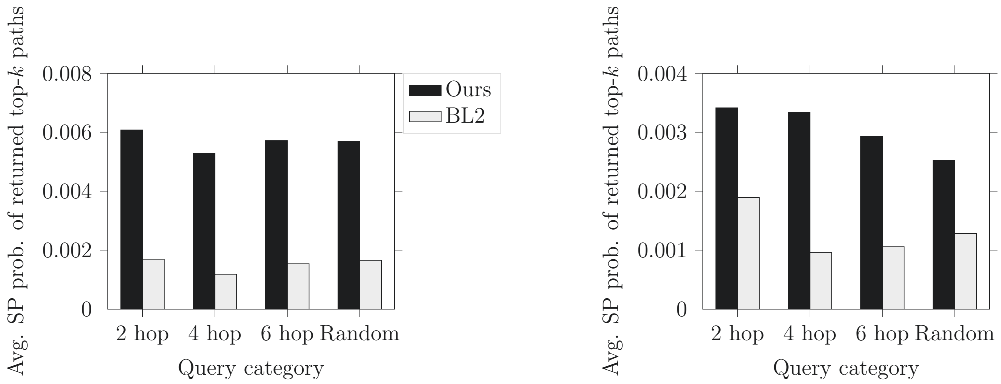


Figure 4.9: Quality of our solution for the top-k MPSPs; **ER** graph with $|V| = 10^5$, $|E| = 10^6$, $k = 5$ (left) and $k = 10$ (right)

4.5.6 Single-Source and Single-Target Queries

Figure 4.10 (left) shows the running times of single-source multi-target queries (Section 4.3.4) on **ER** graphs. The y-axis is logarithmic and the query answering time is the aggregated time required for both phases 1 and 2. We find that the running time of Phase 2 is much higher than that of Phase 1. However, the running time of Phase 1 is increased by a small factor because Dijkstra is not run separately for an individual target. Our Phase 1 is several orders of magnitude faster than *Phase 1-Naive*, which is running Phase 1 separately for each target node. Figure 4.10 (right) shows similar improved efficiency for multi-source single-target queries.

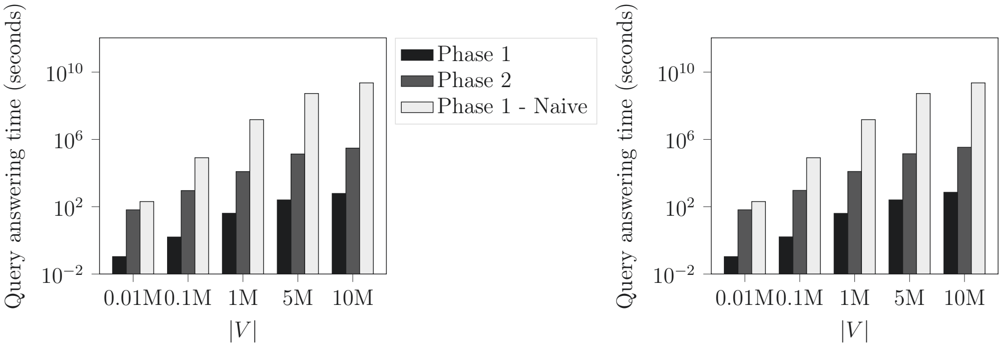


Figure 4.10: Running time of single-source (left) and single-target (right) queries; **ER** graphs with $|E|/|V| = 10$

4.5.7 Case Studies

Sensor Network

Intel Lab Data (Madden, 2004) is a collection of sensor communication data with 54 sensors deployed in the Intel Berkeley Research Lab between Feb. 28 and Apr. 5, 2004. The probabilities on (directed) edges denote the percentages of messages from a sender successfully reached to a receiver. The edge length is the spatial distance (in meters) between the coordinates of the two sensors.

We show MPSPs from node 48 to node 22 in Figure 4.11. We observe that the MPSP is the *sixth* shortest path in the certain version of the graph. The first few shortest paths have smaller probabilities of existence, showcasing the usefulness of MPSPs in uncertain graphs.

Brain Networks

A brain network can be defined as a weighted uncertain graph, where nodes are brain regions of interest (ROIs), (bi-directed) edges indicate co-activation between ROIs, edge distance represents the physical distance between ROIs, and edge probability indicates the strength of the co-activation signal (i.e., the pairwise Pearson correlation between the time series of each pair of ROIs). We use a publicly available dataset from the Autism Brain Imaging Data Exchange (ABIDE) project (Craddock et al., 2013). The dataset contains data of 52 *Typically Developed* (TD) children and 49 children suffering from *Autism Spectrum Disorder* (ASD) whose age is at most nine years (Kojaku and Masuda, 2019; Lanciano, Bonchi, and Gionis, 2020; Masuda, Kojaku, and Sano, 2018; Tzourio-Mazoyer et al., 2002): each subject corresponds to a graph of 116 nodes (ROIs).

We aggregate the information of the groups ASD and TD in two summary uncertain graphs $\mathcal{G}_{ASD}$ and $\mathcal{G}_{TD}$, respectively. $\mathcal{G}_{ASD}$ and $\mathcal{G}_{TD}$ are weighted uncertain graphs, defined over the same set of nodes as the original graphs, while the weight and probability of each edge are the average of the respective values of the same edge in every graph in the ASD and TD groups.

In Figures 4.12 and 4.13, we show the MPSPs for six s, t-pairs of both $\mathcal{G}_{TD}$ (left) and $\mathcal{G}_{ASD}$ (right). Consider the pink path in Figure 4.12 from the *inferior frontal gyrus, opercular part* (IFGoperc.L) to the *cerebellum* (CRBL1). The MPSP in $\mathcal{G}_{TD}$ is a path with two hops over a longer distance, compared to that in $\mathcal{G}_{ASD}$ with 6 shorter hops. This is consistent with the results of different works in neuroscience (Di Martino et al., 2010; Noonan, Haist, and Müller, 2009) indicating that ASD is characterized by underconnectivity between distant brain regions and overconnectivity between closer ones. Moreover, children with ASD have brains that are overly connected compared to typically developed children (García Domínguez et al., 2013; Keown et al., 2013; Supekar et al., 2013). In addition, the hemispheres in ASD group are more symmetrical than those of the TD group (Postema et al., 2019). We highlight this in Figure 4.13: the MPSPs in the left and right cerebral hemispheres of the brain are indeed more similar and symmetrical in children with autism, while in the TD group the paths can cross the hemispheres and also span the same regions. Our consistent findings underline the importance of MPSPs in uncertain graphs.

Network Centrality

We run experiments to demonstrate the application of MPSP in computing the betweenness centrality of nodes in an uncertain graph. We compare the top-k most central nodes according to three centrality computation methods already introduced in Section 4.4 and another centrality measure called PSP-BTW: (1) MPSP-BTW with sampled s, t-pairs, (2) MPSP-BTW with all s, t-pairs, (3) expected betweenness centrality (Pfeiffer and Neville, 2011; Wang and Lin, 2019) (by sampling possible worlds and using (Riondato and Kornaropoulos, 2016) for every sampled world) , and (4) PSP-BTW (Wang and Lin, 2019).

The notion of PSP-BTW was introduced as a fast approximation for EXP-BTW (Wang and Lin, 2019). It tries to approximate expected between-

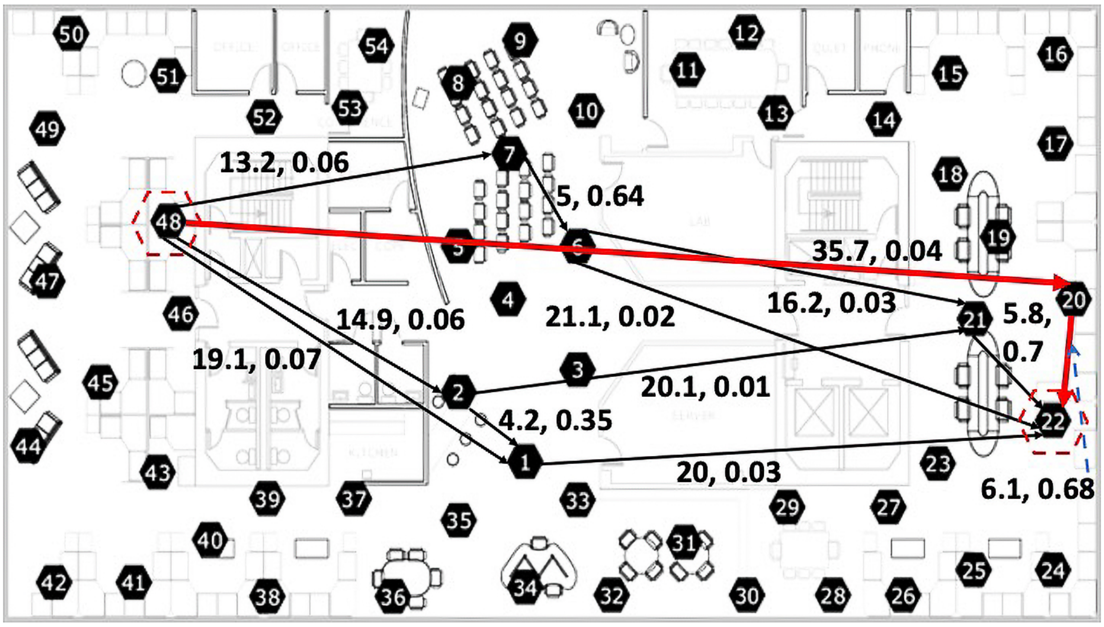


Figure 4.11: Case study on sensor network. Paths from node 48 to node 22 in the sensor network. The node sequences of the top 6 shortest paths (in ascending order of length) are $(48, 1, 22)$, $(48, 2, 1, 22)$, $(48, 7, 6, 22)$, $(48, 7, 6, 21, 22)$, $(48, 2, 21, 22)$, $(48, 20, 22)$. The 6^{th} shortest path, shown in red, is the MPSP.

ness by ignoring the correlation on the existence of paths. As this method is supposed to be efficient we include it to benchmark the efficiency of our algorithm.

We first run these methods on six different brain graphs (randomly selected from 52 TD brains), each with 116 nodes. Following Riondato and Kornaropoulos (2016) we set $\epsilon = 0.05$ and $\delta = 0.1$ for all the methods when required. For every method, we compute the betweenness centrality of all nodes and rank them in descending order of centrality. Given a value of $k \in \{10, 20, 50\}$, for each of the $\binom{4}{2} = 6$ possible pairs of methods, we compare the similarity of the sets of top-k nodes returned by both methods using the overlap coefficient. The overlap coefficient of two sets A and B, each of size k, is defined as $\frac{|A \cap B|}{k}$. We report these results averaged over six graphs in Figure 4.14(a). For every value of k, methods 1 and 2 (both deal with MPSP-BTW) produce very similar results showing that *our sampling based method yields good approximation.* The overlap with other methods is a bit lower indicating that there is a slight difference in the top-k nodes produced by each method.

Next, to assess the efficiency and scalability of our method, we compute the centrality ranking for the six brain graphs ($|V| = 116$), a Twitter graph ($|V| =$ 6.3M, $|E| =$ 11.1M), and the ER graphs with $|V| \in \{0.01M, 0.1M, 1M\}$ and $|E| = 10|V|$. Twitter (Leskovec and Krevl, 2014) is a social network where users post new tweets or retweet those of other users. This data is used to construct a

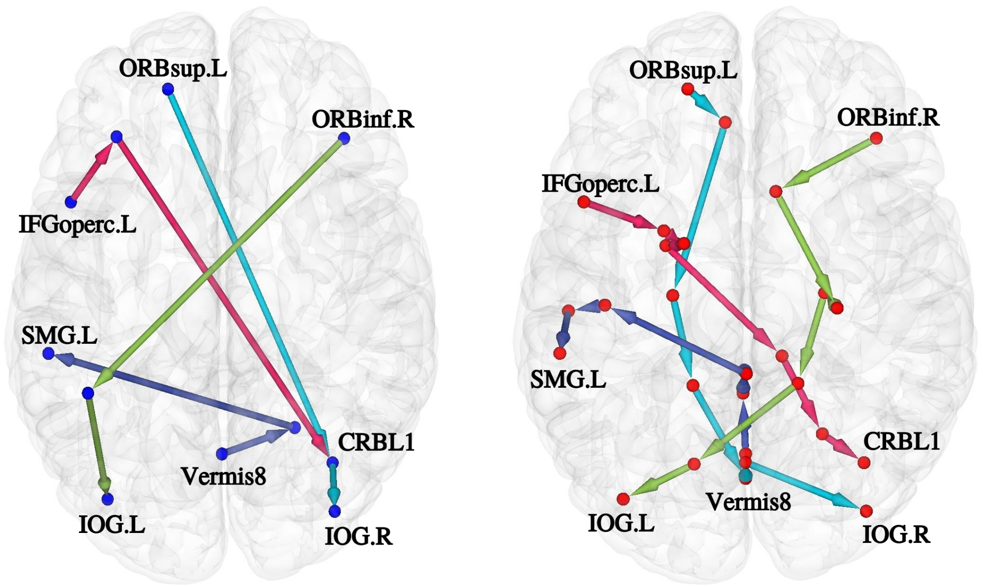


Figure 4.12: MPSPs for the TD group (left) and ASD group (right) of the brain networks. Each of the 4 MPSPs is represented by edges of the same color.

directed graph in which nodes are users and edges are retweets. Each edge has a weight of one, and the probability is given by: $1 - \exp(-t/\mu)$, where t is the number of retweets between the corresponding users. We set $\mu = 10$.

The sequential running times are shown in Figure 4.14(b). A missing bar means that the run did not terminate within a day. It turns out that *only our method (0) terminates within a reasonable time for all graphs.* Notice that even for the $1M$ node graph, our method finishes within 17 hours. Although the Twitter graph (6.3M nodes) is larger than the **ER** graph with 1M nodes, the running time on the Twitter graph is less than that on the **ER** graph, because the former is more sparse.

Finally, we run the parallel implementation of our method (method 1) on our two largest graphs: Twitter ($|V| = 6.3$M, $|E| = 11.1$M) and the **ER** graph with $|V| = 10$M and $|E| = 100$M. All 40 cores of the server are used and up to 40 threads are employed for parallelization via POSIX threads. Figure 4.14(c) shows that increasing the number of threads leads to shorter running times. *With 40 threads, centrality computation on Twitter ($|V| = 6.3M$, $|E| = 11.1M$) requires only 11 minutes, and on* **ER** *($|V| = 10M$, $|E| = 100M$), it finishes in 18 hours. These results demonstrate good parallelizability and scalability of our algorithm over large graphs.*

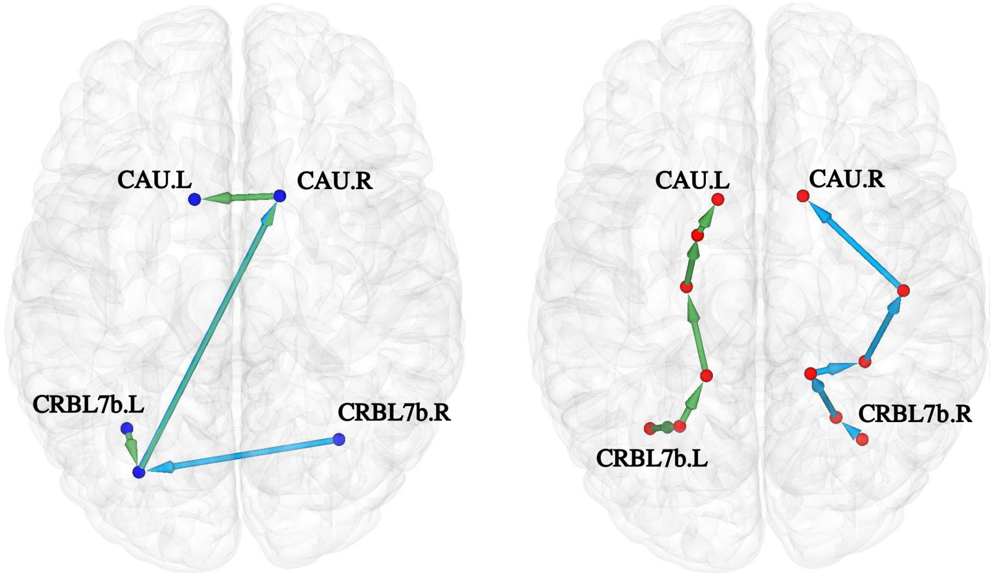
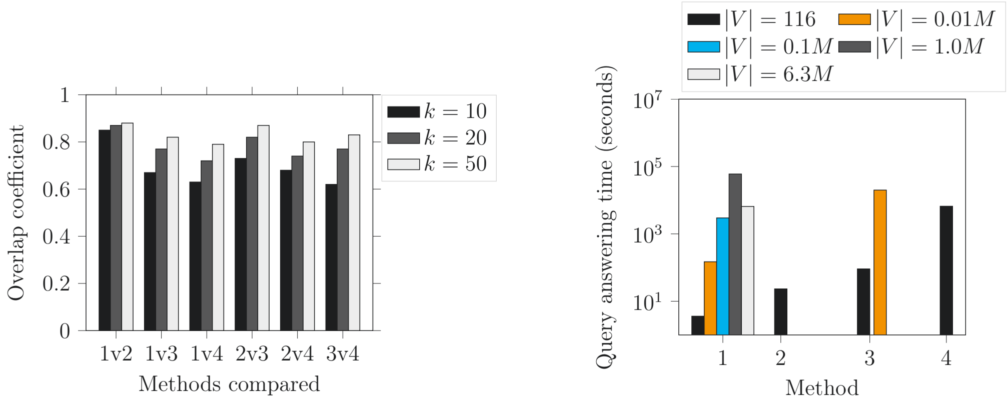


Figure 4.13: MPSPs for the TD group (left) and ASD group (right) of the brain networks. Each of the 2 MPSPs is represented by edges of the same colour.



(a) Top-k central node similarity

(b) Running time (sequential)

# Threads	Twitter, $|V| = 6.3\text{M}, |E| = 11.1\text{M}$	ER, $|V| = 10\text{M}, |E| = 100\text{M}$
1	6 520.65 sec	> 2 days
10	930.98 sec	> 2 days
20	795.91 sec	125 603.60 sec
40	666.76 sec	61 668.80 sec

(c) Parallelization: Our method (method 1)'s running time

Figure 4.14: Centrality results; 4 methods are described in Section 4.5.7

4.6 Conclusion

In this chapter, we investigated the problem of finding the Most Probable Shortest Path (MPSP) between two nodes in an uncertain graph. We proved that computing the probability that a path is the shortest path is #**P**-hard, and also derived some other properties of MPSPs that make our problem challenging. Our proposed solution proceeds in two phases: sampling of candidate paths using DIJKSTRA+MC, followed by approximating the probability of being the shortest path of each candidate path using the Luby-Karp algorithm. We proved probabilistic guarantees on our algorithm returning the correct path and have extended the algorithm to find the top-k MPSPs. Based on the notion of MPSP we defined a new centrality measure and proposed an algorithm for efficiently approximating it. Finally, we did extensive experiments on both synthetic and real-world networks to validate the performance of our algorithm and the usefulness of MPSPs.

The previous state-of-the-art (Zou, Peng, and Zhao, 2011) took the edge lengths as a basis for candidate generation, while we have immediately involved the edge probabilities in this phase of the algorithm. This change of perspective has yielded promising results, and we hope it serves as inspiration for future work. Where we have kept the value of m constant throughout experiments, it seems that a more extensive candidate generation could be useful for bigger graphs. It is an interesting question if the best m can be determined dynamically based on the input.

Finally, our algorithm consists of two phases, and in both phases, a Monte Carlo algorithm is used. In phase 1, one can increase m to make the probability that the top-k paths are not all included in the candidate set arbitrarily small. In phase 2, one can increase N to make the probability that the algorithm does not return the top-k paths arbitrarily small. That the combined algorithm is also a Monte Carlo algorithm could be made more concrete by showing what is needed in Theorem 4.3.4 to make sure $d_P \geq 0$.

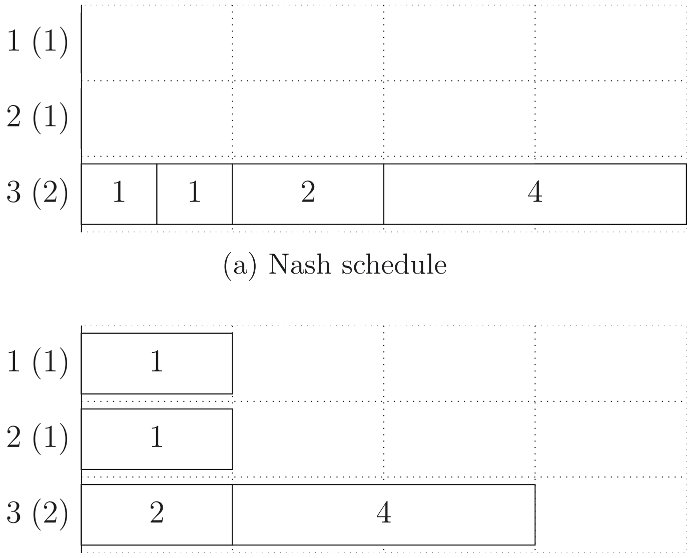
Chapter 5

The Price of Anarchy of Related Machine Scheduling

5.1 Introduction

Consider the situation in which there are more tasks than processors on a supercomputer. Or more planes than runways in an airport. Or more jobs in a job shop than there are machines. In each of the situations, we have to make a schedule such that all activities are allocated over the limited resources. Supercomputers consist of various types of processors, and on a faster processor, tasks are completed faster. We are interested in this setting, in which the problem consists of scheduling a set of jobs, each having a processing time, on a set of machines, each having a speed. The time it takes to process a job is its processing time divided by the speed of the machine it is scheduled on, and a job starts being processed after the jobs in front of it have been processed. Each job will have a completion time and we are interested in minimizing the sum of completion times. Dividing by the number of jobs gives the average time a job is in the system, which is why this quantity is also called the *mean flow time*. Using the 3-field scheduling notation (Graham, Lawler, et al., 1979) this problem is denoted by $Q||\sum C_j$. An algorithm that produces decent but not necessarily optimal solutions is the *Ibarra-Kim algorithm* (Ibarra and Kim, 1977). It sorts the jobs from smallest to largest and greedily schedules the next job on the machine that minimizes its completion time.

In the traditional setting, a central decision-maker has full control over the jobs. However, with the rise of distributed systems, there is not always a central decision-maker. In fact, there are multiple parties with access to the same system, and each of them is interested in optimizing their own objective. In machine scheduling, we can model this as follows. Suppose each job is controlled by a different player. Each player wants to minimize the completion time of their job.



(a) Nash schedule

(b) Optimal schedule

Figure 5.1: Machine scheduling example

On a single machine, the total completion time is minimized when scheduling jobs in non-decreasing order (more details below). Given that this is the local policy, players only need to select which machine will process their job. We are interested in the maximal ratio of the sum of completion times when players are in a Nash equilibrium and the optimal sum of completion times. In other words, what is the *price of anarchy* of this game?

Since Koutsoupias and Papadimitriou (1999) introduced this notion, the study of the price of anarchy in all types of settings has been a growing field of research. Examples of settings include routing in networks, scheduling, and auctions (Roughgarden, Syrgkanis, and Tardos, 2017). It is now a standard way of quantifying the inefficiency caused by selfish behavior.

We take a look at an example to get more intuition.

Example 5.1.1. Consider an instance with three machines and four jobs. Two machines have a speed of 1, and one machine has a speed of 2. There are two jobs with processing time 1, a job with processing time 2, and a job with processing time 4. In Figure 5.1(a) we have depicted a Nash schedule (or a schedule produced by a run of the Ibarra-Kim greedy algorithm) and in Figure 5.1(b) an optimal schedule. The three rows indicate the machines, where the machine number and speed of the machine (within parentheses) are written on the left. The rectangles represent jobs, and the numbers inside indicate their processing time.

In the Nash schedule, the first job clearly prefers the fast machine as it will have a completion time of $\frac{1}{2}$. The other three jobs would have the same completion time if they switched to another machine and therefore have no incentive to deviate.

The total completion time of the Nash schedule is

$$\frac{1}{2}+\frac{1+1}{2}+\frac{1+1+2}{2}+\frac{1+1+2+4}{2}=\frac{15}{2}.$$

As we will see in Section 5.2.1 the schedule in Figure 5.1(b) is a schedule minimizing the sum of completion times. The total completion time of this schedule is

$$\frac{1}{1}+\frac{1}{1}+\frac{2}{2}+\frac{2+6}{2}=6.$$

The Nash schedule in Figure 5.1 can be verified to be the Nash schedule with the highest sum of completion times, and so the price of anarchy of this instance is

$$\frac{\frac{15}{2}}{6}=\frac{5}{4}.$$

It can be shown that the approximation factor of the Ibarra-Kim algorithm and the pure price of anarchy of the related machine scheduling game are actually the same and it is somewhere in between $\frac{e}{e-1}$ and 2 (Hoeksma and Uetz, 2019). Studying one is therefore equivalent to studying the other but from a different perspective. The upper bound of 2 is proved using a variant on the smoothness technique discussed in the preliminaries called *semi-smoothness*. As with smoothness, bounds proved by semi-smoothness extend all the way to coarse correlated equilibria. If we restrict our scheduling instances to instances where all machines have the same speed (without loss of generality equal to one), it is known that the pure price of anarchy is 1 (Conway, Maxwell, and Miller, 1967), while the robust price of anarchy is $\frac{3}{2}-\frac{1}{2m}$, where m is the number of machines, and this is tight for mixed Nash equilibria (Rahn and Schäfer, 2013). If the separation between the pure price of anarchy and the mixed price of anarchy were also there for related machine scheduling, then the semi-smoothness technique is not strong enough to give a tight bound on the pure price of anarchy, and we need other tools.

A technique that is able to capture the specifics of pure Nash equilibria is a primal-dual method developed by Bilò (2018). First, it tries to capture finding the worst-case Nash equilibria into a linear program. Secondly, it leverages strong duality by finding a feasible solution for the dual program which gives a bound on the value of the primal program. This in turn also gives a bound on the pure price of anarchy.

5.1.1 Our Contributions

This chapter contains the following contributions:

1. We outline how the primal-dual method can be used in the context of related machine scheduling and show, for example, how we can easily reobtain the bound of 2 by exemplifying a feasible dual solution.

2. We show how our first attempt is not strong enough to improve the upper bound as it cannot prove a pure price of anarchy of 1 for identical machine scheduling and $\frac{e}{e-1}$ for related machine scheduling.

3. But, we also give a stronger program which gives more flexibility for proving better bounds. We use it to prove that the processing times in the Hoeksma-Uetz lower bound instance are, in fact, worst-case with respect to the Nash and optimal schedules.

4. The $\frac{e}{e-1}$ lower bound of Hoeksma and Uetz (2019) is for instances where the number of machines and jobs goes to infinity. For a fixed number of jobs n and machines m we give a procedure to find instances with a slightly improved lower bound compared to what was previously known in the literature.

The initial goal of this research was to prove that the pure price of anarchy for related machine scheduling is $\frac{e}{e-1}$ or find evidence that it is not. Unfortunately, we have not succeeded, but we hope that this chapter gives more insights and contributes as a 'push in the back' for future work.

5.1.2 Related Work

There is a lot of work done on scheduling games in which the social cost is equal to the *makespan*, i.e., the latest moment a job finishes, of the schedule. These games are also called *load balancing games*. When Koutsoupias and Papadimitriou introduced the *coordination ratio*, now known as the price of anarchy, they looked at a game that can be seen as a scheduling game while minimizing makespan (Koutsoupias and Papadimitriou, 1999). For various variants of the game, tight bounds on the price of anarchy have been obtained (Awerbuch et al., 2006; Czumaj and Vöcking, 2007; Gairing et al., 2010).

For the sum of completion times as social cost on related machines, the best-known upper bound on the price of anarchy is 2 (Hoeksma and Uetz, 2019). If we fix the number of jobs to n and machines to m this bound can be slightly improved to $2 - \frac{2}{(n+m)(n+1)}$ (Zhang et al., 2019). A lower bound on the price of anarchy when allowing instances of any size is $\frac{e}{e-1}$ (Hoeksma and Uetz, 2019). For two machines, the best-known lower bound is 1.875... (Zhang et al., 2019).

In the related machine setting there is still a gap between $\frac{e}{e-1}$ and 2. But, for unrelated machines, where each job has a possibly different processing time per machine, the price of anarchy is 4 and this is tight (Cole et al., 2015; Correa and Queyranne, 2012). For identical machines, the pure price of anarchy is 1, while the robust price of anarchy is $\frac{3}{2} - \frac{1}{2m}$ and this is tight for mixed Nash equilibria (Rahn and Schäfer, 2013).

The primal-dual method by Bilò (2018) has been especially powerful when applied to congestion games. Writing down a linear program, finding a fitting for the dual, and some calculus results in the best-known bounds for various types of cost functions. It is mainly used for bounding the pure price of anarchy, but there are settings in which an extension theorem can lift this bound up to coarse-correlated equilibria.

5.2 Preliminaries

In related machine scheduling, we have a set of n jobs $N = [n]$ and a set of m machines $M = [m]$. Each job $j \in N$ has a *processing time* p_j and each machine $i \in M$ has a *speed* s_i. We assume, without loss of generality, that $p_1 \leq p_2 \leq \ldots \leq p_n$ and $s_1 \leq s_2 \leq \ldots \leq s_n$. The time it takes to process job j on machine i is $\frac{p_j}{s_i}$. A *schedule* consists of an assignment $\boldsymbol{x} \in [m]^n$ of every job to a machine. On a single machine, jobs are scheduled in non-decreasing order of processing times and ties are broken consistently on index. This order is called *shortest processing time first (SPT).*

Given a schedule $\boldsymbol{x}$ the *completion time* of a job is given by

$$C_j(\boldsymbol{x}) = \sum_{\substack{k \in N \\ k \leq j \\ x_k = x_j}} \frac{p_k}{s_{x_k}} = \frac{1}{s_{x_j}} \sum_{\substack{k \in N \\ k \leq j \\ x_k = x_j}} p_k.$$

If it is clear that k iterates over N we do not always write it explicitly. The sum of completion times (also called the *mean flow time*) is then

$$SC(\boldsymbol{x}) = \sum_{j \in N} C_j(\boldsymbol{x}) = \sum_{j \in N} \sum_{\substack{k \leq j \\ x_k = x_j}} \frac{p_k}{s_{x_k}}.$$

For a schedule $\boldsymbol{x}$, a machine i and a job j let

$$z_i(\boldsymbol{x}, j) = |\{k \in N | k > j, x_k = i\}| \tag{5.1}$$

be the number of jobs that are processed after job j under $\boldsymbol{x}$ if job j would be scheduled on machine i. Observe that we can rewrite the sum of completion times as

$$SC(\boldsymbol{x}) = \sum_{j \in N} \frac{z_{x_j}(\boldsymbol{x}, j) + 1}{s_{x_j}} \cdot p_j.$$

Every job j contributes $\frac{p_j}{s_{x_j}}$ for itself and once for every job that is scheduled after it to the sum of completion times.

5.2.1 Optimal Schedule

Suppose we have an instance with only one machine. Without loss of generality assume that the speed of this machine is one. Assume the jobs are processed in

Algorithm 8 MFT Algorithm

```
Input: $n$ jobs with processing times $p_1 \leq \ldots \leq p_n$, $m$ machines with speeds
    $s_1, \ldots, s_m$
Output: A schedule $\boldsymbol{x}^*$ minimizing the sum of completion times
1: $z_i \leftarrow 0$ for all $i \in [m]$
2: for $j = n$ to 1 do              /* From longest to shortest processing time */
3:         Assign job $j$ to a machine $i$ with minimal $\frac{z_i+1}{s_i}$
4:         $x_j^* \leftarrow i$
5:         $z_i \leftarrow z_i + 1$
6: end for
7: return $\boldsymbol{x}^*$
```

order of a permutation $\pi : [n] \to [n]$, i.e., job j is the $\pi(j)$th job being processed. The sum of completion times with respect to π is

$$\sum_{j \in N} \sum_{\substack{k \in N \\ \pi(k) \leq \pi(j)}} p_k = \sum_{j \in N} (n + 1 - \pi(j)) p_j.$$

This expression is minimized when the processing time of the job with the ith longest processing time is matched to the ith smallest coefficient. This is exactly the result when scheduling the jobs in SPT order.

The idea of matching the highest coefficient to the shortest job can also be extended to multiple machines. For m machines we can compute nm coefficients: for each machine i, compute $\frac{1}{s_i}, \frac{2}{s_i}, \ldots, \frac{n}{s_i}$. If job j is assigned to the coefficient $\frac{k}{s_i}$ that means it will be scheduled on machine i and that there are still $k - 1$ jobs coming after it so that it contributes $\frac{k}{s_i} \cdot p_j$ to the sum of completion times. To minimize the sum of completion times, the jobs with the largest processing times need to be matched to the smallest coefficients. More specifically, the ith smallest coefficient needs to be matched to the job with the ith longest processing time. The *Mean Flowtime Algorithm (MFT)* algorithm (Algorithm 8) does exactly this (Horowitz and Sahni, 1976). Note that if there are coefficients with the same value, then the jobs matched to these coefficients can be permuted without changing the contribution to the sum of completion times. It is not hard to see that based on the way ties are broken (in line 3 of Algorithm 8) the MFT algorithm can produce *any* optimal schedule.

Theorem 5.2.1. ((Horowitz and Sahni, 1976)). *The Mean Flow Time Algorithm produces an optimal schedule for related machine scheduling and any optimal schedule can be produced by the MFT algorithm with the proper tie-breaking rule.*

5.2.2 Related Machine Scheduling Game

We can analyze related machine scheduling also in a game-theoretic context. A related machine scheduling game Γ consists of n players, each player $j \in N = [n]$ controls a job j with processing time p_j. The strategy set for every player is equal to M, i.e., the players can assign their job to exactly one of the machines. The cost of a player in a strategy profile $\boldsymbol{x} \in M^N$, in which player j has chosen machine x_j, is the completion time of their job

$$c_j(\boldsymbol{x}) = C_j(\boldsymbol{x}).$$

The social cost we want to minimize is the sum of completion times

$$SC(\boldsymbol{x}) = \sum_{j \in N} C_j(\boldsymbol{x}) = \sum_{j \in N} \sum_{\substack{k \leq j \\ x_k = x_j}} \frac{p_k}{s_{x_k}}.$$

In this context a pure Nash equilibrium is a strategy profile $\boldsymbol{x} \in M^N$ that satisfies for every player $j \in N$

$$\frac{1}{s_{x_j}} \sum_{\substack{k \leq j \\ x_k = x_j}} p_k = c_j(\boldsymbol{x}) \leq c_j(i, \boldsymbol{x}_{-j}) = \frac{1}{s_i} \left(p_j + \sum_{\substack{k < j \\ x_k = i}} \right) \qquad \text{for all } i \in M,$$

and the pure price of anarchy is defined as

$$\mathrm{PoA}(\Gamma) = \sup_{\boldsymbol{x} \in \mathrm{PNE}(\Gamma)} \frac{SC(\boldsymbol{x})}{SC(\boldsymbol{x}^*)},$$

where $\boldsymbol{x}^* \in M^N$ is a strategy profile minimizing $SC(\boldsymbol{x}^*)$.

Because on every machine the jobs are processed in order of index, the cost of player j only depends on the strategies of players 1 up to j. The *Ibarra-Kim algorithm* is an approximation algorithm for the optimization version of the related machine scheduling problem that schedules jobs in non-decreasing order of processing times, assigning each job to the machine on which it will have the minimal completion time. Also here, the completion time of job j only depends on the schedule of the first j jobs. Just like the MFT algorithm can produce all optimal schedules based on how ties are broken, the Ibarra-Kim algorithm (Algorithm 9) can produce all Nash schedules (Ibarra and Kim, 1977) based on how ties are broken (in line 3 of Algorithm 9).

Theorem 5.2.2. (Heydenreich, Müller, and Uetz, 2007; Immorlica et al., 2009). *The set of pure Nash equilibria for the related machine scheduling game is precisely the set of solutions that can be generated by the Ibarra-Kim algorithm depending on how ties are broken*

Studying the price of anarchy of the related machine scheduling game is therefore equivalent to studying the approximation factor of the Ibarra-Kim algorithm for related machine scheduling.

Algorithm 9 Ibarra-Kim algorithm

```
Input: $n$ jobs with processing times $p_1 \leq \ldots \leq p_n$, $m$ machines with speeds $s_1, \ldots, s_m$
Output: A Nash schedule $\boldsymbol{x}$
$\ell_i \leftarrow 0$ for all $i \in [m]$
for $j = 1$ to $n$ do
    Assign job $j$ to a machine $i$ with minimal $\ell_i + \frac{p_j}{s_i}$
    $x_j \leftarrow i$
    $\ell_i \leftarrow \ell_i + \frac{p_j}{s_i}$
end for
return $\boldsymbol{x}$
```

5.3 Best-Known Bounds

The best-known bounds on the price of anarchy for related machine scheduling with SPT as local scheduling policy are proven using the smoothness framework (Roughgarden, 2015) which we already introduced in the preliminaries.

The notion can sometimes be relaxed a bit without losing on its implications. One way is called *semi-smoothness* and it makes specific use of properties of an optimal solution (Caragiannis, Kaklamanis, Kanellopoulos, Kyropoulou, Lucier, et al., 2015; Lucier and Paes Leme, 2011).

Definition 5.3.1. A cost-minimization game is called (λ, μ)-*semi-smooth* with $\lambda \geq 0, \mu < 1$ if there exists a (mixed) strategy profile $\boldsymbol{\sigma}$ such that for an optimal strategy profile $\boldsymbol{x}^*$ and any other strategy profile $\boldsymbol{x}$ it holds that

$$\mathbb{E}\left[\sum_{j \in N} c_j(\sigma_j, \boldsymbol{x}_{-j})\right] \leq \lambda \cdot C(\boldsymbol{x}^*) + \mu \cdot C(\boldsymbol{x}).$$

Just like with normal smoothness, if a game is (λ, μ)-semi-smooth we get a bound of $\frac{\lambda}{1-\mu}$ on the price of anarchy which extends all the way to coarse correlated equilibria.

5.3.1 Upper Bounds

Using semi-smoothness Hoeksma and Uetz (2019) proved an upper bound on the robust price of anarchy of 2. As $\boldsymbol{\sigma}$ they take an optimal solution $\boldsymbol{x}^*$ and show that

$$\sum_{j \in N} c_j(x_j^*, \boldsymbol{x}_{-j}) \leq 2 \cdot C(\boldsymbol{x}^*).$$

They rewrite the left-hand side as

$$\sum_{j \in N} c_j(x_j^*, \boldsymbol{x}_{-j}) = \sum_{j \in N} \frac{p_j}{s_{x_j^*}} + \sum_{j \in N} z_{x_j}(\boldsymbol{x}^*, j) \cdot \frac{p_j}{s_{x_j}}. \tag{5.2}$$

First, we note that the first term in (5.2) is upper bounded by $C(\boldsymbol{x}^*)$ as the completion time of a job is at least the time it takes to process it (plus jobs that may be processed before it).

To bound the second term Hoeksma and Uetz (2019) give the following characterization of optimal strategy profiles.

Theorem 5.3.2. ((Hoeksma and Uetz, 2019)). *A schedule $\boldsymbol{x}^*$ is optimal if and only if*

$$\frac{z_\ell(\boldsymbol{x}^*, j)}{s_\ell} \leq \frac{z_i(\boldsymbol{x}^*, j) + 1}{s_i} \qquad \textit{for all } i, \ell.$$

Using this theorem the second term in (5.2) is bounded by

$$\sum_{j \in N} z_{x_j}(\boldsymbol{x}^*, j) \cdot \frac{p_j}{s_{x_j}} \leq \sum_{j \in N} \left(z_{x_j^*}(\boldsymbol{x}^*, j) + 1 \right) \cdot \frac{p_j}{s_{x_j^*}} = C(\boldsymbol{x}^*). \tag{5.3}$$

So, (5.2) is upper bounded by $2 \cdot C(\boldsymbol{x}^*)$.

Later Zhang et al. (2019) improved this bound slightly by observing that for the nth term the bound in (5.3) is too crude as $z_i(\boldsymbol{x}^*, n) = 0$ for all machines i. As $\frac{p_n}{s_{x_n^*}} \geq \frac{2}{(n+m)(n+1)} C(\boldsymbol{x}^*)$ the better bound of $2 - \frac{2}{(n+m)(n+1)}$ follows.

5.3.2 Lower Bound

The best-known lower bound is given by Hoeksma and Uetz (2019).

Example 5.3.3. (Hoeksma-Uetz instances). The instances are parametrized by two integers m and s. An instance consists of m machines of which one has speed s while the rest has speed 1. The instance has $n = m + s - 1$ jobs. The processing time of job j is defined as

$$p_j = \begin{cases} 1 & \text{if } j \leq s - 1 \\ \left(\frac{s}{s-1}\right)^{j-s} & \text{if } j \geq s. \end{cases}$$

Observe that for $j = s$ the processing time is actually 1, so we also could have put the transition between s to $s + 1$.

Scheduling all jobs on the fastest machine is a Nash equilibrium. In an optimal schedule, the first $m - 1$ jobs are scheduled alone on a slow machine and the remaining jobs are scheduled on the fast machine. An example with $m = 5, s = 3, n = 7$ can be found in Figure 5.2.

The Nash schedule has a total completion time of $(s-1)\left(\left(\frac{s}{s-1}\right)^m - \frac{1}{2}\right)$ while for the optimal schedule it is $(s-1)\left(\left(\frac{s}{s-1}\right)^m - \left(\frac{s}{s-1}\right)^{m-s}\right)$. The price of anarchy of this instance is, therefore, at least

$$\frac{(s-1)\left(\left(\frac{s}{s-1}\right)^m - \frac{1}{2}\right)}{(s-1)\left(\left(\frac{s}{s-1}\right)^m - \left(\frac{s}{s-1}\right)^{m-s}\right)} = \frac{\left(\frac{s}{s-1}\right)^s - \frac{1}{2}\left(\frac{s}{s-1}\right)^{-(m-s)}}{\left(\frac{s}{s-1}\right)^s - 1}. \tag{5.4}$$

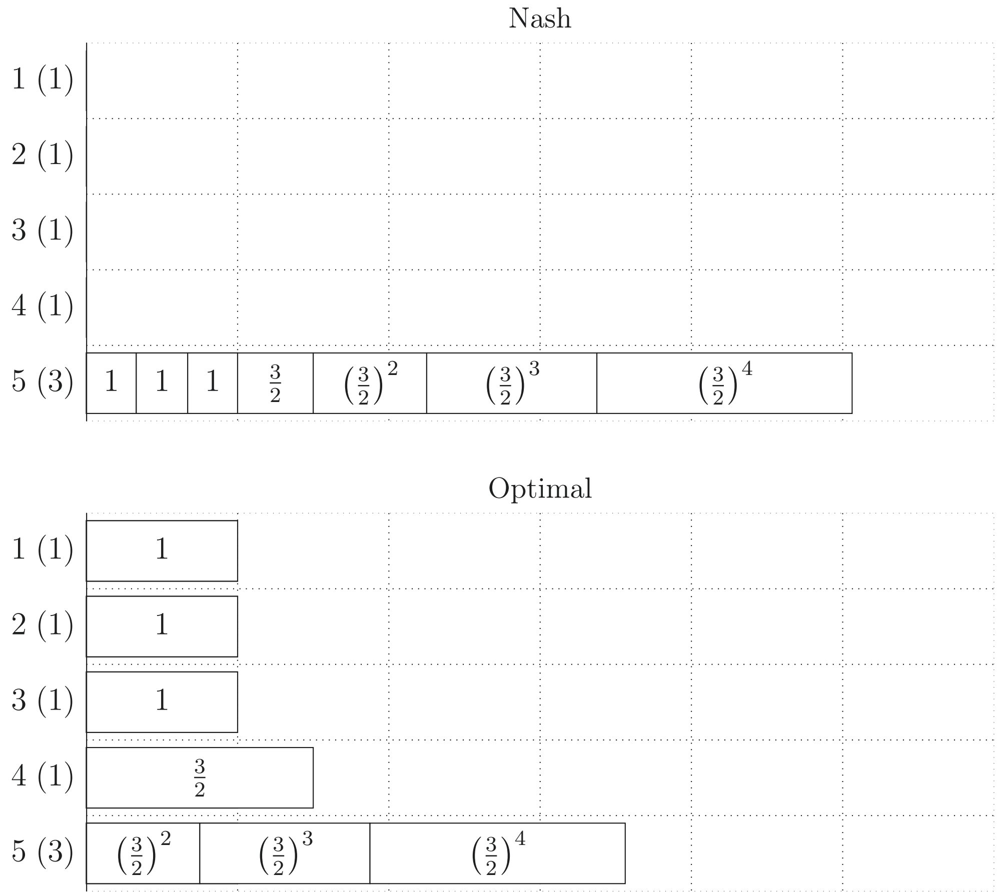

Nash
1 (1)
2 (1)
3 (1)
4 (1)
5 (3)
1
1
1
$\frac{3}{2}$
$\left(\frac{3}{2}\right)^2$
$\left(\frac{3}{2}\right)^3$
$\left(\frac{3}{2}\right)^4$
Optimal
1 (1)
1
2 (1)
1
3 (1)
1
4 (1)
$\frac{3}{2}$
5 (3)
$\left(\frac{3}{2}\right)^2$
$\left(\frac{3}{2}\right)^3$
$\left(\frac{3}{2}\right)^4$

Figure 5.2: Nash and optimal schedule for Hoeksma-Uetz instance with 5 machines and 7 jobs

First sending m to infinity and then s gives a lower bound on the price of anarchy of $\frac{e}{e-1}$ because $\left(\frac{s}{s-1}\right)^s \to e$.

5.4 An Attempt at Improving the Pure Price of Anarchy Bound

The upper bounds we discussed in the previous section are robust price of anarchy bounds: they not only prove bounds on the pure price of anarchy but extend to the coarse correlated price of anarchy. From the identical machine setting it is known, however, that there is a separation between the pure price of anarchy, which is 1, and the mixed price of anarchy, which is $\frac{3}{2} - \frac{1}{2m}$. If there also exists a separation for the related machine scheduling setting we need techniques that enable us to prove this. One method that is able to capture specifics of pure Nash equilibria, which has been especially powerful for congestion games, is the primal-dual method introduced by Bilò (2018).

On a high level, Bilò's primal-dual method applied to related machine scheduling games works as follows. Given a class of games $\mathcal{G}$ (e.g., the class of related machine scheduling games with m machines, n jobs, and machine speeds $s_1 \leq \ldots \leq s_m$), where each game in the class is defined by a set of parameters (e.g., the processing times of the jobs), together with a social cost function C we are interested in the game $\Gamma \in \mathcal{G}$, Nash equilibrium $\boldsymbol{x}$ and optimal strategy profile $\boldsymbol{x}^*$ such that $C(\boldsymbol{x})/C(\boldsymbol{x}^*)$ is maximized. Suppose we are given a Nash equilibrium $\boldsymbol{x}$ and optimal strategy profile $\boldsymbol{x}^*$ and suppose we can write down a linear program $LP(\boldsymbol{x}, \boldsymbol{x}^*)$ that maximizes $C(\boldsymbol{x})$ over the set of parameters while the parameters are constrained by $C(\boldsymbol{x}^*)$ being normalized to one and $\boldsymbol{x}$ must be a Nash equilibrium. Note that, here it is assumed that one can globally and linearly scale down the parameters defining the game such that $C(\boldsymbol{x}^*)$ is one. Weak duality implies that a feasible solution for the dual with some objective value γ gives an upper bound on the primal program of γ which is a bound on $C(\boldsymbol{x})$. Since $C(\boldsymbol{x}^*)$ is 1, this is also a bound on the price of anarchy. If one can find a dual fitting with objective value γ for Nash equilibrium $\boldsymbol{x}$ and optimal strategy profile $\boldsymbol{x}^*$ for which one can argue that they are worst-case this gives a bound on the price of anarchy for the whole class of games. Another way is to have a procedure that for every Nash equilibrium and every optimal strategy profile provides a dual fitting procedure with objective value at most γ. We will formalize this latter approach for the related machine scheduling setting below.

A First Attempt

In our first attempt, we take as input to our linear program a Nash schedule $\boldsymbol{x}$, an optimal strategy profile $\boldsymbol{x}^*$, and machine speeds $\boldsymbol{s} = (s_1, \ldots, s_m)$. We optimize over the processing times of the jobs. Following smoothness, we only add the constraint that a player has no incentive to change their strategy to the one they play in the optimal strategy profile.

We obtain the following linear program

$$\begin{aligned} LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) = \max_{p_1, p_2, \ldots, p_n} C(\boldsymbol{x}) & \qquad \text{s.t.} && (5.5)\\ C(\boldsymbol{x}^*) = 1 & && (5.6)\\ c_j(\boldsymbol{x}) - c_j(x_j^*, \boldsymbol{x}_{-j}) \leq 0 & \qquad \forall j \in N : x_j \neq x_j^* && (5.7)\\ p_j - p_{j+1} \leq 0 & \qquad \forall j \in [n-1] && (5.8)\\ -p_j \leq 0 & \qquad \forall j \in N, && (5.9) \end{aligned}$$

where we maximize the cost of a Nash equilibrium $\boldsymbol{x}$ while satisfying constraints (5.6)–(5.9). Constraint (5.6) normalizes the sum of completion times of the optimal schedule to 1. No player has an incentive to deviate from the machine in the Nash strategy profile to the machine in the optimal strategy profile because of constraint (5.7). Lastly, constraints (5.8) and (5.9) make sure that the processing are non-decreasing and non-negative.

Filling in the definitions of the cost functions yields

$$LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) = \max_{p_1,p_2,\dots,p_n} \sum_j \frac{1}{s_{x_j}} \sum_{\substack{k \leq j \\ x_k = x_j}} p_k \qquad \text{s.t.} \tag{5.10}$$

$$\sum_j \frac{1}{s_{x_j^*}} \sum_{\substack{k \leq j \\ x_k^* = x_j^*}} p_k = 1 \tag{5.11}$$

$$\frac{1}{s_{x_j}} \left(p_j + \sum_{\substack{k<j \\ x_k = x_j}} p_k \right) - \frac{1}{s_{x_j^*}} \left(p_j + \sum_{\substack{k<j \\ x_k = x_j^*}} p_k \right) \leq 0 \qquad \forall j \in N : x_j \neq x_j^* \tag{5.12}$$

$$p_j - p_{j+1} \leq 0 \qquad \forall j \in [n-1] \tag{5.13}$$

$$-p_j \leq 0 \qquad \forall j \in N.$$

This linear program has a dual linear program. For the constraint (5.11) we introduce the variable γ. For each of the constraints in (5.12) we introduce y_j for each $j \in N$ and for the constraints in (5.13) we have variables $z_1, z_2, \dots, z_{n-1}$. For notational convenience, we define $z_0 = z_n = 0$. Observe that if $x_j = x_j^*$ there is no Nash constraint while we have introduced a variable y_j. This is also merely a notational convenience as the corresponding variables cancel in the constraints below.

$$Dual(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) = \min \gamma \qquad \text{s.t.} \tag{5.14}$$

$$\gamma \cdot \frac{1}{s_{x_k^*}} \sum_{\substack{j \geq k \\ x_j^* = x_k^*}} 1 + \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} y_j - \frac{1}{s_{x_k^*}} y_k -$$

$$\frac{1}{s_{x_k}} \sum_{\substack{j > k \\ x_j^* = x_k}} y_j + z_k - z_{k-1} \geq \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} 1 \qquad \forall k \in N$$

$$y_j \geq 0 \qquad \forall j \in N$$

$$z_k \geq 0 \qquad \forall k \in [n-1].$$

The procedure we envision is outlined in the following theorem.

Theorem 5.4.1. *Consider a related machine scheduling game with m machines and n jobs. If, for every Nash equilibrium $\boldsymbol{x}$, optimal strategy profile $\boldsymbol{x}^*$ and machine speeds $\boldsymbol{s} = (s_1, \dots, s_m)$, there exist $\gamma', y_1, \dots, y_n, z_1, \dots, z_{n-1}$, that are feasible for (5.14) with $\gamma' \leq \gamma$ for some constant γ. Then, the price of anarchy for related machine scheduling with m machines and n jobs is at most γ.*

Proof: Consider a machine scheduling game with m machines with speeds $\boldsymbol{s} = (s_1, \ldots, s_m)$ and n jobs with processing times $p_1 \leq \ldots \leq p_n$ and assume $s_1, \ldots, s_m, p_1, \ldots, p_n$ are such that this game has the maximal price of anarchy of any related machine scheduling game with m machines and n jobs.

Let $\boldsymbol{x}^*$ be an optimal strategy profile of this game and let $\boldsymbol{x}$ be the worst-case Nash equilibrium, i.e., the Nash equilibrium with the highest sum of completion times.

Let $p'_j = \frac{p_j}{C(\boldsymbol{x}^*)}$ for each $j \in N$. Let $C'(\boldsymbol{x})$ and $C'(\boldsymbol{x}^*)$ be the sum of completion times with respect to $p'_1, \ldots, p'_n$ of $\boldsymbol{x}$ and $\boldsymbol{x}^*$, respectively. Observe that $p'_1, \ldots, p'_n$ are feasible for $LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s})$ (by assumption, (5.12) is satisfied for $p_1, \ldots, p_n$ and the scaling of the processing times does not change that) and therefore

$$C'(\boldsymbol{x}) \leq LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}). \tag{5.15}$$

By the theorem statement, there exist $\gamma', y_1, \ldots, y_n, z_1, \ldots, z_{n-1}$ that are feasible for the dual program (5.14). Hence, by weak duality, the objective value of the primal program $LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s})$ is at most γ'. Putting it all together gives

$$\frac{C(\boldsymbol{x})}{C(\boldsymbol{x}^*)} = \frac{C'(\boldsymbol{x})}{C'(\boldsymbol{x}^*)} = C'(\boldsymbol{x}) \leq LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) \leq \gamma' \leq \gamma.$$

By assumption, this is the instance with the maximal price of anarchy and so we can conclude that any related machine scheduling game with n jobs and m machines has a price of anarchy at most γ. □

We can now easily recover the bound of 2 on the pure price of anarchy for any related machine scheduling game (of any size).

Theorem 5.4.2. *The pure price of anarchy of any related machine scheduling game is at most 2.*

Proof: We will verify that $\gamma = 2, y_j = 1, z_j = 0$ for all j is a feasible solution for the dual (for all possible $\boldsymbol{s}, \boldsymbol{x}^*$ and $\boldsymbol{x}$). With this solution we get

$$2 \cdot \frac{1}{s_{x_k^*}} \sum_{\substack{j \geq k \\ x_j^* = x_k^*}} 1 + \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} 1 - \frac{1}{s_{x_k^*}} y_k - \frac{1}{s_{x_k}} \sum_{\substack{j > k \\ x_j^* = x_k}} 1 + z_k - z_{k-1} \geq \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} 1. \tag{5.16}$$

Using the notation defined in (5.1), the left hand side rewrites to

$$\begin{aligned}
&2\cdot\frac{z_{x_k^*}(x^*,k)+1}{s_{x_k^*}}+\frac{z_{x_k}(x,k)+1}{s_{x_k}}-\frac{1}{s_{x_k^*}}-\frac{z_{x_k}(x^*,k)}{s_{x_k}}\\
&\geq 2\cdot\frac{z_{x_k^*}(x^*,k)+1}{s_{x_k^*}}+\frac{z_{x_k}(x,k)+1}{s_{x_k}}-\frac{1}{s_{x_k^*}}-\frac{z_{x_k^*}(x^*,k)+1}{s_{x_k^*}}\\
&=\frac{z_{x_k^*}(x^*,k)+1}{s_{x_k^*}}+\frac{z_{x_k}(x,k)+1}{s_{x_k}}-\frac{1}{s_{x_k^*}}\\
&\geq\frac{z_{x_k}(x,k)+1}{s_{x_k}}.
\end{aligned}$$

where the first inequality follows from Theorem 5.3.2 and the second inequality from the fact that $z_{x_k^*}(x^*,k)\geq 0$. We end up with the right hand side of (5.16) and thus the constraints are satisfied and the dual has objective value 2. Invoking Theorem 5.4.1 completes the proof. □

Not Strong Enough

Example 5.4.3. Consider the instance depicted in Figure 5.3. There are 2 machines of speed 1 and 3 jobs. The numbers written inside the boxes are the indices of the jobs. Let $\boldsymbol{x}=(2,1,1)$ be the schedule where job 1 is alone on machine 2 and jobs 2 and 3 are scheduled on machine 1. Also let $\boldsymbol{x}^*=(1,2,1)$. Now, jobs 1 and 3 are on machine 1, while job 2 is alone on machine 2. Note that these are feasible Nash and optimal schedules when all three jobs have a processing time of 1.

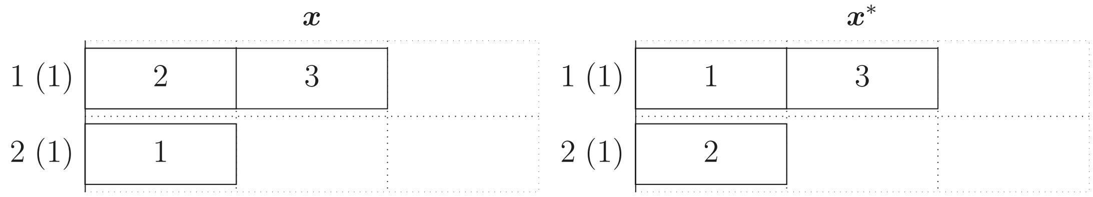


Figure 5.3: Schedules in Example 5.4.3.

The dual for this instance becomes

$$\begin{aligned}
\min\quad \gamma\qquad &\text{s.t.}\\
&\gamma\cdot 2-y_2+z_1\geq 1\\
&\gamma\cdot 1+z_2-z_1\geq 2\\
&\gamma\cdot 1-z_2\geq 1.
\end{aligned}$$

Summing the last two constraints gives $2\gamma - z_1 \geq 3$. As $z_1 \geq 0$, this implies that $\gamma \geq \frac{3}{2}$. While we know that the pure price of anarchy for identical machine scheduling is 1.

Also for related machine scheduling this formulation is not as strong as we would like it to be. Take the machine speeds, Nash and optimal schedule from the lower bound instance in Example 5.3.3, i.e., machine m has speed s and all the other machines have speed 1, $\boldsymbol{x}^* = (1, 2, \ldots, m, m, \ldots, m)$ and $\boldsymbol{x} = (m, \ldots, m)$. An optimal dual fitting is given by

$$\gamma = \frac{5s+1}{3s+1} - \frac{s-1}{3s+1} \cdot \left(\frac{s-1}{s}\right)^{m-s-1} \tag{5.17}$$

$$z_j = \frac{j(s-j)}{2s} \qquad \text{for } 1 \leq j \leq s-1 \tag{5.18}$$

$$y_{s+j} = 1 - \frac{1}{2} \cdot \left(\frac{s-1}{s}\right)^{j} \qquad \text{for } 1 \leq j \leq m-s-2 \tag{5.19}$$

$$(y_j \text{ for } s+1 \leq j \leq m-2)$$

$$y_{m-1} = \frac{(s+2)^2-3}{3s+1} - \frac{s(s+3)}{2(3s+1)} \left(\frac{s-1}{s}\right)^{m-s-1} \tag{5.20}$$

$$z_{n-j} = (\gamma - 1) \cdot \frac{j(j+1)}{2s} \qquad \text{for } 1 \leq j \leq s \tag{5.21}$$

$$(z_j \text{ for } n-1 \geq j \geq m-1),$$

where the remaining variables are 0.

Theorem 5.4.4. *The dual fitting* (5.17)–(5.21) *is optimal.*

In the proofs of Theorems 5.4.4 and 5.4.6 we regularly use the following identities, which follow from the closed-form formula for a geometric sum.

Fact 5.4.5. For $k \in \mathbb{N}$ and $s > 1$ it holds that:

$$\sum_{j=1}^{k} \left(\frac{s}{s-1}\right)^{j} = s\left(\left(\frac{s}{s-1}\right)^{k} - 1\right),$$

$$\sum_{j=1}^{k} \left(\frac{s-1}{s}\right)^{j} = (s-1)\left(1 - \left(\frac{s-1}{s}\right)^{k}\right).$$

Proof of Theorem 5.4.4: Recall the constraints (5.14) in the dual

$$\gamma \cdot \frac{1}{s_{x_k^*}} \sum_{\substack{j \geq k \\ x_j^* = x_k^*}} 1 + \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} y_j - \frac{1}{s_{x_k^*}} y_k - \frac{1}{s_{x_k}} \sum_{\substack{j > k \\ x_j^* = x_k}} y_j + z_k - z_{k-1} \geq \frac{1}{s_{x_k}} \sum_{\substack{j \geq k \\ x_j = x_k}} 1 \quad \forall k \in [n].$$

First, we verify that these constraints are satisfied.

We make a few observations. For all k we have $s_{x_k} = s$. Note that jobs with index $k \leq m-1$ are alone on a slow machine in the optimal schedule and thus $s_{x^*_k} = 1$ while for $k \geq m$ it holds that $s_{x_k} = s$. Furthermore, $z_j = 0$ for $s \leq j \leq m-1$ and $y_j = 0$ for $j \leq s$ and $j \geq m$. And $n = m+s-1$. Next, it holds that

$$\sum_{j \geq k : x_j = x_k} 1 = n+1-k,$$

$$\sum_{j \geq k : x^*_j = x^*_k} 1 = \begin{cases} 1 & \text{if } k \leq m-1 \\ n+1-k & \text{if } k \geq m. \end{cases}$$

We verify the constraints for all values of k. For $k \geq m$ the constraint becomes

$$\gamma \frac{n+1-k}{s} + z_k - z_{k-1} \geq \frac{n+1-k}{s}.$$

As $z_k = (\gamma-1)\frac{(n-k)(n-k+1)}{2s}$ we have

$$(\gamma-1)\left(\frac{n+1-k}{s} + \frac{(n-k)(n-k+1)}{2s} - \frac{(n-k+1)(n-k+2)}{2s}\right) \geq 0,$$

which is actually an equality, and so the inequality is satisfied.

We precompute

$$\begin{aligned} \gamma + \frac{1}{s} y_{m-1} &= \frac{5s+1}{3s+1} - \frac{s-1}{3s+1} \cdot \left(\frac{s-1}{s}\right)^{m-s-1} + \\ &\quad \frac{(s+2)^2-3}{s(3s+1)} - \frac{s+3}{2(3s+1)} \left(\frac{s-1}{s}\right)^{m-s-1} \\ &= \frac{(3s+1)(2s+1)}{s(3s+1)} - \frac{3s+1}{2(3s+1)} \left(\frac{s-1}{s}\right)^{m-s-1} \\ &= \frac{2s+1}{s} - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1}. \end{aligned}$$

For $k = m-1$ the constraint becomes (recall $n = m+s-1$ and so $n+1-k = s+1$)

$$\gamma + \frac{1}{s} y_{m-1} - y_{m-1} + z_{m-1} \geq \frac{s+1}{s}.$$

Using the result for $\gamma + \frac{1}{s}y_{m-1}$ and filling in z_{m-1} yields

$$\begin{aligned}
\iff\quad & \frac{2s+1}{s} - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1} + (\gamma-1)\frac{s+1}{2} - \frac{s+1}{s} \geq y_{m-1} \\
\iff\quad & 1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1} + \gamma\frac{s+1}{2} - \frac{s+1}{2} \geq y_{m-1} \\
\iff\quad & \begin{array}{l} 1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1} + \frac{(5s+1)(s+1)}{2(3s+1)} \\ - \frac{(s-1)(s+1)}{2(3s+1)} \cdot \left(\frac{s-1}{s}\right)^{m-s-1} - \frac{s+1}{2} \end{array} \geq y_{m-1} \\
\iff\quad & \frac{s^2+4s+1}{3s+1} - \frac{s(s+3)}{2(3s+1)} \cdot \left(\frac{s-1}{s}\right)^{m-s-1} \geq y_{m-1},
\end{aligned}$$

which is exactly the definition of y_{m-1} and thus an equality.

Let $s+1 \leq k \leq m-2$.

$$\begin{aligned}
& \gamma + \frac{1}{s}y_{m-1} + \frac{1}{s}\sum_{j=k}^{m-2} y_j - y_k \geq \frac{n+1-k}{s} \\
\iff\quad & \begin{array}{l} \frac{2s+1}{s} - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1} \\ + \frac{1}{s}\sum_{j=k}^{m-2}\left(1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{j-s}\right) \end{array} \geq \frac{m+s-k}{s} + y_k \\
\iff\quad & \begin{array}{l} \frac{2s+1}{s} - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s-1} + \frac{m-1-k}{s} \\ - \frac{1}{2}\left(\left(\frac{s-1}{s}\right)^{k-s} - \left(\frac{s-1}{s}\right)^{m-s-1}\right) \end{array} \geq \frac{m+s-k}{s} + y_k \\
\iff\quad & \frac{2s+1}{s} + \frac{m-1-k}{s} - \frac{1}{2}\left(\frac{s-1}{s}\right)^{k-s} \geq \frac{m+s-k}{s} + y_k \\
\iff\quad & 1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{k-s} \geq y_k.
\end{aligned}$$

Lastly, we check the constraints for $k \leq s$. Making use of the above computations for $k = s+1$ we precompute

$$\begin{aligned}\gamma + \frac{1}{s}y_{m-1} + \frac{1}{s}\sum_{j=s+1}^{m-2} y_j &= 1 - \frac{s-1}{2s} + \frac{m-1}{s}\\ &= \frac{m}{s} + \frac{s-1}{2s}.\end{aligned}$$

For $k \leq s$ the constraint becomes

$$\begin{aligned}&\gamma + \frac{1}{s}y_{m-1} + \frac{1}{s}\sum_{j=s+1}^{m-2} y_{1,j} + z_k - z_{k-1} \geq \frac{m+s-k}{s}\\ \iff \quad &\frac{m}{s} + \frac{s-1}{2s} + \frac{k(s-k)}{2s} - \frac{(k-1)(s+1-k)}{2s} \geq \frac{m+s-k}{s},\end{aligned}$$

which holds with equality.

For optimality we will give a primal solution which has γ as objective value. Since this is the objective value of the dual strong duality implies that both solutions are optimal. Let

$$p'_j = \begin{cases} 1 & \text{for } j \leq s \\ \left(\frac{s}{s-1}\right)^{j-s} & \text{for } s+1 \leq j \leq m-1 \\ \left(\frac{s}{s-1}\right)^{m-s-1} & \text{for } m \leq j, \end{cases}$$

and let $p_j = \frac{p'_j}{C'(\boldsymbol{x}^*)}$. By the scaling we know that (5.11) is satisfied and because $s \geq 2$ the processing times are non-decreasing. The jobs for which $x_j \neq x^*_j$ and so for which we have to verify (5.12) are the jobs with $j \leq m-1$. For jobs $1,\ldots,s$ we see that their completion time is

$$C_j(\boldsymbol{x}) = \frac{1}{C'(\boldsymbol{x}^*)}\sum_{k=1}^{j}\frac{1}{s} = \frac{1}{C'(\boldsymbol{x}^*)}\frac{j}{s} \leq p_j,$$

which is less than their completion time on a slow machine. For jobs $s+1,\ldots,m-1$ it holds that

$$\begin{aligned}C_j(\boldsymbol{x}) &= \frac{1}{s}\sum_{k=1}^{j} p_j\\ &= \frac{1}{C'(\boldsymbol{x}^*)}\frac{1}{s}\left(s + \sum_{k=1}^{j}\left(\frac{s}{s-1}\right)^{k-s}\right)\\ &= \frac{1}{C'(\boldsymbol{x}^*)}\frac{1}{s}\left(s + s\left(\left(\frac{s}{s-1}\right)^{j-s} - 1\right)\right)\\ &= \frac{1}{C'(\boldsymbol{x}^*)}\left(\frac{s}{s-1}\right)^{j-s} = p_j.\end{aligned}$$

Hence these processing times are indeed a feasible solution to the primal. It remains to determine the objective value.

The cost of Nash with respect to $p'_1, \ldots, p'_n$ is

$$\begin{aligned}
C'(\boldsymbol{x}) &= \sum_{j=1}^{n} C'_j(\boldsymbol{x}) \\
&= \sum_{j=1}^{s} C'_j(\boldsymbol{x}) + \sum_{j=s+1}^{m-1} C'_j(\boldsymbol{x}) + \sum_{j=m}^{m+s-1} C'_j(\boldsymbol{x}) \\
&= \frac{1}{s}\sum_{j=1}^{s} j + \sum_{j=s+1}^{m-1} \left(\frac{s}{s-1}\right)^{j-s} + \frac{1}{s}\sum_{j=m}^{m+s-1} (1+j+1-m)\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= \frac{s+1}{2} + s\left(\left(\frac{s}{s-1}\right)^{m-s-1} - 1\right) + \frac{1}{s}\sum_{j=1}^{s}(1+j)\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= \frac{s+1}{2} + s\left(\left(\frac{s}{s-1}\right)^{m-s-1} - 1\right) + \frac{5s+1}{2}\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= \frac{5s+1}{2}\left(\frac{s}{s-1}\right)^{m-s-1} - \frac{s-1}{2}.
\end{aligned}$$

And the cost of $\boldsymbol{x}^*$ with respect to $p'_1, \ldots, p'_n$ is

$$\begin{aligned}
C'(\boldsymbol{x}^*) &= \sum_{j=1}^{n} C'_j(\boldsymbol{x}^*) \\
&= \sum_{j=1}^{s} C'_j(\boldsymbol{x}^*) + \sum_{j=s+1}^{m-1} C'_j(\boldsymbol{x}^*) + \sum_{j=m}^{m+s-1} C'_j(\boldsymbol{x}^*) \\
&= \sum_{j=1}^{s} 1 + \sum_{j=s+1}^{m-1} \left(\frac{s}{s-1}\right)^{j-s} + \frac{1}{s}\sum_{j=m}^{m+s-1} (j+1-m)\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= s + s\left(\left(\frac{s}{s-1}\right)^{m-s-1} - 1\right) + \frac{1}{s}\sum_{j=1}^{s} j\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= s\left(\frac{s}{s-1}\right)^{m-s-1} + \frac{3s+1}{2}\left(\frac{s}{s-1}\right)^{m-s-1} \\
&= \frac{3s+1}{2}\left(\frac{s}{s-1}\right)^{m-s-1}.
\end{aligned}$$

Hence,

$$\frac{C(\boldsymbol{x})}{C(\boldsymbol{x}^*)} = \frac{C'(\boldsymbol{x})}{C'(\boldsymbol{x}^*)} = \frac{5s+1}{3s+1} - \frac{s-1}{3s+1}\left(\frac{s-1}{s}\right)^{m-s-1}. \tag{5.22}$$

Thus, the primal and dual objective values are the same, and therefore we know that the solutions are optimal. □

Equation (5.22) gives an upper bound on the price of anarchy of $\frac{5}{3}$ for this particular Nash schedule and optimal strategy profile when sending m and s to infinity, while we know that the upper bound can be as small as $\frac{e}{e-1}$.

Stronger Formulation

To strengthen our formulation we not only add the Nash condition that a job cannot move from its Nash machine to the optimal machine, but we also add the Nash condition for all other machines.

$$
\begin{aligned}
LP(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) = \max_{p_1, p_2, \ldots, p_n} \sum_j \sum_{\substack{k \le j \\ x_k = x_j}} \frac{1}{s_{x_k}} p_k \qquad \text{s.t.}& \\
\sum_j \frac{1}{s_{x_j^*}} \sum_{k \le j : x_k^* = x_j^*} p_k = 1& \\
\frac{1}{s_{x_j}} \left(p_j + \sum_{\substack{k<j \\ x_k = x_j}} p_k \right) - \frac{1}{s_i} \left(p_j + \sum_{k<j : x_k = i} p_k \right) \le 0& \qquad \forall i \in M, j \in N \\
p_j - p_{j+1} \le 0& \qquad \forall j \in N \\
-p_j \le 0& \qquad \forall j \in N.
\end{aligned}
$$

The dual of this primal program is

$$
Dual(\boldsymbol{x}^*, \boldsymbol{x}, \boldsymbol{s}) = \min \gamma \qquad \text{s.t.} \tag{5.23}
$$

$$
\gamma \cdot \frac{1}{s_{x_k^*}} \sum_{\substack{j \ge k \\ x_j^* = x_k^*}} 1 + \sum_i \left(\frac{1}{s_{x_k}} \left(\sum_{\substack{j \ge k \\ x_j = x_k}} y_{ij} \right) - \frac{1}{s_i} y_{ik} \right) \tag{5.24}
$$

$$
-\frac{1}{s_{x_k}} \sum_{j>k} y_{x_k j} + z_k - z_{k-1} \ge \frac{1}{s_{x_k}} \sum_{\substack{j \ge k \\ x_j = x_k}} 1 \quad \forall k \in N
$$

$$
y_{ij} \ge 0 \qquad \forall i \in M, j \in N \tag{5.25}
$$

$$
z_k \ge 0 \qquad \forall k \in [n-1]. \tag{5.26}
$$

If we put the machine speeds, Nash, and optimal schedule from Example 5.3.3

in here. We can find the following optimal dual fitting:

$$\gamma = \frac{\left(\frac{s}{s-1}\right)^s - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s}}{\left(\frac{s}{s-1}\right)^s - 1} \tag{5.27}$$

$$z_j = \frac{j(s-j)}{2s} \qquad \text{for } 1 \le j \le s-1 \tag{5.28}$$

$$y_{1,s+j} = 1 - \frac{1}{2} \cdot \left(\frac{s-1}{s}\right)^j \qquad \text{for } 1 \le j \le m-s \tag{5.29}$$

$$(y_j \text{ for } s+1 \le j \le m)$$

$$y_{1,n+1-j} = (\gamma - 1)\left(\left(\frac{s}{s-1}\right)^j - 1\right) \qquad \text{for } 1 \le j \le n-m = s-1 \tag{5.30}$$

$$(y_j \text{ for } n \ge j \ge m+1),$$

and the remaining variables are set to 0. Observe that (5.29) and (5.30) coincide for $y_{1,m}$.

Theorem 5.4.6. *The dual fitting* (5.27)–(5.30) *is optimal.*

Proof: The Hoeksma-Uetz instance in Example 5.3.3 has exactly this γ as price of anarchy (see (5.4)). Moreover, we know that the processing times, scaled down by a factor $C(\boldsymbol{x}^*)$ are feasible for the primal with objective value γ. Hence, we only have to verify that the solution is feasible and strong duality implies that it is optimal.

Observe that

$$\gamma - 1 = \frac{1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{m-s}}{\left(\frac{s}{s-1}\right)^s - 1}.$$

Consider $k \ge m$. Job k will both in the Nash schedule and in the optimal schedule be on the fastest machine. Removing all variables that are 0, the constraint (5.24) simplifies to

$$\begin{aligned}
& \gamma \cdot \frac{1}{s}\sum_{j \ge k} 1 + \frac{1}{s}\sum_{j\ge k} y_{1,j} - y_{1,k} \ge \frac{1}{s}\sum_{j \ge k} 1 \\
\iff \quad & \frac{1}{s}\sum_{j \ge k} \left(y_{1,j} + (\gamma - 1)\right) \ge y_{1,k} \\
\iff \quad & \frac{1}{s}\sum_{j=1}^{n+1-k} (\gamma - 1)\left(\frac{s}{s-1}\right)^j \ge (\gamma - 1)\left(\left(\frac{s}{s-1}\right)^{n-k-1} - 1\right) \\
\iff \quad & \frac{1}{s}\sum_{j=1}^{n+1-k} \left(\frac{s}{s-1}\right)^j \ge \left(\frac{s}{s-1}\right)^{n-k-1} - 1,
\end{aligned}$$

which is an equality by Fact 5.4.5.

Let $s+1 \le k \le m-1$. The job will be on a slow machine in the optimal schedule. The constraint becomes

$$\begin{aligned}
&& \gamma + \frac{1}{s}\sum_{j\ge k} y_{1,j} - y_{1,k} &\ge \frac{1}{s}\sum_{j\ge k} 1 \\
\Longleftrightarrow && \gamma + \frac{1}{s}\sum_{j=k}^{m-1}(y_{1,j}-1) + \frac{1}{s}\sum_{j=m}^{m+s-1}(y_{1,j}-1) &\ge y_{1,k} \\
\Longleftrightarrow && \gamma - \frac{1}{2s}\sum_{j=k}^{m-1}\left(\frac{s-1}{s}\right)^{j-s} + \frac{1}{s}\sum_{j=1}^{s}(\gamma-1)\left(\left(\frac{s}{s-1}\right)^{j} - 1\right) - 1 &\ge y_{1,k} \\
\Longleftrightarrow && \begin{array}{l} \gamma - \frac{1}{2}\left(\left(\frac{s-1}{s}\right)^{k-s} - \left(\frac{s-1}{s}\right)^{m-s}\right) \\ + (\gamma-1)\left(\left(\left(\frac{s}{s-1}\right)^{s} - 1\right) - 1\right) - 1 \end{array} &\ge y_{1,k} \\
\Longleftrightarrow && -\frac{1}{2}\left(\left(\frac{s-1}{s}\right)^{k-s} - \left(\frac{s-1}{s}\right)^{m-s}\right) + 1 - \frac{1}{2}\left(\frac{s}{s-1}\right)^{m-s} &\ge y_{1,k} \\
\Longleftrightarrow && 1 - \frac{1}{2}\left(\frac{s-1}{s}\right)^{k-s} &\ge y_{1,k},
\end{aligned}$$

which is an equality.

Finally, $k \le s$. Making use of the calculations above for $k = s+1$ we note that for all $k \le s$

$$\begin{aligned}
\gamma + \frac{1}{s}\sum_{j\ge k}(y_{1,j}-1) &= \gamma + \frac{1}{s}\sum_{j=s+1}^{n}(y_{1,j}-1) - \frac{s+1-k}{s} \\
&= 1 - \frac{1}{2}\left(\frac{s-1}{s}\right) - \frac{s+1-k}{s} \\
&= \frac{2k-s-1}{s}.
\end{aligned}$$

Recalling that $z_0 = 0$ the constraint becomes

$$\begin{aligned}
&& \gamma + \frac{1}{s}\sum_{j\ge k} y_{1,j} + z_k - z_{k-1} &\ge \frac{1}{s}\sum_{j\ge k} 1 \\
\Longleftrightarrow && \gamma + \frac{1}{s}\sum_{j\ge k}(y_{1,j}-1) + z_k - z_{k-1} &\ge 0 \\
\Longleftrightarrow && \frac{2k-1-s}{2s} + z_k - z_{k-1} &\ge 0 \\
\Longleftrightarrow && \frac{2k-1-s}{2s} + \frac{k(s-k)}{2s} - \frac{(k-1)(s+1-k)}{2s} &\ge 0,
\end{aligned}$$

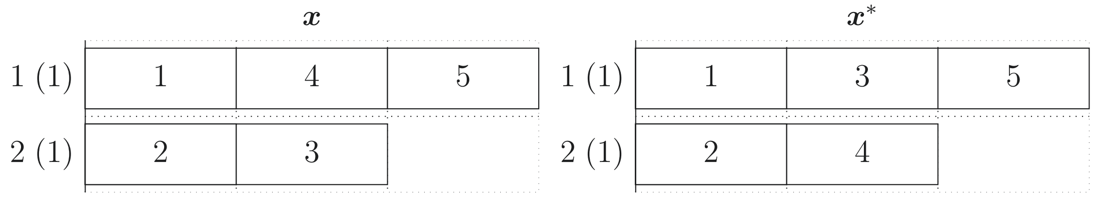


Figure 5.4: Non-negative y variables are necessary.

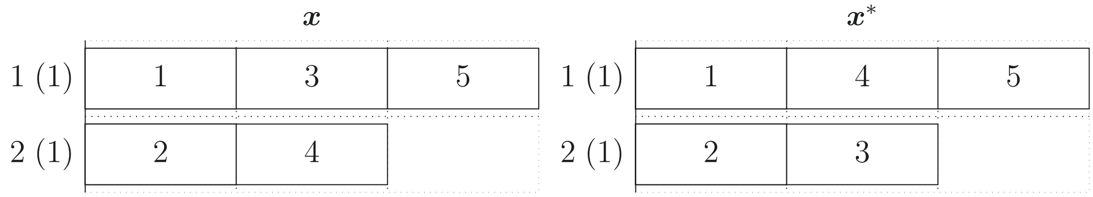


Figure 5.5: Non-negative z variables are necessary.

which is an equality. Thus the fitting is feasible and as mentioned because the (scaled) processing times from the Hoeksma-Uetz instance give a primal solution with the same objective value strong duality implies that it is optimal. □

Now we have that the dual has objective value $\gamma \leq \frac{e}{e-1}$ coinciding with the price of anarchy bound we knew. We can conclude that the processing times in Example 5.3.3 are worst-case for these machine speeds, Nash schedule, and optimal schedule.

We have not been able to find a dual fitting independent of $\boldsymbol{x}$ and $\boldsymbol{x}^*$ smaller than 2, but we also have not been able to find instances for which the objective value is greater than $\frac{e}{e-1}$, leaving the possibility open that the technique is strong enough to prove a bound of $\frac{e}{e-1}$ using this technique. We conjecture that this is indeed possible.

Observations. In Theorem 5.4.2 we only made use of the y variables to prove a bound of 2. The instances in Figures 5.4 and 5.5 show that for proving a tight bound we need to use both the y and the z variables. They both consist of 5 jobs with processing time 1. The number in the jobs is their index. If we force $y_{ij} = 0$ for all i, j the objective value of the instance in Figure 5.4 is 3/2. While if we force $z_j = 0$ for all j in the instance in Figure 5.5 the objective value is 4/3. The objective values were obtained by running an LP solver.

5.5 Improved Lower Bound Instances

In this section, we outline a procedure to construct lower bound instances that, for any n and m, have a greater price of anarchy than the Hoeksma-Uetz instances

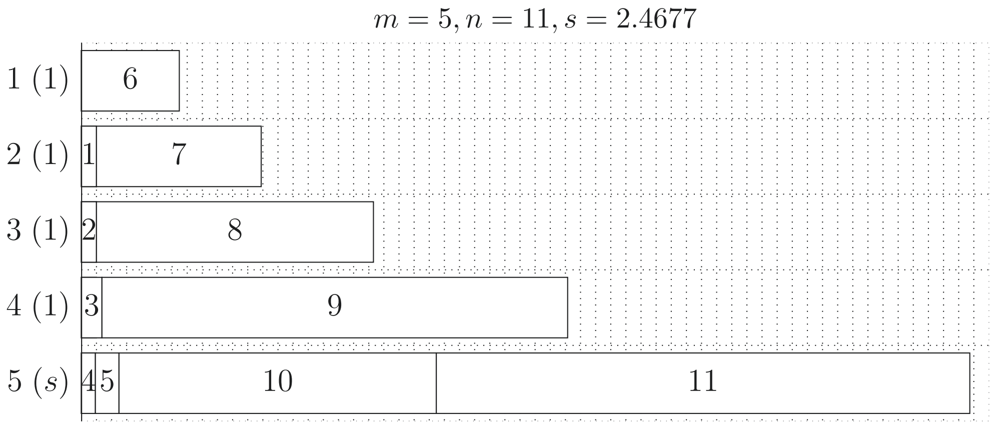


Figure 5.6: Optimal schedule for instance with seven machines and 30 jobs.

(Example 5.3.3) and for 2 machines have a higher price of anarchy than the instance by Zhang et al. (2019). To our knowledge, these are the best-known bounds in the literature.

Define instances parametrized by $n, m, s > 1$ with m machines all having speed 1, except for the last one, which has speed s, and n jobs. The processing times are defined as

$$p_j = \begin{cases} 1 & \text{if } j = 1 \\ \max\left\{p_{j-1}, \frac{\sum_{k=1}^{j-1} p_k}{s-1}\right\} & \text{for } j \geq 2. \end{cases}$$

We verify that all jobs being processed on the fastest machine is a Nash schedule. Consider job j, its completion time is

$$\frac{p_j + \sum_{k=1}^{j-1} p_k}{s} = \frac{\frac{1}{s-1}\sum_{k=1}^{j-1} p_k + \sum_{k=1}^{j-1} p_k}{s} = \frac{\sum_{k=1}^{j-1} p_k}{s-1} = \frac{p_j}{s-1},$$

which would be the same completion time if they unilateraly move to any of the other machines with speed 1.

If the machine speeds and the processing times are fixed the MFT algorithm (Algorithm 8) can quickly compute an optimal schedule. An example for $m = 5, n = 11$ and $s = 2.4677$ is displayed in Figure 5.6.

For a fixed n, m we can numerically find an s maximizing the price of anarchy. Figure 5.7 show the price of anarchy curve as a function of s for $n = 20$ jobs and $m = 2$ or $m = 5$ machines.

Some results are displayed in Table 5.1. For the Hoeksma-Uetz instance the speeds must be an integer between 2 and $m - 1$, and after the speed has been fixed the number of jobs is equal to $n = m + s - 1$. If $n \neq 250$ in the table we have displayed the n which maximizes the price of anarchy of the Hoeksma-Uetz instance for that fixed number of machines m.

We improve the price of anarchy lower bound for fixed n and m. Moreover,

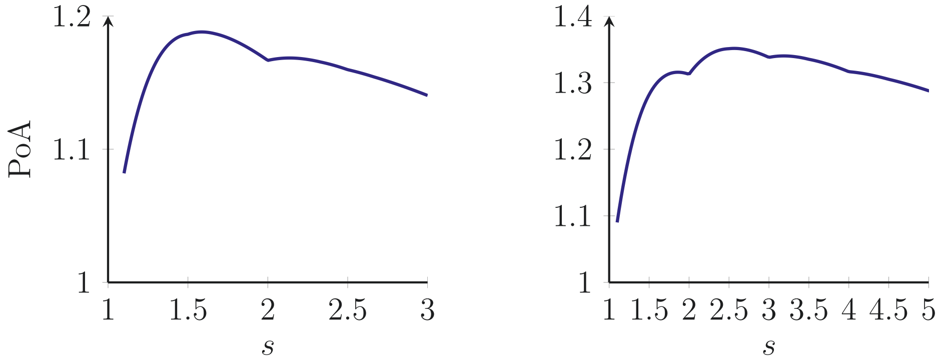


Figure 5.7: Price of anarchy of improved lower bound instance as a function of s with $n = 20$ jobs. On the left with $m = 2$ machines and on the right with $m = 5$ machines.

for a fixed number of machines we increase the price of anarchy lower bound even more. But, note that all bounds are still below $\frac{e}{e-1} \approx 1.5819...$.

For $m = 2$, the Hoeksma-Uetz instance gives a trivial instance, but Zhang et al. (2019) came up with an instance with $m = 2, n = 7, s = 1.5737$ that reaches a price of anarchy of 1.1875. We improve this bound to 1.188.

Finally, note that the $n = 250$ is rather arbitrary, but taking n larger does not change the price of anarchy for the first four decimals.

m	n	s	Our PoA	HU-PoA
2	7	1.5738	1.1875	1.1875 (Zhang)
2	250	1.58605	1.1880	
3	4	1.70710	1.2612	1.25
3	250	1.70704	1.2641	
4	5	2.19192	1.2999	1.2916
4	250	2.41553	1.3162	
5	7	2.51793	1.3461	1.3274
5	250	2.56646	1.3513	
10	13	3.5767	1.4296	1.4216
10	250	4.34235	1.4323	
20	25	5.5838	1.4870	1.4839
20	250	6.452366	1.4909	
50	61	11.4583	1.5344	1.5332
50	250	12.45967	1.5363	
100	119	19.47601	1.5545	1.5541
100	250	20.54459	1.5556	

Table 5.1: Price of anarchy bounds for lower bound instances

5.6 Conclusion

In this chapter, we have seen how the approximation factor of an approximation algorithm and the price of anarchy of a game can be closely related. We have introduced a variant of the primal-dual method of Bilò applied to related machine scheduling games and outlined how it can be used to prove bounds on the pure price of anarchy for related machine scheduling. Although we found dual fittings for the Hoeksma-Uetz instance, we have not been able to find a procedure that works in general. Understanding the different roles of the y and z variables in the dual will probably give a hint on how the procedure should look like. However, we have not been able to do this, even when restricting to identical machines. On the other hand, we also have not found (numerical) evidence yet that the technique is not strong enough. Following Hoeksma and Uetz (2019), we conjecture that the pure price of anarchy is (asymptotically) below 2. We conjecture that it is $\frac{e}{e-1}$ and that our dual-fitting approach should be able to prove this. However, if one has ideas for better lower-bound instances, solving the LP with a solver is an easy way to test them.

Chapter 6

Corruption in Auctions

6.1 Introduction

Hosting an auction is not something anyone can do. One can lack the facilities, time, or expertise to organize one, and so, often, dedicated auctioneers are recruited to organize auctions on behalf of a client. For example, individual sellers usually involve auction houses or auction websites when they want to sell particular objects (such as real estate, cars, artwork, etc.). In private companies, the responsible finance officers are typically in charge of handling the procurement auctions[1]. Similarly, government procurement is usually carried out by some entity that acts on behalf of the government. The dilemma in such settings is that the incentives of the seller and the auctioneer do not align in general: while the seller is interested in extracting the highest payments for the objects (or getting service at the lowest cost), the agent primarily cares about maximizing their own gains from hosting the auction. Although undesirably, this misalignment leads (unavoidably) to fraudulent schemes which might be used by the auctioneer to manipulate the auction to their own benefit.

Corruption in auctions, where an auctioneer engages in bid rigging with one (or several) of the bidders, occurs rather frequently in practice, especially in the public sector (e.g., in construction and procurement auctions). For example, in 1999, the procurement auction for the construction of the new Berlin Brandenburg airport had to be rerun after investigations revealed that the initial winner was able to change the bid after they had illegally acquired information about the application of one of their main competitors (The Wall Street Journal, 1999).

[1] A procurement auction is in some sense the opposite of a standard auction. Instead of a single seller and multiple buyers, there is a single buyer picking the best option among multiple sellers.

As another example, in 1993, the New York City School Construction Authority caused a scandal when an investigation revealed that they used a simple (but effective) bid-rigging scheme in a procurement auction setting (Olmstead, 1993):

> *"In what one investigator described as a nervy scheme, that worker would unseal envelopes at a public bid opening, saving for last the bid submitted by the contractor who had paid him off. At that point, knowing the previous bids, the authority worker would misstate the contractor's bid, insuring that it was low enough to secure the contract but as close as possible to the next highest bid so that the contractor would get the largest possible price."*

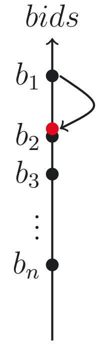
This kind of bid rigging, where the winning bid "magically" aligns with the highest losing bid, is also known as *magic number cheating* (Ingraham, 2005). We refer the reader to (Lengwiler and Wolfstetter, 2010; Menezes and Monteiro, 2006) (and the references therein) for several other bid-rigging examples. Despite the fact that this form of corruption occurs frequently in practice, its negative impact is still poorly understood theoretically and only a few studies exist (mostly in the economics literature, see the related work section).

Our goal is to initiate the study of the social welfare loss caused by corrupt auctioneers in fundamental auction settings. We focus on a basic model that captures the magic number cheating mentioned above and generalizations thereof. Clearly, more sophisticated bid-rigging models are conceivable, and we hope that our work will trigger some future work along these lines.

6.1.1 Capturing Corruption with Hybrid Auctions

Consider the single-item auction setting, and suppose the auctioneer runs a sealed-bid first-price auction. After receipt of all bids, the auctioneer approaches the highest bidder with the offer that they can lower their bid to the second-highest bid in exchange for a bribe. If the highest bidder agrees, they win the auction and pay the second-highest bid for the items plus the corresponding bribe to the auctioneer. If the highest bidder disagrees, they still win the auction but pay their bid for the item according to the first-price auction format. We assume that the bribe to be paid to the auctioneer is a pre-determined fraction $\gamma \in [0, 1]$ of the savings of the highest bidder, i.e., the auctioneer's bribe amounts to γ times the difference between the highest and second-highest bid.

To get more intuition for the scheme, we take a look at an example.



Example 6.1.1. Consider a single-item auction with n bidders. Assume that the bidders submit bids $b_1 > b_2 > \ldots > b_n$. The auctioneer receives the (sealed) bids and concludes that bidder 1 is the winner. The auctioneer approaches bidder 1 and offers to announce that bidder 1 has bid $b_2 + \epsilon$ (for some small $\epsilon > 0$). If they accept they will have to pay $b_1 - (b_2 + \epsilon)$ less to receive the item and the auctioneer expects a γ fraction of it as a bribe. In the end, bidder 1 pays only

$$b_2 + \epsilon + \gamma(b_1 - (b_2 + \epsilon)) = \gamma b_1 + (1 - \gamma)b_2 + (1 - \gamma)\epsilon.$$

If ϵ is small enough, this is less than b_1 which is the payment of player 1 if they would not participate in the corruption scheme.

Note that ϵ can be chosen arbitrarily small such that $b_2 + \epsilon$ gets arbitrarily close to b_2. In this chapter, we assume that ϵ goes to zero and we say that the auctioneer offers the winning bidder to lower their bid to the second-highest bid. Another way of looking at it is that for $\epsilon = 0$, it becomes a tie-breaking issue, and we assume ties are broken in favor of the winner.

In a multi-unit auction, k identical items are up for auction and each bidder submits a non-increasing bid vector which contains their marginal bids for the first up to kth item. The k items are assigned to the k highest marginal bids. If a player submits multiple high marginal bids, it is possible they are allocated multiple items (more details below). In this setting, the procedure described above is adapted accordingly by offering the winning bidders to lower their bids to the highest losing bid. Observe that the payment scheme described above essentially reduces to the winning bidders paying a convex combination of γ times their bids and $(1 - \gamma)$ times the highest losing bid. As we will argue below, this setting is tantamount to studying a *hybrid auction* (γ-HYA), where the items are assigned to the highest bidders (according to the respective single-item or multi-unit auction scheme) and the payments are a convex combination of the first-price and the second-price payments. By varying the parameter $\gamma \in [0, 1]$, γ-HYA thus interpolates between the respective second-price auction ($\gamma = 0$) and the first-price auction ($\gamma = 1$) schemes.

More elaborate corruption schemes are, of course, conceivable. For example, the auctioneer might ask for a fixed amount rather than a fraction of the gains. Or, to avoid setting all bids to the magic number, the auctioneer may want to announce different (bribed) bids for every winning bidder. To capture more general corruption schemes, we also study what we term *γ-approximate first-price auctions* (γ-FPA) in this chapter. Basically, these auctions implement a payment scheme that recovers at least a fraction of $\gamma \in [0, 1]$ of the first-price payment

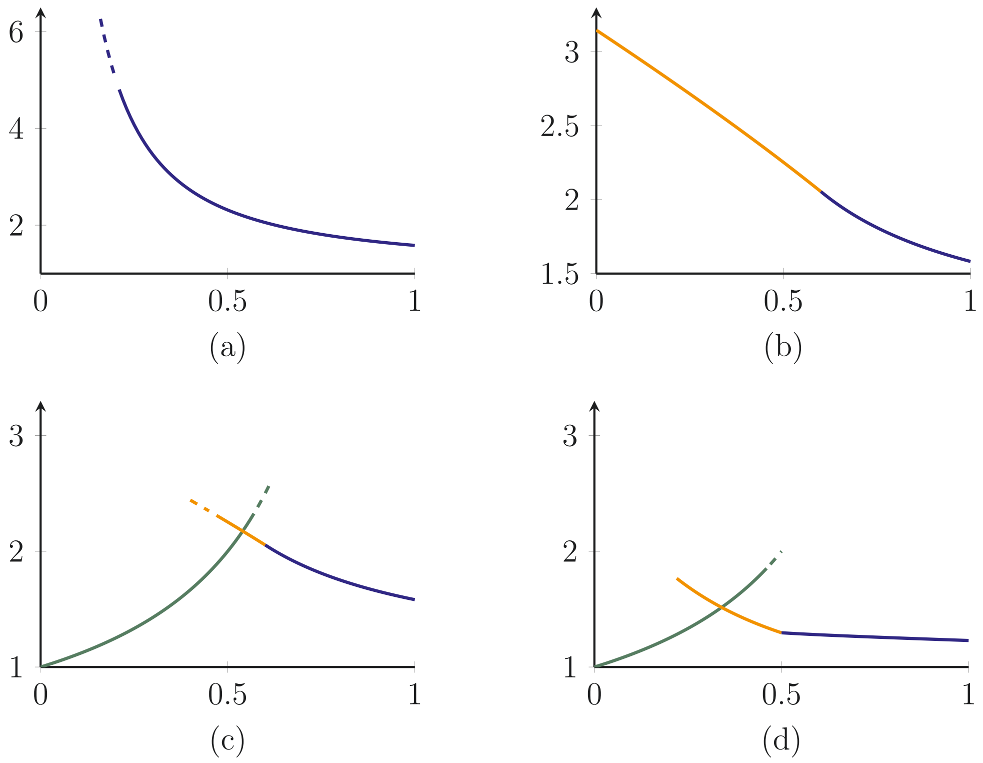


Figure 6.1: Overview of our upper bounds on the PoA (y-axis) for γ-FPA and γ-HYA, respectively, as a function of γ (x-axis). (a) CCE-PoA for multi-unit γ-FPA with overbidding (Theorem 6.4.1). (b) CCE-PoA for multi-unit γ-FPA without overbidding (Theorems 6.4.1 & 6.5.3). (c) CCE-PoA for single-item γ-HYA without overbidding (Theorems 6.4.1, 6.5.3 & 6.5.5). (d) CCE-PoA for single-item γ-HYA without overbidding and $n = 2$ bidders (Theorems 6.5.5, 6.5.6 & 6.5.7).

rule (formalized below). The γ-HYA also belongs to this class. Not only does this capture more elaborate bribing schemes, it also handles the situation where some bidders have moral objections against partaking in such a scheme and do not accept the bribe. Additionally, this also enables us to capture corruption schemes with *heterogeneous bidders*, i.e., where the auctioneer handles a different γ_i for each bidder i.

In our view, the corruption settings described above serve as suitable motivations to analyze the resulting auctions γ-HYA and γ-FPA. But, at the same time, we feel that the study of such *hybrid* auction formats is interesting in its own right, purely from an auction design perspective. For example, tight bounds on the price of anarchy (as a function of γ) provide insights on which payment rule should ideally be used to reduce the inefficiency.

6.1.2 Our Contributions

We study the inefficiency of equilibria of γ-FPA and γ-HYA, both in the single-item and multi-unit auction setting. More specifically, our goal is to obtain a precise understanding of the *(robust) price of anarchy* (PoA) (Koutsoupias and Papadimitriou, 1999; Roughgarden, 2015; Syrgkanis and Tardos, 2013). We opt for the price of anarchy notion here because it is one of the most appealing and widely accepted measures to assess the efficiency of equilibria, especially in the context of social welfare analysis. We focus on the analysis of the robust price of anarchy under the complete information setting, incorporating equilibrium notions ranging from pure Nash equilibria (PNE) to coarse correlated equilibria (CCE). Moreover, we analyze the price of anarchy, distinguishing between the cases when bidders can overbid and when they cannot overbid their actual valuations for the items.

The main results that we obtain in this chapter are summarized below (see Figure 6.1 for an overview). Without any restrictions on the bids, we obtain the following result:

1. We prove an upper bound of $(1/\gamma) \cdot e^{1/\gamma}/(e^{1/\gamma} - 1)$ on the coarse correlated PoA (CCE-PoA) of any γ-FPA in the multi-unit auction setting when bidders can overbid; see Figure 6.1(a). Our upper bound follows from a suitable adaptation of the smoothness technique for multi-unit auctions (de Keijzer et al., 2013; Syrgkanis and Tardos, 2013). Further, by means of a single-item γ-HYA, we prove a matching lower bound over the entire range $\gamma \in [0, 1]$. As a result, our bound settles the CCE-PoA of γ-FPA exactly for both the single-item and multi-unit auction setting over the entire range of $\gamma \in [0, 1]$.

A standard assumption that often needs to be made to derive meaningful bounds on the POA is that the bidders cannot overbid (see also related work section). Under the no-overbidding assumption, a more fine-grained landscape of the price of anarchy emerges:

2. We show that the pure PoA (PNE-PoA) of γ-HYA in the multi-unit auction setting is 1 for $\gamma \in (0, 1)$. This result is complemented by PNE-PoA $= 2.1885$ for $\gamma = 0$ (Birmpas et al., 2019) and PNE-PoA $= 1$ for $\gamma = 1$ (de Keijzer et al., 2013). Notice that this reveals an interesting transition at $\gamma = 0$.

3. We prove that the CCE-PoA of any γ-FPA in the multi-unit auction setting is upper bounded by
$$-(1-\gamma)\mathcal{W}_{-1}\left(-\frac{1}{e^{(2-\gamma)/(1-\gamma)}}\right),$$
for $\gamma \lessapprox 0.607$ where $\mathcal{W}$ is the Lambert-$\mathcal{W}$ function. Combined with our upper bound (first contribution above) for $\gamma > 0.607$ (i.e. with overbidding), we obtain the combined bound depicted in Figure 6.1(b).

4. We prove that the correlated PoA (CE-PoA) of γ-HYA in the single-item auction setting is 1 for every $\gamma \in (0, 1)$. This result together with CE-PoA = 1 for $\gamma = 1$ (Feldman, Lucier, and Nisan, 2016) and our next result, shows that CE-PoA = 1 for the entire range $\gamma \in [0, 1]$.

5. We show that the CCE-PoA of γ-HYA in the single-item auction setting with n bidders is bounded as indicated in Figure 6.1(c). Concretely, we prove an upper bound of $1/(1 - \gamma)$ and combine it with the multi-unit bounds from Figure 6.1(b).

6. We show that the CCE-PoA of γ-HYA in the single-item auction setting with $n = 2$ bidders is bounded as indicated in Figure 6.1(d). This bound is derived by combining three different upper bounds, one of which the $1/(1 - \gamma)$ bound from Figure 6.1(c). Technically, this is the most challenging part of the chapter as we use the cumulative distribution functions (CDF) of equilibrium bids directly to derive these bounds.

Implications

Altogether, our bounds provide a rather complete picture of the PoA of γ-FPA and, in particular, γ-HYA, for different equilibrium notions both in the single-item and the multi-unit auction setting and with and without overbidding. If the bidders can overbid, then our (tight) bound on the CCE-PoA (Figure 6.1(a)) shows that the PoA increases from a small constant $e/(e - 1)$ to infinity as γ decreases from 1 to 0. Thinking about γ-HYA, we feel that this makes sense intuitively. As γ approaches 0, the auctioneer only withholds a small fraction of the surplus and the bidders are thus incentivized to exploit the corruption (as it comes at a low cost). In contrast, as γ approaches 1, the auctioneer charges a significant fraction of the surplus, and while the bidders still have good reasons to participate in the corruption (explained below), they exploit it less drastically as it comes at a large cost.

Our bounds reveal that there is a substantial difference in the PoA depending on whether or not bidders can overbid; e.g., compare the bounds depicted in (a) and (b) (multi-unit setting), or (a) and (c) (single-item setting) in Figure 6.1. In general, it is not well-understood how the no-overbidding assumption influences the PoA of auctions; this question also relates to the *price of undominated anarchy* studied by Feldman, Lucier, and Nisan (2016) (see related work below). Our bounds shed some light on this question for γ-FPA.

Technical Merits

Our upper bounds for γ-FPA are based on an adapted smoothness notion that relates directly to the highest marginal winning bids (i.e., first-price payments). In particular, our smoothness argument does *not* exploit the second-price payments of γ-HYA at all. As it turns out, this allows us to derive *tight* bounds for γ-HYA and, more generally, for γ-FPA when bidders can overbid. On a high level, our results thus reveal that the (approximate) first-price payments are the determining component of such composed payment schemes. This triggers some interesting questions for future research.

In contrast, when overbidding is not allowed it becomes crucial to exploit the second-price payments of γ-HYA to obtain improved bounds. The price of anarchy of both the first-price auction and the second-price auctions is well understood in the single-item setting. However, it is not straightforward to extend these bounds to the combined payment scheme of γ-HYA. In fact, to prove our bounds in Theorem 6.5.6 and Theorem 6.5.7, we exploit constraints on the CDF of the first-price payments which are imposed by the CCE conditions; but, additionally, we have to get a grip on the CDF of the second-price payments. We need several new insights (and a somewhat involved numerical analysis) to derive these bounds.

Extensions

Although we focus on the complete information setting in this chapter, most of our bounds can be lifted to the incomplete information setting as introduced by Harsanyi (1967), where players have private valuation functions drawn from a common prior. Several of our upper bounds are based on an adapted smoothness approach for multi-unit auctions, which extends (basically) directly to this incomplete information setting and (mixed) Bayes-Nash equilibria.[2] All bounds displayed in Figure 1(a–c) remain valid for Bayes-Nash equilibria as well.

6.1.3 Related Work

There is a large body of research in economics studying collusion among bidders in auctions (see, e.g., (Graham and Marshall, 1987; McAfee and McMillan, 1992) for some standard references). Collusion between the auctioneer and the bidders in the form of bid rigging (as considered in this chapter) has also been studied in the literature, but less intensively. Most existing works study certain aspects of equilibrium outcomes (e.g., equilibrium structure, auctioneer surplus, seller revenue, optimal bribe schemes, etc.); for an overview of the existing works along these lines, see (Lengwiler and Wolfstetter, 2010; Lengwiler and Wolfstetter, 2000; Menezes and Monteiro, 2006) and the references therein.

The specific bid-rigging model that we consider here was first studied by Menezes and Monteiro (2006) and a slight generalization thereof by Lengwiler and Wolfstetter (2000), both for the single-item auction setting. These works consider a Bayesian setting where the valuations are independent draws from a

[2] More specifically, these extensions can be proven along similar lines of arguments as in (de Keijzer et al., 2013), where smoothness is used to bound the Bayes-Nash PoA of (standard) multi-unit auctions.

common distribution function. Menezes and Monteiro (2006) prove the existence of symmetric equilibrium bidding strategies and derive an optimal bribe function for the auctioneer. The authors also study a *fixed-price* bribe scheme, where the auctioneer charges a fixed amount that is independent of the gained surplus.

Subsequently, Lengwiler and Wolfstetter (2010) study a more complex bid-rigging scheme for the single-item auction setting, where the auctioneer additionally offers the second-highest bidder to increase their bid. To the best of our knowledge, none of the existing works studied the price of anarchy of corrupt auctions.

Studying the price of anarchy in auctions has recently received a lot of attention; we refer to the survey paper by Roughgarden, Syrgkanis, and Tardos (2017) for an overview. A lot of work has gone into deriving bounds on the price of anarchy for various auction formats, both in the complete and incomplete information setting. The *smoothness notion*, originally introduced by Roughgarden (2015) to analyze the robust price of anarchy of strategic games, turned out to be very useful in an auction context as well. Syrgkanis and Tardos (2013) build upon this notion and provide a powerful (smoothness-based) toolbox for the analysis of a broad range of auctions that fall into their composition framework.

With respect to the multi-unit auction setting, de Keijzer et al. (2013) use an adapted smoothness approach to derive bounds on the PoA of Bayes-Nash equilibria for the first-price and the second-price multi-unit auction (mostly focusing on the setting with no overbidding). Our bounds coincide with theirs for the extreme points $\gamma = 0$ and $\gamma = 1$. For the more general class of subadditive valuations, the PoA of Bayes-Nash equilibria for the first-price multi-unit auction is 2, which follows from (de Keijzer et al., 2013) and (Christodoulou et al., 2016). Birmpas et al. (2019) recently settled the PNE-PoA of the second-price multi-unit auction and show that it is 2.1885.

Our bounds on the CCE-PoA are also based on a smoothness approach. We use an adapted smoothness notion (inspired by (de Keijzer et al., 2013; Syrgkanis and Tardos, 2013)) to derive our bounds, both in the overbidding and the no-overbidding setting. Interestingly, our smoothness proofs crucially exploit that the payments recover at least a fraction of γ of the first-price payments (but never exceed them). As a side result, Syrgkanis and Tardos (2013) also derive a first bound on the CE-PoA for γ-HYA in the single-item auction setting; our bound (significantly) improves on theirs and exploits some additional ideas.

The PoA of the first-price and second-price auction has been investigated intensively for both the single-item and the multi-unit auction setting. An assumption that often needs to be made to derive meaningful bounds is that the bidders cannot overbid. For example, it is folklore that the PNE-PoA of the second-price single-item auction is unbounded if the bidders can overbid. On the other hand, it is one if bidders cannot overbid. In the second-price single-item auction, overbidding is a dominated strategy for each bidder, and the no-overbidding assumption thus emerges naturally. But this might not be true in general. For example, for the second-price multi-unit auction, this analogy breaks already. We refer to Feldman, Fu, et al. (2013) for a more general discussion of the no-overbidding assumption.

In general, the impact that the no-overbidding assumption has on the price of anarchy is not well-understood. This aspect also relates to the *price of undominated anarchy* studied by Feldman, Lucier, and Nisan (2016). The authors prove a clear separation for the PoA in single-item first-price auctions: While the CE-PoA is 1 (even with overbidding), the CCE-PoA increases to 1.229 (without overbidding) and $e/(e-1)$ (with overbidding). A similar separation holds for the multi-unit auction setting and the uniform price auction, where the PNE-PoA is $(e-1)/e$ (without overbidding) (Markakis and Telelis, 2015) and 2.1885 (with overbidding) (Birmpas et al., 2019). Our results contribute to this line of research also because we show that the PoA might improve significantly under the no-overbidding assumption.

6.2 Preliminaries

6.2.1 Standard Auction Formats

We focus on the description of the multi-unit auction setting; the single-item auction setting follows as a special case (choosing $k = 1$ below). In the multi-unit auction setting, there are $k \geq 1$ identical items (or goods) that we want to sell to $n \geq 2$ bidders (or players). We identify the set of bidders N with $[n] = \{1, \dots, n\}$. Each bidder i has a non-negative and non-decreasing valuation function $v_i : \{0, \dots, k\} \to \mathbb{R}_{\geq 0}$ with $v_i(0) = 0$, where $v_i(j)$ specifies i's valuation for receiving j items. We assume that for each bidder $i \in N$ the valuation function v_i is *submodular* or, equivalently, that the marginal valuations are non-increasing, i.e., for every $j \in [k-1]$, $v_i(j) - v_i(j-1) \geq v_i(j+1) - v_i(j)$. The valuation function v_i is assumed to be private information, i.e., it is only known to bidder i themselves. We use $\boldsymbol{v} = (v_1, \dots, v_n)$ to denote the profile (or vector) of the valuation functions of the bidders. We assume that the bidders submit their bids according to the following *standard format*: Each bidder i submits a bid vector $\boldsymbol{b}_i = (b_i(1), \dots, b_i(k))$ of k non-negative and non-increasing *marginal bids*, i.e., $b_i(j)$ specifies the additional amount i is willing to pay for receiving j instead of $j-1$ items. The overall amount that i bids for receiving q items is thus $\sum_{j=1}^{q} b_i(j)$. For $k = 1$ we write $b_i = b_i(1)$.

Consider a multi-unit auction setting and suppose the auctioneer uses an *auction mechanism* $\mathcal{M}$ to determine an assignment of the items and the respective payments of the bidders. Each bidder submits their bid vector $\boldsymbol{b}_i$ to the mechanism. Based on the bidding profile $\boldsymbol{b} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n)$, the mechanism $\mathcal{M}$ orders the submitted marginal bids non-increasingly (breaking ties in an arbitrary but consistent way) and assigns the k items to the bidders who submitted the k highest marginal bids (according to this order). We use $\beta_j(\boldsymbol{b})$ to refer to the j-th lowest winning (marginal) bid in $\boldsymbol{b}$, i.e., $\beta_k(\boldsymbol{b}) \geq \ldots \geq \beta_1(\boldsymbol{b})$. We use $\boldsymbol{x}(\boldsymbol{b}) = (x_1(\boldsymbol{b}), \ldots, x_n(\boldsymbol{b}))$ to refer to the resulting allocation, where $x_i(\boldsymbol{b})$ specifies the number of items that bidder i receives; $x_i(\boldsymbol{b}) = 0$ if i does not receive any item. Each bidder i who receives at least one item is called a *winner*.

There are two standard payment schemes that determine for each winner i the respective payment $p_i(\boldsymbol{b})$; we adopt the convention that $p_i(\boldsymbol{b}) = 0$ for each bidder i who is not a winner.

> *First-price payment scheme*: Every bidder i pays their bid for the received items, i.e., $p_i(\boldsymbol{b}) = \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j)$.
>
> *Second-price payment scheme*: Every bidder i pays the highest losing bid $\bar{p}(\boldsymbol{b})$ for each received item, i.e., $p_i(\boldsymbol{b}) = x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b})$.

Suppose we fix the payment scheme of mechanism $\mathcal{M}$ according to one of these schemes. We refer to mechanism $\mathcal{M}$ with the first-price payment or the second-price payment scheme, respectively, as FP-Auction or SP-Auction.[3]

The utility $u_i^{v_i}(\boldsymbol{b})$ of bidder i is defined as the total valuation minus the payment for receiving $x_i(\boldsymbol{b})$ items, i.e., $u_i^{v_i}(\boldsymbol{b}) = v_i(x_i(\boldsymbol{b})) - p_i(\boldsymbol{b})$; note that $u_i^{v_i}(\boldsymbol{b}) = 0$ by definition if bidder i is not a winner. Whenever v_i is clear from the context, we simply denote the utility of bidder i by $u_i(\boldsymbol{b})$. We assume that each bidder strives to maximize their utility.

Finally, recall some standard assumptions that we use throughout this chapter; we adopt the convention that the first two must always be satisfied by a mechanism.

1. *No positive transfers (NPT):* The payment of each bidder i is non-negative, i.e., $p_i(\boldsymbol{b}) \geq 0$.
2. *Individual rationality (IR):* The payment of each bidder i does not exceed their bid, i.e., $p_i(\boldsymbol{b}) \leq \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j)$.
3. *No overbidding (NOB):* The bid vector of each bidder i does not exceed their valuations, i.e., for every $q \in [k]$, $\sum_{j=1}^{q} b_i(j) \leq v_i(q)$.

6.2.2 Approximate First-Price Auctions

In this chapter, we also consider auctions with first-price approximate payment schemes. The allocation is still determined as above, but the payment scheme is relaxed as follows: We say that a mechanism $\mathcal{M}$ with payment rule $\boldsymbol{p} = (p_1(\boldsymbol{b}), \ldots, p_n(\boldsymbol{b}))$ is a *γ-approximate first-price auction (γ-FPA)* for some $\gamma \in [0, 1]$ if it always recovers at least a fraction of γ of the first-price payments, i.e., for every bidding profile $\boldsymbol{b}$, $\sum_{i \in N} p_i(\boldsymbol{b}) \geq \gamma \sum_{j=1}^{k} \beta_j(\boldsymbol{b})$. Further, if for every bidding profile

[3] We remark that in the multi-unit auction setting these auctions are usually referred to as *discriminatory price auction* and *uniform price auction*; however, here we stick to the given naming convention to align it with the common terminology of the single-item auction setting.

$\boldsymbol{b}$ it holds that $\sum_{i\in N} p_i(\boldsymbol{b}) \le \sum_{j=1}^{k} \beta_j(\boldsymbol{b})$ then we call the mechanism *first-price dominated.* Note that every mechanism that satisfies individual rationality must be first-price dominated.

6.2.3 Equilibrium Notions and the Price of Anarchy

In this subsection, we briefly recall the definitions of the different equilibrium notions, the social welfare, and the price of anarchy that are used in this chapter.

Equilibrium Notions

A bidding profile $\boldsymbol{b} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n)$ is a *pure Nash equilibrium (PNE)* if no bidder has an incentive to deviate unilaterally; more formally, $\boldsymbol{b}$ is a PNE if for every bidder i and every bidding profile $\boldsymbol{b}'_i$ of i it holds that $u_i(\boldsymbol{b}) \ge u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})$. Here we use the standard notation $\boldsymbol{b}_{-i}$ to refer to the bid vector $\boldsymbol{b}$ with the ith component being removed; $(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})$ then refers to the bid vector $\boldsymbol{b}$ with the ith component being replaced by $\boldsymbol{b}'_i$.

We also consider randomized bid vectors. Suppose bidder i chooses their bid vectors randomly according to a probability distribution $\boldsymbol{\sigma}_i$, independently of the other bidders. Let $\boldsymbol{\sigma} = \prod_{i\in N} \boldsymbol{\sigma}_i$ be the respective product distribution. Then $\boldsymbol{\sigma}$ is a *mixed Nash equilibrium (MNE)* if for every bidder i and every bid vector $\boldsymbol{b}'_i$ it holds that $\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b})] \ge \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})]$. We may also allow correlation among the bidders. Let $\boldsymbol{\sigma}$ be a joint distribution over bidding profiles of the bidders. Then $\boldsymbol{\sigma}$ is a *correlated equilibrium (CE)* if for every bidder $i \in N$ and for every deviation function $m_i(\boldsymbol{b}_i)$ it holds that $\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b})] \ge \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(m_i(\boldsymbol{b}_i), \boldsymbol{b}_{-i})]$. Intuitively, conditional on bid vector $\boldsymbol{b}_i$ being realized, i has no incentive to deviate to any other bid vector $m_i(\boldsymbol{b}_i)$. The most general equilibrium notion that we consider in this chapter is defined as follows: Let $\boldsymbol{\sigma}$ be a joint distribution over bidding profiles of the bidders. Then $\boldsymbol{\sigma}$ is a *coarse correlated equilibrium (CCE)* if for every bidder i and every bid vector $\boldsymbol{b}'_i$ it holds that $\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b})] \ge \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})]$. Below, we also use PNE($\boldsymbol{v}$), MNE($\boldsymbol{v}$), CE($\boldsymbol{v}$) and CCE($\boldsymbol{v}$) to refer to the sets of pure, mixed, correlated and coarse correlated equilibria with respect to a valuation profile $\boldsymbol{v} = (v_1, \dots, v_n)$, respectively.

Inefficiency of Equilibria

We define the *social welfare* of a bidding profile $\boldsymbol{b} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n)$ as the overall valuation obtained by the bidders, i.e., $\mathrm{SW}(\boldsymbol{b}) = \sum_{i\in N} v_i(x_i(\boldsymbol{b}))$. Note that although social welfare is defined independently of the payments, we can equivalently write $SW(\boldsymbol{b}) = \sum_{i\in N} u_i(\boldsymbol{b}) + p_i(\boldsymbol{b})$. The expected social welfare of a joint distribution $\boldsymbol{\sigma}$ over bidding profiles is then defined as $\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] = \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[\mathrm{SW}(\boldsymbol{b})]$. We use $\boldsymbol{x}^*(\boldsymbol{v})$ to refer to an assignment that maximizes the social welfare with respect to the valuation functions $\boldsymbol{v} = (v_1, \dots v_n)$; i.e., $\mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v})) = \sum_{i\in N} v_i(x^*_i(\boldsymbol{v}))$ is the maximum social welfare achievable for the bidders. The assignment $\boldsymbol{x}^*(\boldsymbol{v})$ is also called a *social optimum.*

The *price of anarchy* is defined as the maximum ratio of the social welfare of the social optimum and the (expected) social welfare of an equilibrium. Let X be a placeholder that refers to one of the equilibrium notions above, i.e., $X \in \{\text{PNE}, \text{MNE}, \text{CE}, \text{CCE}\}$. More formally, given a valuation profile $\boldsymbol{v} = (v_1, \dots, v_n)$, the *price of anarchy* with respect to X (or X-PoA for short) is defined as $X\text{-PoA}(\boldsymbol{v}) = \sup_{\boldsymbol{\sigma} \in X(\boldsymbol{v})} \text{SW}(\boldsymbol{x}^*(\boldsymbol{v}))/\mathbb{E}[\text{SW}(\boldsymbol{\sigma})]$. The price of anarchy of an auction format then refers to the worst-case price of anarchy over all possible valuation profiles, i.e., $X\text{-PoA} = \sup_{\boldsymbol{v}} X\text{-PoA}(\boldsymbol{v})$. We use PNE-PoA, MNE-PoA, CE-PoA and CCE-PoA to refer to the respective price of anarchy notions.

6.3 Capturing Corruption with γ-FPA

We give a formal description of the model that we consider and elaborate on its relation to the γ-hybrid auction. We also introduce the adapted smoothness approach.

6.3.1 Corruption in Auctions

Suppose the bidders submit their bid vectors $\boldsymbol{b} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n)$ in a “sealed manner”, i.e., at first only the auctioneer sees the bidding profile $\boldsymbol{b}$.[4] After receipt of the bidding profile $\boldsymbol{b}$, the auctioneer runs a first-price multi-unit auction (see Section 6.2) to obtain the respective assignment $\boldsymbol{x}(\boldsymbol{b}) = (x_1(\boldsymbol{b}), \dots, x_n(\boldsymbol{b}))$ and payments $\boldsymbol{p}(\boldsymbol{b}) = (p_1(\boldsymbol{b}), \dots, p_n(\boldsymbol{b}))$ but does not reveal this outcome yet. The auctioneer then approaches each winning bidder i individually with the offer that they can lower all their $x_i(\boldsymbol{b})$ winning bids to the highest losing bid $\bar{p}(\boldsymbol{b})$ (while receiving the same number of items), in exchange for a fixed fraction $\gamma \in [0, 1]$ of the surplus gained by i. The bidder can either reject or accept this offer. If bidder i rejects the offer, the allocation $x_i(\boldsymbol{b})$ and respective payment $p_i(\boldsymbol{b})$ remain unmodified. If bidder i accepts the offer, they receive the $x_i(\boldsymbol{b})$ items at a reduced price of $\bar{p}(\boldsymbol{b})$ each, but additionally pay a fee f_i^γ of γ times the surplus to the auctioneer; more formally, the total payment of a winning bidder i who accepts the offer is

$$p_i^\gamma(\boldsymbol{b}) = x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b}) + f_i^\gamma(\boldsymbol{b}) \ \text{ where } \ f_i^\gamma(\boldsymbol{b}) = \gamma \sum_{j=1}^{x_i(\boldsymbol{b})} (b_i(j) - \bar{p}(\boldsymbol{b})).$$

[4] It is important to realize though that the final bids, which might not necessarily correspond to the submitted ones, might have to be revealed eventually because the bidders might want to verify the “soundness” of the outcome of the auction.

We also refer to this setting as the γ-*corrupt auction.*[5]

Note that the change in the bid vector of player i conforms to the imposed bidding format, i.e., the modified marginal bids of bidder i are still non-negative and non-increasing.

Dominant Strategy to Accept Offer

We show that it is a dominant strategy for every winning bidder to always accept the offer of the auctioneer (independent of γ).

Proposition 6.3.1. *Fix some $\gamma \in [0, 1]$ and consider a γ-corrupt auction. We can assume without loss of generality that each winning bidder always accepts the offer of the auctioneer.*

Proof: Observe that the total payment to be made by a winning bidder i who accepts the offer becomes

$$p_i^\gamma(\boldsymbol{b}) = x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b}) + f_i^\gamma(\boldsymbol{b}) = \gamma \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j) + (1-\gamma)x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b}). \tag{6.1}$$

Clearly, each winning bid j of i satisfies $b_i(j) \geq \bar{p}(\boldsymbol{b})$. Thus, $p_i^\gamma(\boldsymbol{b}) \leq \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j) = p_i(\boldsymbol{b})$, where $p_i(\boldsymbol{b})$ is the payment that i would have to pay when rejecting the offer. In fact, this inequality is strict unless all winning bids of i are equal to $\bar{p}(\boldsymbol{b})$ or $\gamma = 1$. In both these cases, the offer made by the auctioneer does not have any effect for i (as there is no surplus generated in the former case and no difference in the final payment of i in the latter case). Said differently, each winning bidder can only benefit from accepting the offer. Observe also that the above arguments hold for every winning bidder independently of what the other bidders do. Further, the final allocation remains invariant (assuming a consistent tie-breaking rule). We conclude that it is a dominant strategy for every winning bidder to accept the offer of the auctioneer. □

Subsequently, we assume that each winning bidder always accepts the offer.

6.3.2 Hybrid Auction Scheme

We introduce our novel hybrid auction scheme, which we term γ-*hybrid auction* (or γ-HYA for short): γ-HYA uses the same allocation rule as in the multi-unit auction setting (see Section 6.2), but uses a convex combination of the first-price and second-price payment scheme (parameterized by γ), i.e.,

$$p_i^\gamma(\boldsymbol{b}) = \gamma \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j) + (1-\gamma)x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b}). \tag{6.2}$$

[5] As the final payments are dependent on γ, we (implicitly) assume that the bidders are aware of this parameter when considering the complete information setting here (much alike it is assumed that the bidders know the used payment scheme in other auction formats).

Said differently, γ-HYA interpolates between SP-Auction ($\gamma = 0$) and FP-Auction ($\gamma = 1$) as γ varies from 0 to 1. It is immediate that every γ-HYA is a γ-FPA. We also use $p^\gamma(\boldsymbol{b})$ to refer to the above payment in the single-item auction setting.

The following proposition follows immediately from the discussion in section 6.3.1 and allows us to focus on the PoA of γ-HYA to study γ-corrupt auctions.

Proposition 6.3.2. *Fix some $\gamma \in [0, 1]$. Then the γ-corrupt auction and* γ-HYA *admit the same set of equilibria and have identical social welfare objectives. Therefore, the price of anarchy for both these settings is the same.*

6.3.3 Other Corruption Models

In our basic bid-rigging model introduced above all winning bidders lower their bids to the highest losing bid. While this magic number bidding phenomenon has been observed in real-life for single-item auctions (as mentioned in the introduction), it might seem somewhat awkward in the multi-unit auction setting. We, therefore, consider more general corruption schemes that also capture non-uniform bid rigging.

A Non-Uniform Bid-Rigging Scheme for γ-HYA

As before, the bidders submit their bid vectors $\boldsymbol{b} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n)$ to the auctioneer who runs a first-price multi-unit auction. The auctioneer then approaches each winning bidder i individually with the offer that they can lower their $x_i(\boldsymbol{b})$ winning bids. However, in contrast to the basic model, the auctioneer and bidder i agree to "camouflage" their bid rigging by bidding the highest losing bid $\bar{p}(\boldsymbol{b})$ plus a fraction $\alpha \in [0, 1]$ of the surplus $b_i(j) - \bar{p}(\boldsymbol{b})$ for each $j \in [x_i(\boldsymbol{b})]$. Note that this maintains the relative order among the winning bids and the magic number cheating becomes less obvious (as the winning bids fluctuate more). The remaining surplus of $(1 - \alpha)(b_i(j) - \bar{p}(\boldsymbol{b}))$ is then split, where the auctioneer withholds a fraction of $\beta \in [0, 1]$. As before, bidder i can either reject or accept the offer. But, also here, it is not hard to see that accepting the offer is a dominant strategy. The total payment of a winning bidder i is then

$$p_i^{(\alpha,\ \beta)}(\boldsymbol{b}) = \sum_{j=1}^{x_i(\boldsymbol{b})} (\bar{p}(\boldsymbol{b}) + \alpha(b_i(j) - \bar{p}(\boldsymbol{b}))) + f_i^{(\alpha,\ \beta)}(\boldsymbol{b}), \quad \text{where}$$

$$f_i^{(\alpha,\ \beta)}(\boldsymbol{b}) = \beta \sum_{j=1}^{x_i(\boldsymbol{b})} (1 - \alpha)(b_i(j) - \bar{p}(\boldsymbol{b})).$$

After simplifying, we obtain

$$p_i^{(\alpha,\,\beta)}(\boldsymbol{b}) = (\alpha + \beta(1-\alpha)) \sum_{j=1}^{x_i(\boldsymbol{b})} b_i(j) + (1 - \alpha - \beta(1-\alpha))x_i(\boldsymbol{b})\bar{p}(\boldsymbol{b}).$$

If we define $\gamma = \alpha + \beta - \alpha\beta$, the above payments $p_i^{(\alpha,\,\beta)}$ are equivalent to p_i^γ as defined in (6.2). Note also that this mapping satisfies $\gamma \in [0,1]$ for every $\alpha, \beta \in [0,1]$. Said differently, given $\alpha, \beta \in [0,1]$ the price of anarchy of the above non-uniform bid-rigging scheme is determined by the price of anarchy of γ-HYA with $\gamma = \alpha + \beta - \alpha\beta$.

Non-Uniform Bid Rigging Schemes for γ-FPA

Most of our upper bounds hold for the more general class of γ-FPA. These auctions capture several additional corruption settings. For example, suppose some bidders never accept the offer of the auctioneer (say due to moral objections) and their payments thus remain the first-price payments. While this setting is not covered by γ-HYA, it is covered by γ-FPA. As another example, if the auctioneer handles a different fraction γ_i for each bidder i, the resulting auction is γ-FPA with $\gamma = \min_{i \in N} \gamma_i$.

6.3.4 Adapted Smoothness Notion

We introduce our adapted smoothness notion (based on the ones given in (de Keijzer et al., 2013; Syrgkanis and Tardos, 2013)) to derive upper bounds on the coarse correlated price of anarchy of γ-hybrid auction.

Recall that for a bidding profile $\boldsymbol{b}$, we let $\beta_j(\boldsymbol{b})$ refer to the jth lowest winning bid under $\boldsymbol{b}$.

Definition 6.3.3. A mechanism $\mathcal{M}$ for the multi-unit auction setting is (λ, μ)-*smooth* for some $\lambda > 0$ and $\mu \geq 0$ if for every valuation profile $\boldsymbol{v}$ and for each bidder $i \in N$ there exists a (possibly randomized) deviation $\boldsymbol{\sigma}_i'$ such that for every bidding profile $\boldsymbol{b}$ we have

$$\sum_{i \in N} \mathbb{E}_{\boldsymbol{b}_i' \sim \boldsymbol{\sigma}_i'} [u_i(\boldsymbol{b}_i', \boldsymbol{b}_{-i})] \geq \lambda \mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v})) - \mu \sum_{j=1}^{k} \beta_j(\boldsymbol{b}).$$

In essence, this definition comes close to the *weak smoothness* definition in (Syrgkanis and Tardos, 2013), but relates more directly to the winning bids in the multi-unit auction setting. A similar definition is also used in (de Keijzer et al., 2013), but there it is imposed on a per-player basis and used for the Bayesian setting.

Theorem 6.3.4. *Let $\mathcal{M}$ be a γ-*FPA *which is (λ, μ)-smooth. Then* CCE-PoA $\leq \max\{1, 1 + \mu - \gamma\}/\lambda$, *where we need that the no-overbidding assumption holds if $\mu > \gamma$.*

Proof: Fix a valuation profile $\boldsymbol{v}$ and let $\boldsymbol{\sigma}$ be a coarse correlated equilibrium. Consider some player i and let $\boldsymbol{\sigma}'_i$ be the (randomized) deviation of bidder i as given by the smoothness definition. Exploiting the coarse correlated equilibrium condition for i, we have for every (deterministic) bid vector $\boldsymbol{b}'_i$ that $\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b})] \geq \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})]$ and thus also

$$\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_i(\boldsymbol{b})] \geq \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[\mathbb{E}_{\boldsymbol{b}'_i\sim\boldsymbol{\sigma}'_i}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})]]. \tag{6.3}$$

Using this, we obtain

$$\begin{aligned}\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &= \sum_{i\in N} \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[u_i(\boldsymbol{b}) + p_i(\boldsymbol{b})\right] &(6.4)\\ &\geq \sum_{i\in N} \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[\mathbb{E}_{\boldsymbol{b}'_i\sim\boldsymbol{\sigma}'_i}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})] + p_i(\boldsymbol{b})\right]\\ &\geq \lambda\mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v})) + (\gamma - \mu)\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[\sum_{j=1}^{k} \beta_j(\boldsymbol{b})\right], &(6.5)\end{aligned}$$

where the first inequality follows from (6.3) and the second inequality holds because of the smoothness definition and because $\mathcal{M}$ is first-price γ-approximate.

We distinguish two cases:

Case 1: $\mu \leq \gamma$. Using (6.5), we obtain $\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] \geq \lambda\mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v}))$ and thus $\mathrm{PoA}(\boldsymbol{v}) \leq 1/\lambda$.

Case 2: $\mu > \gamma$. Exploiting that the no-overbidding assumption holds in this case, we get that $\sum_{j=1}^{k} \beta_j(\boldsymbol{b}) \leq \sum_{i\in N} v_i(x_i(\boldsymbol{b}))$. Using (6.5), we obtain $\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] \geq \lambda\mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v})) + (\gamma - \mu)\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})]$. Rearranging terms yields $\mathrm{PoA}(\boldsymbol{v}) \leq (1 + \mu - \gamma)/\lambda$. Combining both cases proves the claim. □

We will use the above smoothness definition in combination with the following lemma, which we import from (de Keijzer et al., 2013) (adapted to our setting).

Lemma 6.3.5. (Lemma 3 in (de Keijzer et al., 2013)). *Let $\mathcal{M}$ be a mechanism that is first-price dominated and let $\alpha > 0$ be fixed arbitrarily. Then for every valuation profile $\boldsymbol{v}$ and for every bidder i there exists a randomized deviation $\boldsymbol{\sigma}'_i$ such that for every bidding profile $\boldsymbol{b}$ we have*

$$\mathbb{E}_{\boldsymbol{b}'_i\sim\boldsymbol{\sigma}'_i}[u_i(\boldsymbol{b}'_i, \boldsymbol{b}_{-i})] \geq \alpha\left(1 - \frac{1}{e^{1/\alpha}}\right) v_i(\boldsymbol{x}^*_i(\boldsymbol{v})) - \alpha \sum_{j=1}^{\boldsymbol{x}^*_i(\boldsymbol{v})} \beta_j(\boldsymbol{b}). \tag{6.6}$$

We can now prove Theorem 6.3.6.

Theorem 6.3.6. *Let $\alpha > 0$ be fixed arbitrarily. The coarse correlated price of anarchy of any γ-FPA is*

$$\text{CCE-PoA} \leq \frac{\max\{1, 1+\alpha-\gamma\}}{\alpha(1-e^{-1/\alpha})}, \tag{6.7}$$

where we need that the no-overbidding assumption holds if $\alpha > \gamma$.

Proof: We use both Lemma 6.3.5 and Theorem 6.3.4.

Note that $\sum_{i \in N} \sum_{j=1}^{\boldsymbol{x}_i^*(\boldsymbol{v})} \beta_j(\boldsymbol{b}) \leq \sum_{j=1}^{k} \beta_j(\boldsymbol{b})$. Hence, by summing inequality (6.6) over all players, we obtain that the mechanism is $(\alpha(1-e^{-1/\alpha}), \alpha)$-smooth. The claimed bound now follows from Theorem 6.3.4. □

6.4 Overbidding

We derive a tight bound on the coarse correlated price of anarchy of γ-FPA for $\gamma > 0$ in the multi-unit auction setting when bidders can overbid. Interestingly, tightness is already achieved by a single-item γ-HYA. It is known that the price of anarchy is unbounded for SP-Auction ($\gamma = 0$). The bound is displayed in Figures 6.1(a) and 6.2. We give a sketch of the proof of Theorem 6.4.1 below.

Theorem 6.4.1. *Consider a multi-unit γ-FPA and suppose that bidders can overbid. For $\gamma \in (0, 1]$, the coarse correlated price of anarchy is* CCE-PoA $\leq \frac{1}{\gamma(1-e^{-1/\gamma})}$. *Further, this bound is tight, even for single-item γ-HYA.*

Proof:

Upper bound: This bound is based on Theorem 6.3.6. Since bidders can overbid in this setting, we restrict to the part of equation (6.7) that does not require the no-overbidding assumption, namely $\alpha \leq \gamma$, with

$$\text{CCE-PoA} \leq \frac{1}{\alpha\,(1-e^{-1/\alpha})}.$$

To minimize this upper bound for any given $\gamma \in [0, 1]$, consider its derivative with respect to α,

$$-\frac{1}{\alpha^2(1-e^{-1/\alpha})^2}\left(1 - e^{-1/\alpha} - \alpha\frac{1}{\alpha^2}e^{-1/\alpha}\right) = -\frac{1-(1+\frac{1}{\alpha})e^{-1/\alpha}}{\alpha^2(1-e^{-1/\alpha})^2}.$$

As $(1 + 1/\alpha)e^{-1/\alpha} < 1$ for all $\alpha > 0$, the derivative is negative for all $\alpha > 0$. Therefore, the bound is minimized by maximizing $\alpha \in (0, \gamma]$. Substituting $\alpha = \gamma$ for any $\gamma \in (0, 1]$ yields the upper bound.

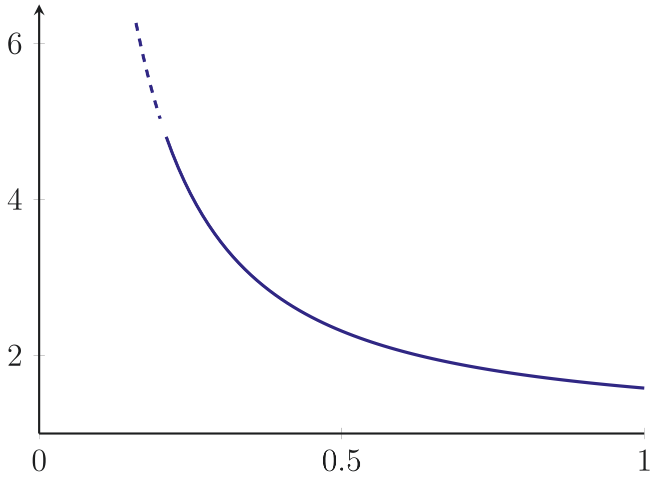


Figure 6.2: CCE-PoA for multi-unit γ-FPA with overbidding (Theorem 6.4.1)

Tight lower bound: This bound can be proven to be tight for all $\gamma \in (0, 1]$ by generalizing an example used by Syrgkanis (2014) to provide a lower bound on the CCE-PoA for the first-price single-item auction: Consider a single-item auction with two bidders and using the γ-hybrid pricing rule as defined above. We have $v_1 = v$ for some $v > 0$ and $v_2 = 0$. If both bidders bid 0, the tie is broken in favor of bidder 2, whereas bidder 1 wins the auction if bidders tie with any positive bid. We construct a coarse correlated equilibrium for any $\gamma \in (0, 1]$, with a welfare loss that matches the upper bound.

Let t be a random variable with support $[0, (1 - e^{-1/\gamma})v]$ whose cumulative distribution function (CDF) F and density function f (which is well-defined for any $t \in (0, (1 - e^{-1/\gamma})v])$, respectively, are given as

$$F(t) = (1 - \gamma) + \frac{v}{v - t}\gamma e^{-1/\gamma} \qquad \text{and} \qquad f(t) = \frac{v}{(v - t)^2}\gamma e^{-1/\gamma}.$$

Note that F has an atom at 0 with mass $(1 - \gamma) + \gamma e^{-1/\gamma}$.

Consider a bidding profile $\boldsymbol{\sigma} = (t, t)$. Since ties are broken in favor of bidder 2 for $t = 0$, they win with probability $(1 - \gamma) + \gamma e^{-1/\gamma}$, which yields

$$\frac{\mathrm{SW}(\boldsymbol{x}^*(\boldsymbol{v}))}{\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})]} = \frac{v}{(1 - F(0))v} = \frac{1}{1 - (1 - \gamma) - \gamma e^{-1/\gamma}} = \frac{1}{\gamma\,(1 - e^{-1/\gamma})}.$$

It remains to show that $\boldsymbol{\sigma}$ is a CCE. For bidder 2, this is quite obvious, since they either win by bidding 0, or lose if $t > 0$. Given any positive bid from bidder 1, the payment would be strictly greater than $v_2 = 0$, meaning bidder 2 could never profitably deviate.

For bidder 1, we show that for any $\gamma \in (0, 1]$, any deviation to a fixed bid $b_1 = b$ with $b \in (0, (1 - e^{-1/\gamma})v]$ leads to an expected utility of at most $\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})]$. To start with $\boldsymbol{\sigma}$ itself, note that bidder 1 wins whenever $t > 0$, and since both bidders bid t, we have a payment of $\gamma t + (1-\gamma)t = t$. Recalling that $v_1 = v$, we get

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})] &= \int_0^{(1-e^{-1/\gamma})v} (v-t)f(t)dt \\
&= \int_0^{(1-e^{-1/\gamma})v} \frac{v}{v-t}\gamma e^{-1/\gamma}dt \\
&= v\gamma e^{-1/\gamma}\left[-\ln(v-t)\right]_0^{(1-e^{-1/\gamma})v} \\
&= v\gamma e^{-1/\gamma}\left(\ln(v) - \ln(e^{-1/\gamma}v)\right) \\
&= v\gamma e^{-1/\gamma}\tfrac{1}{\gamma} = ve^{-1/\gamma}.
\end{aligned}$$

If bidder 1 deviates to b, they win the item if $b \geq t$, and for each $t \in (0,b]$ bidder 1 pays $\gamma b + (1-\gamma)t$. Hence, the expected utility of bidder 1 becomes

$$\mathbb{E}_{t\sim F(t)}[u_1(b,t)] = \int_0^b (v - \gamma b - (1-\gamma)t)f(t)dt.$$

To facilitate the calculations, note that

$$\begin{aligned}
\int_0^b tf(t)dt &= \gamma ve^{-1/\gamma}\int_0^b \frac{t}{(v-t)^2}dt \\
&= \gamma ve^{-1/\gamma}\left[\frac{v}{v-t} + \ln(v-t)\right]_0^b \\
&= \left(\frac{v}{v-b} - 1\right)\gamma ve^{-1/\gamma} + \ln\left(\frac{v-b}{v}\right)\gamma ve^{-1/\gamma}
\end{aligned}$$

and

$$\int_0^b f(t)dt = F(b) - F(0) = \left(\frac{v}{v-b} - 1\right)\gamma e^{-1/\gamma}.$$

Using this, we get

$$\begin{aligned}
\mathbb{E}_{t\sim F(t)}[u_1(b,t)] &= (v-\gamma b)\int_0^b f(t)dt - (1-\gamma)\int_0^b tf(t)dt \\
&= (v - \gamma b - (1-\gamma)v)\left(\frac{v}{v-b} - 1\right)\gamma e^{-1/\gamma} \\
&\quad - (1-\gamma)\ln\left(\frac{v-b}{v}\right)\gamma ve^{-1/\gamma} \\
&= \gamma(v-b)\left(\frac{v}{v-b} - 1\right)\gamma e^{-1/\gamma} \\
&\quad - (1-\gamma)\ln\left(\frac{v-b}{v}\right)\gamma ve^{-1/\gamma} \\
&= b\gamma^2 e^{-1/\gamma} - (1-\gamma)\ln\left(\frac{v-b}{v}\right)\gamma ve^{-1/\gamma}.
\end{aligned}$$

Since $0 < b < v$, note that $-\ln\left(\frac{v-b}{v}\right)$ is increasing in b. Since $\gamma \in (0,1]$, this implies the entire function above is increasing in b. Hence, it can be upper bounded by substituting the upper bound of the support: $b = (1-e^{-1/\gamma})v$. This yields

$$\begin{aligned}\mathbb{E}_{t\sim F(t)}[u_1(b,t)] &\leq (1-e^{-1/\gamma})v\gamma^2 e^{-1/\gamma} - (1-\gamma)\ln\left(e^{-1/\gamma}\right)\gamma v e^{-1/\gamma} \\ &= \left((1-e^{-1/\gamma})\gamma^2 + (1-\gamma)\right) v e^{-1/\gamma} \\ &= \left((1-e^{-1/\gamma})\gamma^2 + (1-\gamma)\right) \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})].\end{aligned}$$

Therefore, $\mathbb{E}_{t\sim F(t)}[u_1(b,t)] \leq \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})]$ for any $b \in (0,(1-e^{-1/\gamma})v]$ if

$$(1-e^{-1/\gamma})\gamma^2 + (1-\gamma) \leq 1 \quad \iff \quad \gamma(1-e^{-1/\gamma}) \leq 1,$$

which holds for any $\gamma \in (0,1]$ as required. This shows that bidder 1 does not have any profitable deviation in the interval $(0,(1-e^{-1/\gamma})v]$. Finally, since $b = (1-e^{-1/\gamma})v$ already gives $F(b) = 1$, any higher bid will only lead to a (strictly) higher payment (since $\gamma > 0$), thereby being (strictly) worse than bidding $b = (1-e^{-1/\gamma})v$. Hence, deviations to a bid higher than this upper bound of the support of $F(t)$ need not be considered.

Concluding, $\boldsymbol{\sigma}$ is a CCE for which the ratio of the social welfare of the social optimum and the expected social welfare of $\boldsymbol{\sigma}$ exactly coincides with the upper bound derived in the previous section. □

6.5 No Overbidding

6.5.1 Multi-Unit Auction

In the previous section, we have completely settled the coarse correlated price of anarchy of γ-FPA when overbidding is allowed. We see that especially when γ gets small this has an extremely negative effect on the price of anarchy. In this section, we will investigate how these bounds improve under the no-overbidding assumption (NOB as defined above). It is a standard assumption to make and we will see that it leads to a significant improvement of the price of anarchy bounds, most notably for lower values of γ.

We can show that pure Nash equilibria of γ-HYA without overbidding are always efficient for all $\gamma \in (0,1]$. For coarse correlated equilibria, we can significantly improve the upper bound derived in Theorem 6.4.1 for $\gamma \lessapprox 0.607$.

Pure Price of Anarchy

The pure price of anarchy of γ-HYA without overbidding has been analyzed before for $\gamma = 0$ and $\gamma = 1$: Birmpas et al. (2019) show that the PNE-PoA is 2.1885 for the second-price multi-unit auction ($\gamma = 0$), while de Keijzer et al. (2013) show

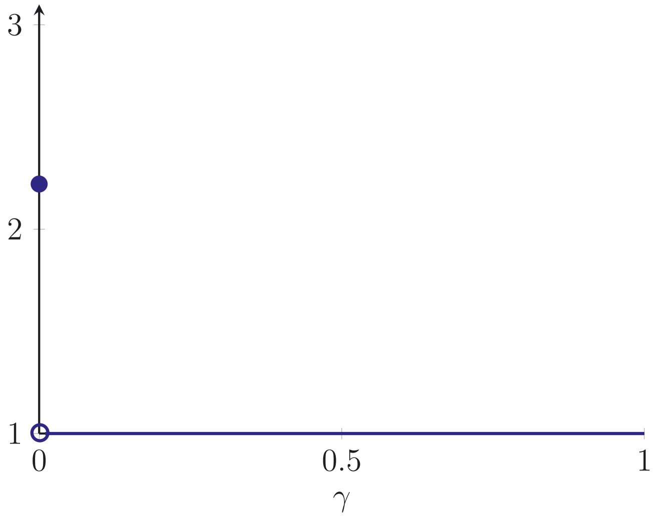


Figure 6.3: PNE-PoA of γ-HYA without overbidding (Theorem 6.5.1)

that the PNE-PoA is 1 for the first-price multi-unit auction ($\gamma = 1$). As it turns out, for γ-HYA the PNE-PoA stays at 1 almost over the entire range, the only exception being at $\gamma = 0$ where it is 2.1885 for γ-HYA by the result of Birmpas et al. (2019) (see Figure 6.3).

Theorem 6.5.1. *Pure Nash equilibria of* γ-HYA *without overbidding are always efficient, i.e.,* PNE-PoA $= 1$*, for all* $\gamma \in (0, 1)$*.*

This theorem follows from a minor adaption of a result by de Keijzer et al. (2013), who show that pure Nash equilibria are always efficient for $\gamma = 1$. For completeness, we also write down the proof for our setting. We start with the following lemma.

Lemma 6.5.2. *Let* $\boldsymbol{b}$ *be a pure Nash equilibrium in a given* γ-HYA*with* $\gamma \in (0, 1)$ *and with* k *items. Let* d *be the highest losing bid, i.e.,* $d = \max\{b_i(j) \mid i \in N, j \in [k], j > x_i(\boldsymbol{b})\}$*. Then, for every bidder* $i \in N$*:*

(i) if $x_i(\boldsymbol{b}) > 0$ *it holds that* $b_i(j) = d$ *for all* $j \in [x_i(\boldsymbol{b})]$*,*

(ii) $\ell \cdot d \leq v_i(x_i(\boldsymbol{b})) - v_i(x_i(\boldsymbol{b} - \ell)$ *for all* $\ell \in [x_i(\boldsymbol{b})]$*,*

(iii) $v_i(x_i(\boldsymbol{b}) + \ell) - v_i(x_i(\boldsymbol{b})) \leq \ell \cdot d$ *for all* $\ell \in [k - x_i(\boldsymbol{b})]$*.*

Proof:

(i) Let c be the highest winning bid, i.e., $c = \max\{b_i(j) \mid i \in N, j \leq x_i(\boldsymbol{b})\}$. Clearly, $c \geq d$. Suppose $c > d$. Let i be a winning player that wins an item by bidding c. When changing their bid to $\frac{c+d}{2}$ they are still a winner and because $\gamma \in (0,1)$ they will pay less, improving their utility. This contradicts $\boldsymbol{b}$ being a pure Nash equilibrium, hence $c = d$. If the highest winning bid is equal to the highest lowest bid then all winning bids must be equal to d.

(ii) Suppose that for some i and ℓ it holds that

$$\ell \cdot d > v_i(x_i(\boldsymbol{b})) - v_i(x_i(\boldsymbol{b} - \ell).$$

This means that player i gets a utility of $v_i(x_i(\boldsymbol{b})) - v_i(x_i(\boldsymbol{b} - \ell))$ for the last ℓ won items while paying $\ell \cdot d$ for them. If player i now changes their bid to $b_i(j) = 0$ for $j \geq x_i(\boldsymbol{b}) - \ell + 1$ their utility goes up by $\ell \cdot d - (v_i(x_i(\boldsymbol{b})) - v_i(x_i(\boldsymbol{b} - \ell)) > 0$ which contradicts the pure Nash equilibrium conditions.

(iii) Suppose that for some i and ℓ it holds that

$$v_i(x_i(\boldsymbol{b}) + \ell) - v_i(x_i(\boldsymbol{b}) > \ell \cdot d. \tag{6.8}$$

In this case if player i changes $b_i(j)$ to $d + \epsilon$ for $1 \leq j \leq x_i(\boldsymbol{b}) + \ell$. For the items they already won they will pay $\gamma \cdot \epsilon \cdot x_i(\boldsymbol{b})$ more and for the new items they will pay $\ell \cdot (\gamma \cdot (d + \epsilon) + (1 - \gamma) \cdot d) = \ell \cdot d + \ell \cdot \gamma \cdot \epsilon$ while gaining $v_i(x_i(\boldsymbol{b}) + \ell) - v_i(x_i(\boldsymbol{b}))$ in utility. This totals to

$$v_i(x_i(\boldsymbol{b}) + \ell) - v_i(x_i(\boldsymbol{b}) - \ell \cdot d - \epsilon \cdot \gamma \cdot (x_i(\boldsymbol{b}) + \ell).$$

Since (6.8) is strict, we can choose ϵ small enough so that this gets positive contradicting the pure Nash equilibrium conditions.

□

This lemma is the main ingredient for proving Theorem 6.5.1.

Proof of Theorem 6.5.1: Let $\boldsymbol{b}^*$ be a bid vector obtaining optimal social welfare and let $\boldsymbol{b}$ be a pure Nash equilibrium. Let d be as in Lemma 6.5.2.

$$\begin{aligned}
&\sum_{i \in N} v_i(x_i(\boldsymbol{b})) - \sum_{i \in N} v_i(x_i^*(\boldsymbol{b})) \\
&= \sum_{i \in N : x_i(\boldsymbol{b}) > x_i(\boldsymbol{b}^*)} v_i(x_i(\boldsymbol{b})) - v_i(x_i(\boldsymbol{b}^*)) - \sum_{i \in N : x_i(\boldsymbol{b}) < x_i(\boldsymbol{b}^*)} v_i(x_i(\boldsymbol{b}^*)) - v_i(x_i(\boldsymbol{b})) \\
&\geq \sum_{i \in N : x_i(\boldsymbol{b}) > x_i(\boldsymbol{b}^*)} (x_i(\boldsymbol{b}) - x_i(\boldsymbol{b}^*)) \cdot d - \sum_{i \in N : x_i(\boldsymbol{b}) < x_i(\boldsymbol{b}^*)} (x_i(\boldsymbol{b}^*) - x_i(\boldsymbol{b})) \cdot d \\
&= 0,
\end{aligned}$$

where the inequality follows from points (ii) and (iii) in Lemma 6.5.2 and the last equality from the fact that $\sum_{i \in N} x_i(\boldsymbol{b}) = \sum_{i \in N} x_i(\boldsymbol{b}^*)$.

□

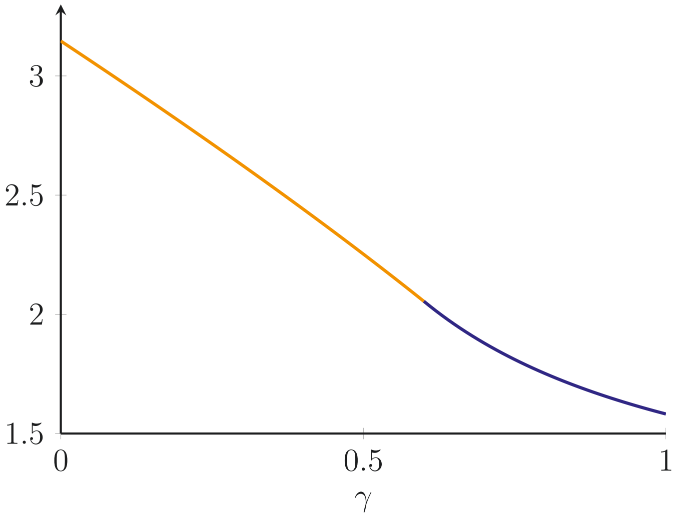


Figure 6.4: CCE-PoA for multi-unit γ-FPA without overbidding (Theorems 6.4.1 & 6.5.3).

Coarse Correlated Price of Anarchy

Theorem 6.5.3. *Consider a multi-unit* γ-FPA *and suppose that bidders cannot overbid. For* $\gamma \lessapprox 0.607$*, the coarse correlated price of anarchy is*

$$\text{CCE-PoA} \leq -(1-\gamma)\mathcal{W}_{-1}\left(-\frac{1}{e^{(2-\gamma)/(1-\gamma)}}\right). \tag{6.9}$$

Combining the improved bound of Theorem 6.5.3 with the bound of Theorem 6.4.1 yields the upper bound displayed in Figures 6.1(b) and 6.4 for all $\gamma \in [0,1]$. In particular, we obtain CCE-PoA $\leq -\mathcal{W}_{-1}(-e^{-2}) \approx 3.146$ for $\gamma = 0$ and CCE-PoA $\leq e/(e-1) \approx 1.582$ for $\gamma = 1$.

Proof: Similar to the proof of Theorem 6.4.1, we choose some $\alpha > 0$ to optimize the upper bound on the price of anarchy in Theorem 6.3.6 for any given $\gamma \in [0,1]$. As argued in the proof of Theorem 6.4.1, it is optimal to use $\alpha = \gamma$ when restricting to $\alpha \leq \gamma$. Using the no-overbidding assumption, we can also set $\alpha \geq \gamma$ and obtain

$$\text{CCE-PoA} \leq \frac{1+\alpha-\gamma}{\alpha\left(1-e^{-1/\alpha}\right)}. \tag{6.10}$$

This upper bound is minimized for

$$\alpha = -\frac{1}{\mathcal{W}_{-1}\left(-e^{-(2-\gamma)/(1-\gamma)}\right)+\frac{2-\gamma}{1-\gamma}}, \tag{6.11}$$

where $\mathcal{W}_{-1}$ is the lower branch of the Lambert $\mathcal{W}$ function. Substituting this into (6.10), we obtain the upper bound in (6.9). Importantly, the optimized bound in (6.9) is only valid if we have $\alpha \geq \gamma$, which does not hold for the entire range $\gamma \in [0, 1]$ if we use (6.11). More concretely, we have $\alpha \geq \gamma$ for all $\gamma \leq 0.607...$ only. Thus, for $\gamma \leq 0.607...$ we can use (6.9) to bound the price of anarchy. For $\gamma \geq 0.607...$ the best we can do is to choose $\alpha = \gamma$ and obtain the same CCE-PoA bound as in Theorem 6.4.1. □

6.5.2 Single-Item γ-HYA

We can further improve the price of anarchy bounds for single-item γ-HYA. It allows us to make more direct use of the payments giving us more control. We start with the general n-player setting, for which we show that the single-item γ-HYA is fully efficient up to correlated equilibria. For coarse correlated equilibria, we then derive a strong bound for low values of γ, namely CCE-PoA $\leq 1/(1-\gamma)$. This bound can in turn be complemented by the bound we derived for multi-unit auctions. Finally, to improve upon this multi-unit bound for the higher range of γ, we derive two technically more involved bounds that work specifically in a two-player setting.

We need some more notation. Given a bid vector $\boldsymbol{b}$, let $HB(\boldsymbol{b}) = \max_i \boldsymbol{b}_i$ and $SB(\boldsymbol{b})$ denote the highest and second-highest bid in $\boldsymbol{b}$, respectively, and let $HB_{-i}(\boldsymbol{b}) = \max_{j \neq i} \boldsymbol{b}_j$ be the highest bid excluding bid $\boldsymbol{b}_i$. For a randomized bid vector $\boldsymbol{\sigma}$, let $HB(\boldsymbol{\sigma})$ be the random variable equal to the highest bid when the bids are distributed according to $\boldsymbol{\sigma}$. We sometimes write $\mathbb{E}[HB(\boldsymbol{\sigma})]$ for $\mathbb{E}_{\boldsymbol{b} \sim \boldsymbol{\sigma}}[HB(\boldsymbol{b})]$ (similarly for $SB(\boldsymbol{\sigma})$ and $HB_{-i}(\boldsymbol{\sigma})$).

Correlated Price of Anarchy

We prove that γ-HYA is fully efficient for all $\gamma \in (0, 1]$ up to correlated equilibria. We extend a result in (Feldman, Lucier, and Nisan, 2016), which only does it for $\gamma = 1$. Below we show that for $\gamma = 0$ even coarse correlated equilibria are always efficient so that Theorem 6.5.4 in fact holds for all $\gamma \in [0, 1]$.

Theorem 6.5.4. *Consider a single-item* γ-HYA *and suppose that bidders cannot overbid. Then, the correlated price of anarchy of* γ-HYA *is* 1 *for all* $\gamma \in (0, 1]$.

Proof: Without loss of generality assume that player 1 has the highest valuation v_1. Assume towards contradiction that the CE-PoA is not 1. Then, there must be a player i for which $v_i < v_1$ who has a positive probability of winning. Let $b^* = \inf\{b \mid \mathbb{P}(HB(\sigma) < b) > 0\}$. Since we assume that players cannot overbid, we know that $b^* \leq v_i < v_1$.

First, suppose $b^* = v_i$. Then $\mathbb{P}(HB(\sigma) < v_i) = 0$. If player 1 bids $(v_i + v_1)/2$ if we draw $\boldsymbol{b} \sim \sigma$ for which $b_1 < b^*$ (in case ties are broken in favor of player 1) or if $b_1 \leq b^*$ (in case ties are broken in favor of player i) then player 1 strictly increases their utility. This contradicts the correlated equilibrium assumption.

Thus we can assume $b^* < v_i$. Define $\tilde{b} = (b^* + v_i)/2$. Fix a bid b such that $b^* < b < \tilde{b}$. By assumption we have $\mathbb{P}(HB(\sigma) < b) > 0$. Either player 1 or player i must win by bidding not higher than b with probability at most $\mathbb{P}(HB(\sigma) < b)/2$. Let it be player i (otherwise just fill in 1 for i in what follows).

Consider the following deviating strategy for player i: bid $(\tilde{b} + b)/2$ if we draw $\boldsymbol{b} \sim \sigma$ for which $b_i \leq b$ and b_i otherwise. For $b_i > b$ nothing changes and so the utility stays the same. Next, consider $b_i \leq b$. Player i already won with probability at most $\mathbb{P}(HB(\sigma) < b)/2$. Now that they bid higher note that the second bid part of the price does not change while the highest bid part goes up by at most $\gamma((\tilde{b}+b)/2 - b^*)$ On the other hand player i will gain (lower bounding the second-highest bid by the highest bid) at least $\mathbb{P}(HB(\sigma) < b)/2 \cdot (v_i - (\tilde{b} + b)/2)$. Net, the utility of player j increases by at least

$$\frac{\mathbb{P}(HB(\sigma) < b)}{2} \left(\left(v_i - \frac{\tilde{b} + b}{2} \right) - \left(\frac{\tilde{b} + b}{2} - b^* \right) \right) >$$
$$\frac{\mathbb{P}(HB(\sigma) < b)}{2} \left(v_i + b^* - 2\tilde{b} \right) = 0,$$

where we use that $b < \tilde{b}$. Again, we find a contradiction with the CE conditions. Hence, there cannot be a player i with $v_i < v_1$ having a positive probability of winning implying that the price of anarchy must be 1. □

Coarse Correlated Price of Anarchy

It is known that the coarse correlated price of anarchy for the first-price auction is approximately 1.229 (Feldman, Lucier, and Nisan, 2016), which implies that the result of Theorem 6.5.4 does not extend to coarse correlated equilibria. We derive the following bound which is good for small values of γ.

Theorem 6.5.5. *Consider a single-item* γ-HYA *and suppose that bidders cannot overbid. Then, the coarse correlated price of anarchy of* γ-HYA *is at most* $1/(1-\gamma)$ *for all* $\gamma \in [0, 1)$.

Proof: Let player 1 be the player with the highest valuation v_1, and if there are multiple players with the highest valuation the player for whom ties are broken in favor when bidding v_1. Let $\boldsymbol{\sigma}$ be a coarse correlated equilibrium. We have

$$\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[\mathrm{SW}(\boldsymbol{b})] = \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})] + \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[\sum_{i\neq 1} u_i(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b}) \right]. \tag{6.12}$$

Define $\mathcal{E}$ as the event that player 1 wins the auction with respect to $\boldsymbol{\sigma}$, and let $\bar{\mathcal{E}}$ be the complement event that player 1 does not win the auction with respect to $\boldsymbol{\sigma}$.

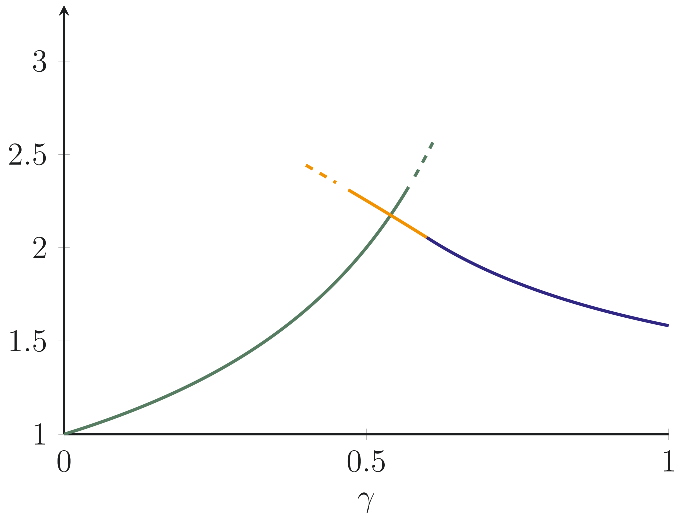


Figure 6.5: CCE-PoA for single-item γ-HYA without overbidding (Theorems 6.4.1, 6.5.3 & 6.5.5).

Suppose player 1 deviates to v_1. Then player 1 wins under $(v_1, \boldsymbol{b}_{-1})$ because either they are the single highest bid or ties are broken in their favor by assumption and no player overbids; note that this holds independently for $\mathcal{E}$ and $\bar{\mathcal{E}}$. By the CCE conditions, we thus have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(\boldsymbol{b})] &\geq \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[u_1(v_1, \boldsymbol{b}_{-1})] \\ &= (1-\gamma)v_1 - (1-\gamma)\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[HB_{-1}(\boldsymbol{b})\right].\end{aligned}$$

Substituting this inequality in (6.12), we obtain

$$\begin{aligned}\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}[\mathrm{SW}(\boldsymbol{b})] \geq (1-\gamma)v_1 &- (1-\gamma)\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[HB_{-1}(\boldsymbol{b})\right] \\ &+ \mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[\sum_{i\neq 1} u_i(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b})\right].\end{aligned}$$

The proof thus follows if we can show that

$$\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[\sum_{i\neq 1} u_i(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b})\right] \geq (1-\gamma)\mathbb{E}_{\boldsymbol{b}\sim\boldsymbol{\sigma}}\left[HB_{-1}(\boldsymbol{b})\right]. \tag{6.13}$$

Case 1: Suppose $\boldsymbol{b} \in \mathcal{E}$. Then player 1 wins the auction with respect to $\boldsymbol{b}$ and we have

$$\sum_{i\neq 1} u_i(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b}) = \gamma b_1 + (1-\gamma)HB_{-1}(\boldsymbol{b}) \geq HB_{-1}(\boldsymbol{b}).$$

Case 2: Suppose $\boldsymbol{b} \in \bar{\mathcal{E}}$. Then some other player $i' \neq 1$ wins the auction with respect to $\boldsymbol{b}$ and we have

$$u_{i'}(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b}) = v_{i'} - p^{\gamma}(\boldsymbol{b}) + p^{\gamma}(\boldsymbol{b}) = v_{i'} \geq b_{i'} = HB_{-1}(\boldsymbol{b}),$$

where last inequality holds because i' does not overbid and the last equality holds because i' being the highest bidder implies that $b_{i'} = HB_{-1}(\boldsymbol{b})$. This concludes the proof. □

Any upper bound for the multi-unit auction setting of course also holds for the single-item setting. By combining the bounds of Theorem 6.4.1, Theorem 6.5.3, and Theorem 6.5.5, we obtain the upper bound displayed in Figures 6.1(c) and 6.5 for the coarse correlated price of anarchy in the single-item auction setting.

Coarse Correlated Price of Anarchy for 2-player Auctions

We now present a more fine-grained picture of the coarse correlated price of anarchy for the 2-player setting. Ultimately, the upper bound for CCE-PoA for two players becomes a combination of three upper bounds, as represented by the three colors in Figures 6.1(d) and 6.6. We already derived the bound we use for small values of γ in Theorem 6.5.5, corresponding to the green graph in the figure. To derive the two remaining bounds, we use an approach inspired by Feldman, Lucier, and Nisan (2016). The extra difficulty we have is bounding the second-price component. The first-price component has a direct relation with winning the auction and so we can use the CCE conditions to bound it while the second-price component is more difficult to get a grip on. These bounds significantly improve on the bounds of Theorem 6.4.1 and Theorem 6.5.3.

First, we tackle the interval $\gamma \in [\frac{1}{2}, 1]$. Note that for $\gamma = 1$ this bound coincides with the (tight) bound in (Feldman, Lucier, and Nisan, 2016).

Theorem 6.5.6. *Consider a 2-player single-item* γ-HYA *and suppose that bidders cannot overbid. For* $\gamma \in [\frac{1}{2}, 1]$*, the coarse correlated price of anarchy of* γ-HYA *is upper bounded by the blue graph in Figures 6.1(d) and 6.6 (with* CCE-PoA $\leq 1.295...$ *for* $\gamma = 0.5$ *and* CCE-PoA $\leq 1.229...$ *for* $\gamma = 1$*).*

Proof: Without loss of generality we assume that player 1 has a valuation of 1 and player 2 has a valuation of $v \leq 1$. Fix γ and consider some coarse correlated equilibrium $\boldsymbol{\sigma}$. Let $\alpha = \mathbb{E}[u_1(\boldsymbol{\sigma})]$ be the utility of player 1 and $\beta = \mathbb{E}[u_2(\boldsymbol{\sigma})]$ be the utility of player 2 in $\boldsymbol{\sigma}$. The maximum social welfare is clearly 1, namely when player 1 wins all the time. Lower bounding the expected welfare of an arbitrary $\boldsymbol{\sigma}$ translates into an upper bound on the price of anarchy. We have

$$\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] \geq \alpha + \beta + \mathbb{E}[p^{\gamma}(\boldsymbol{\sigma})] = \alpha + \beta + \gamma\mathbb{E}[HB(\boldsymbol{\sigma})] + (1-\gamma)\mathbb{E}[SB(\boldsymbol{\sigma})].$$

We try to find the v, α, and β that minimize this expression and this will then give a lower bound on the expected social welfare. Let F_X be the cumulative distribution function of the random variable X where $X \in \{HB, HB_{-1}, HB_{-2}, SB\}$. Then by the CCE conditions, and the fact that a CDF is always bounded by 1, we know that

$$F_{HB_{-1}(\sigma)}(x) \le \min\left\{\frac{\alpha}{1-x}, 1\right\},\ F_{HB_{-2}(\sigma)}(x) \le \min\left\{\frac{\beta}{v-x}, 1\right\}, \tag{6.14}$$

$$F_{HB(\sigma)}(x) \le \min\left\{\frac{\alpha}{1-x}, \frac{\beta}{v-x}, 1\right\}. \tag{6.15}$$

For example, if $F_{HB_{-1}(\boldsymbol{\sigma})} > \frac{\alpha}{1-x}$ and player 1 changes their bid to x their utility will be strictly greater than $\frac{\alpha}{1-x} \cdot (1-x) = \alpha$ which is more than their current utility contradicting the CCE conditions.

Also, note that $\alpha \ge 1 - v$ because player 1 bidding $v + \epsilon$ will yield a utility of at least $1 - v - \epsilon$ for any positive ϵ. The other player is not allowed to bid above v, thus player 1 always wins when bidding $v + \epsilon$.

Observe that for $n = 2$ players the following chain of equalities holds

$$\begin{aligned} F_{SB(\boldsymbol{\sigma})}(x) &= \mathbb{P}(SB(\boldsymbol{\sigma}) \le x) \\ &= \mathbb{P}(\min(HB_{-1}(\boldsymbol{\sigma}), HB_{-2}(\boldsymbol{\sigma})) \le x) \\ &= \mathbb{P}(HB_{-1}(\boldsymbol{\sigma}) \le x) + \mathbb{P}(HB_{-2](\boldsymbol{\sigma})} \le x) - \mathbb{P}(HB(\boldsymbol{\sigma}) \le x) \\ &= F_{HB_{-1}(\boldsymbol{\sigma})}(x) + F_{HB_{-2}(\boldsymbol{\sigma})}(x) - F_{HB(\boldsymbol{\sigma})}(x). \end{aligned} \tag{6.16}$$

Let us get a more explicit expression for the expected payment using (6.16)

$$\begin{aligned} \mathbb{E}[p^{\gamma}(\boldsymbol{\sigma})] &= \gamma \mathbb{E}[HB(\boldsymbol{\sigma})] + (1-\gamma)\mathbb{E}[SB(\boldsymbol{\sigma})] \\ &= \gamma \int_0^1 1 - F_{HB(\boldsymbol{\sigma})}(x)dx + (1-\gamma)\int_0^1 1 - F_{SB(\boldsymbol{\sigma})}(x)dx \qquad (6.17) \\ &= \gamma \int_0^1 1 - F_{HB(\boldsymbol{\sigma})}(x)dx + (1-\gamma)\cdot \\ &\quad \int_0^1 1 - F_{HB_{-1}(\boldsymbol{\sigma})}(x) - F_{HB_{-2}(\boldsymbol{\sigma})}(x) + F_{HB(\boldsymbol{\sigma})}(x)dx \\ &= (2\gamma - 1)\int_0^1 1 - F_{HB(\boldsymbol{\sigma})}(x)dx + \\ &\quad (1-\gamma)\sum_{i=1}^{2}\int_0^1 1 - F_{HB_{-i}(\boldsymbol{\sigma})}(x)dx. \end{aligned}$$

Using the two bounds in (6.14) we can lower bound the two integrals in the summation

$$\int_0^1 1 - F_{HB_{-1}(\boldsymbol{\sigma})}(x)dx \ge \int_0^{1-\alpha} 1 - \frac{\alpha}{1-x}dx = 1 - \alpha + \alpha\ln(\alpha),$$

$$\int_0^1 1 - F_{HB_{-2}(\boldsymbol{\sigma})}(x)dx \ge \int_0^{v-\beta} 1 - \frac{\beta}{v-x}dx = v - \beta + \beta\ln(\beta/v).$$

If $\gamma \geq \frac{1}{2}$ then $2\gamma - 1 \geq 0$ and so we can use (6.15) to lower bound the integral on the left by

$$\int_0^1 1 - F_{HB(\boldsymbol{\sigma})}(x)dx \geq \int_0^1 1 - \min\left\{\frac{\alpha}{1-x}, \frac{\beta}{v-x}, 1\right\} dx.$$

We split up in two cases.

<u>Case 1:</u> $\beta \geq v\alpha$. Then $\frac{\beta}{v-x} \geq \frac{\alpha}{1-x}$ for all $x \in [0, v]$ and so

$$\int_0^1 1 - \min\left\{\frac{\alpha}{1-x}, \frac{\beta}{v-x}, 1\right\} dx = \int_0^{1-\alpha} 1 - \frac{\alpha}{1-x} dx = 1 - \alpha + \alpha\ln(\alpha), \tag{6.18}$$

giving a lower bound on expected welfare of

$$\begin{aligned}\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] \geq \alpha &+ \beta + (2\gamma-1)(1-\alpha+\alpha\ln(\alpha)) \\ &+ (1-\gamma)\left(1-\alpha+\alpha\ln(\alpha)+v-\beta+\beta\ln(\beta/v)\right).\end{aligned}$$

Using that $v \geq 1-\alpha$ and $\beta \geq v\alpha \geq \alpha(1-\alpha)$ this is lower bounded by

$$1 + \alpha(1-\alpha)\ln(\alpha) + \gamma\alpha(1-\alpha+\alpha\ln(\alpha)). \tag{6.19}$$

<u>Case 2:</u> $\beta < v\alpha$. First $\frac{\beta}{v-x}$ is smaller than $\frac{\alpha}{1-x}$ until $x = \theta = \frac{\alpha v - \beta}{\alpha - \beta}$ when $\frac{\beta}{v-x}$ takes over. In this case the integral is bounded from below by

$$\begin{gathered}\int_0^1 1 - \min\left\{\frac{\alpha}{1-x}, \frac{\beta}{v-x}, 1\right\} dx \geq \\ \int_0^\theta 1 - \frac{\beta}{v-x}dx + \int_\theta^{1-\alpha} 1 - \frac{\alpha}{1-x}dx = \\ \alpha\ln\left(\frac{\alpha-\beta}{1-v}\right) + 1 - \alpha + \beta\ln\left(\frac{\beta(1-v)}{v(\alpha-\beta)}\right),\end{gathered}$$

which gives a lower bound on the expected social welfare of

$$\begin{aligned}\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] \geq \alpha &+ \beta \\ &+ (2\gamma-1)\left(\alpha\ln\left(\frac{\alpha-\beta}{1-v}\right) + 1 - \alpha + \beta\ln\left(\frac{\beta(1-v)}{v(\alpha-\beta)}\right)\right) \\ &+ (1-\gamma)\left(1-\alpha+\alpha\ln(\alpha)+v-\beta+\beta\ln(\beta/v)\right).\end{aligned} \tag{6.20}$$

The derivative with respect to v is

$$(2\gamma-1)\frac{\alpha v - \beta}{(1-v)v} + (1-\gamma)(1-\beta/v).$$

For $\beta < v\alpha$ this is positive and thus the minimum is attained when v is smallest, i.e., $v = 1-\alpha$. Substituting that in (6.20) gives

$$\begin{aligned}
\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq \alpha + \beta \\
&\quad + (2\gamma - 1)\left(\alpha \ln\left(\frac{\alpha - \beta}{\alpha}\right) + 1 - \alpha + \beta \ln\left(\frac{\beta\alpha}{(1-\alpha)(\alpha - \beta)}\right)\right) \\
&\quad + (1 - \gamma)\left(1 - \alpha + \alpha\ln(\alpha) + 1 - \alpha - \beta + \beta \ln\left(\frac{\beta}{1-\alpha}\right)\right) \\
&= 1 + \gamma\beta + (2\gamma - 1)(\alpha - \beta)\ln(\alpha - \beta) \\
&\quad + (2 - 3\gamma)\alpha\ln(\alpha) + (2\gamma - 1)\beta\ln(\alpha) \qquad (6.21) \\
&\quad + \gamma\beta\ln(\beta) - \gamma\beta\ln(1 - \alpha).
\end{aligned}$$

Note that filling in $\beta = \alpha(1-\alpha)$ yields the same revenue as in (6.19). Changing $\beta < v\alpha$ to $\beta \leq v\alpha = \alpha(1-\alpha)$ subsumes case 1. So we only have to find the minimum in case 2.

The derivative of (6.21) with respect to β is

$$(2\gamma - 1)\ln\left(\frac{\alpha}{\alpha - \beta}\right) + \gamma \ln\left(\frac{\beta}{1-\alpha}\right) + 1.$$

This becomes 0 when

$$\ln\left(\frac{\alpha^{2\gamma-1}\beta^{\gamma}}{(\alpha-\beta)^{2\gamma-1}(1-\alpha)^{\gamma}}\right) = -1 \iff \frac{\beta^{\gamma}}{(\alpha-\beta)^{2\gamma-1}} - \frac{(1-\alpha)^{\gamma}}{e\alpha^{2\gamma-1}} = 0.$$

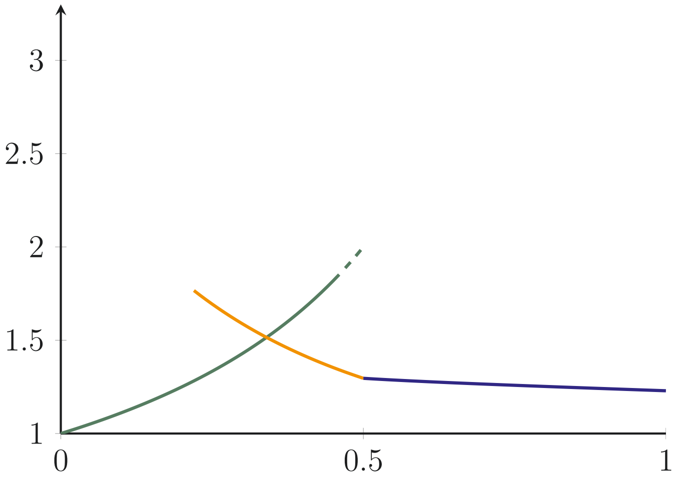
For fixed α the expression on the left is negative for β close to 0, and positive for β close to α. Also the second derivative with respect to β is always positive on $[0, \beta]$. Thus we can use binary search to quickly find β satisfying the equality. Call this β_α.[6]

Then we have

$$\begin{aligned}
\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq 1 + \gamma\beta_\alpha + (2\gamma - 1)(\alpha - \beta_\alpha)\ln(\alpha - \beta_\alpha) \\
&\quad + (2 - 3\gamma)\alpha\ln(\alpha) + (2\gamma - 1)\beta_\alpha\ln(\alpha) \\
&\quad + \gamma\beta_\alpha\ln(\beta_\alpha) - \gamma\beta_\alpha\ln(1 - \alpha).
\end{aligned}$$

For $\gamma = 1/2$ we compute $\beta_\alpha = (1-\alpha)/e^2$ and then the social welfare is minimized for $\alpha = e^{-1-1/e^2} \approx 0.3213...$ with value 0.7716.... While for $\gamma = 1$ we have $\beta_\alpha = \alpha(1-\alpha)/(1 - \alpha + e\alpha)$ where the social welfare is minimized for $\alpha \approx 0.2743...$ with value 0.8135.... In both cases (6.21) becomes a unimodal function. Plotting (6.21) for various values of α, when doing binary search to find β_α as a subroutine, suggests that this is the case for all γ. Making this assumption we can use a ternary search on α with a binary search to find β_α as a subroutine

[6] β_α may violate the case assumption that $\beta \leq v\alpha$ but removing this restriction can only decrease the minimum value of the expected social welfare.



(d)

Figure 6.6: CCE-PoA for single-item γ-HYA without overbidding and $n = 2$ bidders (Theorems 6.5.5, 6.5.6 & 6.5.7).

to quickly find the minimum. Finally, taking 1 over this value gives us an upper bound on the price of anarchy, presented as the blue graph in Figures 6.1(d) and 6.6. □

The previous theorem holds for $\gamma \in [\frac{1}{2}, 1]$. With a similar proof template, making use of an upper bound on the highest bid, we can derive an upper bound on the coarse correlated price of anarchy for the lower to mid-range of γ

Theorem 6.5.7. *Consider a 2-player single-item* γ-HYA *and suppose that bidders cannot overbid. For* $\gamma \in (0.217..., \frac{1}{2}]$*, the coarse correlated price of anarchy of* γ-HYA *is upper bounded by the orange graph in Figures 6.1(d) and 6.6 (with* CCE-PoA $\leq 1.515...$ *for intersection point* $\gamma = 0.339...$ *and* CCE-PoA $\leq 1.295...$ *for* $\gamma = 0.5$*).*

Proof: For $\gamma \in [0, 1/2]$, note that in the final equality of (6.17), we have $(2\gamma - 1) \leq 0$. To *lower* bound the social welfare, we should therefore *upper* bound the expected highest bid. For this, note that due to the fact that players cannot overbid, player 2 never bids higher than v. Therefore, for any $\gamma > 0$, any bid of player 1 that is (strictly) above v is (strictly) dominated by bidding v instead[7]. Using this, it is clear that $\mathbb{E}[HB(\boldsymbol{\sigma})] \leq v$. Again using (6.14) to lower bound the

[7] Formally, player 1 should bid $v + \epsilon$ for any $\epsilon > 0$. Since ϵ can be an arbitrarily small number, we ignore it in the remainder of the proof for notational convenience.

two rightmost integrals, we get

$$\begin{aligned}
&\gamma\mathbb{E}[HB(\boldsymbol{\sigma})] + (1-\gamma)\mathbb{E}[SB(\boldsymbol{\sigma})] \\
&\geq (2\gamma - 1)\int_0^1 1 - F_{HB(\boldsymbol{\sigma})}(x)dx \\
&\quad + (1-\gamma)\left(\int_0^1 1 - F_{HB_{-1}(\boldsymbol{\sigma})}(x)dx + \int_0^1 1 - F_{HB_{-2}(\boldsymbol{\sigma})}(x)dx\right) \\
&\geq (2\gamma-1)v + (1-\gamma)(1 - \alpha + \alpha\ln(\alpha) + v - \beta + \beta\ln(\beta/v)),
\end{aligned}$$

so that

$$\begin{aligned}
\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq \alpha + \beta + (2\gamma-1)v \qquad (6.22)\\
&\quad + (1-\gamma)(1 - \alpha + \alpha\ln(\alpha) + v - \beta + \beta\ln(\beta/v)) \\
&= \gamma(\alpha+\beta+v) + (1-\gamma)(1 + \alpha\ln(\alpha) + \beta\ln(\beta) - \beta\ln(v)).
\end{aligned}$$

The derivative of this bound with respect to β equals

$$\gamma + (1-\gamma)(1 + \ln(\beta) - \ln(v)) = 1 + (1-\gamma)\ln(\beta/v).$$

Note that this derivative is equal to zero for $\beta = ve^{-1/(1-\gamma)}$, and that it is positive for greater β and negative for smaller β. Therefore, the bound attains its minimum at $\beta = ve^{-1/(1-\gamma)}$. Substituting this β in (6.22) yields

$$\begin{aligned}
\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq \gamma(\alpha + (1 + e^{-\frac{1}{1-\gamma}})v) \qquad (6.23)\\
&\quad + (1-\gamma)(1 + \alpha\ln(\alpha) + ve^{-\frac{1}{1-\gamma}}\ln(e^{-\frac{1}{1-\gamma}}) \\
&= \gamma(\alpha + (1 + e^{-\frac{1}{1-\gamma}})v) + (1-\gamma)(1 + \alpha\ln(\alpha)) - ve^{-\frac{1}{1-\gamma}}.
\end{aligned}$$

Next, we take the derivative of (6.23) with respect to v, which gives

$$\gamma(1 + e^{-\frac{1}{1-\gamma}}) - e^{-\frac{1}{1-\gamma}} = \gamma - (1-\gamma)e^{-\frac{1}{1-\gamma}}.$$

This derivative is positive for all $\gamma > 0.21781\ldots$, so for all $\gamma \in (0.21781\ldots, 1/2]$, we minimize the upper bound by setting v to its lowest admissible value, being $v = 1 - \alpha$ (due to the CCE condition that $\alpha \geq 1 - v$).

Substituting the optimal parameter settings $\beta = ve^{-1/(1-\gamma)} = (1-\alpha)e^{-1/(1-\gamma)}$ gives the following social welfare bound

$$\begin{aligned}
\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq \gamma(\alpha + (1-\alpha)(1 + e^{-\frac{1}{1-\gamma}})) + (1-\gamma)(1 + \alpha\ln(\alpha)) - \\
&\quad (1-\alpha)e^{-\frac{1}{1-\gamma}} \\
&= 1 - (1-\gamma)(1-\alpha)e^{-\frac{1}{1-\gamma}} + (1-\gamma)\alpha\ln(\alpha), \qquad (6.24)
\end{aligned}$$

as a function of α only, which we optimize by setting its derivative with respect to α equal to zero. This yields

$$(1-\gamma)e^{-\frac{1}{1-\gamma}} + (1-\gamma)(1+\ln(\alpha)) = 0 \iff \ln(\alpha) = -1 - e^{-\frac{1}{1-\gamma}}$$
$$\iff \alpha = e^{-1-e^{-1/(1-\gamma)}}.$$

To facilitate the simplification of the formula of the final bound, we first substitute only $\ln(\alpha)$ in (6.24), after which α itself is substituted in the final step. We get

$$\begin{aligned}\mathbb{E}[\mathrm{SW}(\boldsymbol{\sigma})] &\geq 1-(1-\gamma)e^{-\frac{1}{1-\gamma}} + (1-\gamma)\alpha e^{-\frac{1}{1-\gamma}} + \\ &\quad (1-\gamma)\alpha(-1-e^{-\frac{1}{1-\gamma}}) \\ &= 1-(1-\gamma)e^{-\frac{1}{1-\gamma}} - (1-\gamma)\alpha \\ &= 1-(1-\gamma)(e^{-\frac{1}{1-\gamma}} + e^{-1-e^{-1/(1-\gamma)}}). \qquad (6.25)\end{aligned}$$

We divide 1 by (6.25) to get the upper bound on the price of anarchy presented as the orange graph in Figures 6.1(d) and 6.6. □

Extensions to n players:

We can modify the procedure in Theorems 6.5.6 and 6.5.7 to work for n players, producing a different graph for every n. Observe that a more general version of (6.16) holds:

$$\begin{aligned}F_{SB(\boldsymbol{\sigma})}(x) &= \sum_{i=1}^{n} F_{HB_{-i}(\boldsymbol{\sigma})}(x) - (n-1)F_{HB(\boldsymbol{\sigma})}(x) \\ &\leq \sum_{i=1}^{2} F_{HB_{-i}(\boldsymbol{\sigma})}(x) - (n-1)F_{HB(\boldsymbol{\sigma})}(x). \qquad (6.26)\end{aligned}$$

We can use this to bound the second-highest bid. Doing a similar analysis as in Theorem 6.5.6 gives us a procedure that works for $\gamma \in \left[\frac{n-1}{n}, 1\right]$ and similarly Theorem 6.5.7 yields a procedure for $\gamma \in \left(0.217..., \frac{n-1}{n}\right]$. We obtain bounds which are always tight at $\gamma = 1$ for any n, but the higher n the faster the bound goes up when γ gets away from 1, and already for $n = 4$ the multi-unit bound lies below what the extension of Theorem 6.5.7 can give us.

6.6 Conclusion

Our bound on the CCE-PoA of γ-FPA is tight over the entire range of $\gamma \in [0, 1]$ if players can overbid, both in the single-item and multi-unit auction setting. Although our bounds on the CCE-PoA are already rather low if players cannot overbid, further improvements might still be possible. For various equilibrium notions, there are open questions, in particular for γ-FPA. We have a tight bound for PNE-PoA and an upper bound for CCE-PoA, but where we have a bound of 1 on the CE-PoA for single-item γ-HYA we do not have anything for γ-FPA. For single-item γ-HYA up until CE-PoA is settled, but the CCE-PoA upper bound can probably be pushed down a bit. A starting point to get better bounds for single-item γ-HYA (with n players) is by getting a better grip on the CDF of the second-highest bid in a CCE. Where the bounds on the highest bid carry over from the 2-player setting to the n-player setting, they do not for the second-highest bid. The idea in (6.26) is too weak to yield any good results. We consider these challenging open problems for future work.

On a more conceptual level, in this chapter, we considered a basic bid-rigging model where the auctioneer colludes with the winning bidders only. It will be very interesting to study the price of anarchy of more complex bid-rigging models; for example, the model introduced by Lengwiler and Wolfstetter (2010) (ideally generalized to the multi-unit auction setting), where the corrupt auctioneer not only approaches winners and offers them to lower their bid, but also approaches losers with the offer to raise their bid, might be a natural next step.

Instead of going to more complicated bid-rigging settings, one can also change the type of auction that is analyzed. The big-rigging examples we gave in the introduction were both procurement auctions, but we analyzed normal auctions. A natural direction to take is to analyze corruption in procurement auctions.

Another direction for future work is to study corruption in a mechanism design context. We have not done that in this chapter because we were interested in the effect of corruption on a first-price auction, one that is widely used in practice. So, opting for price of anarchy analysis was the more logical choice. However, it might be possible to create mechanisms in which the incentives to employ a bid-rigging scheme are taken away.

Chapter 7

Greater Flexibility in Mechanism Design Through Altruism

7.1 Introduction

The *self-interest hypothesis* is an assumption that is made in most models in mathematical economics. It is the assumption that individuals make decisions driven by purely selfish motives. Mechanism design, in particular, relies heavily on the assumption that individuals act self-interestedly. On a high level, mechanism design is concerned with guiding decision-making in a group context. Without incentives, egoism can motivate an individual to over- or understate their actual preferences for the possible alternatives. By providing the right incentives individuals are nudged towards revealing their true preferences that in turn can be used to make the best decision for the group as a whole. Often, payments by or to the participants are used as incentives.

In the preliminaries, we have already seen a family of mechanisms in which participants are incentivized to reveal their true preference: the VCG mechanism. We recall the Vickrey auction in the following example.

Example 7.1.1. Consider a single-item auction in which n bidders with valuations $v_1, \dots, v_n$ participate. The set of alternatives A consists of n elements, each element assigning the item to another player. The social choice function $f : V \to A$ chooses the alternative in which the item is assigned to the highest-bidding player. Given that the players submit bids $\boldsymbol{b} = (b_1, \dots, b_n)$ the winner pays the second-highest bid, while the rest pays nothing. In Definition (2.5.7) (VCG mechanism) this corresponds to

$$h_i(\boldsymbol{b}_{-i}) = [\boldsymbol{b}_{-i}]_1 \quad \text{for all } i \in N,$$

where $[\boldsymbol{x}]_j$ is the j-th highest value in $\boldsymbol{x}$. And so the payment functions are

$$p_i(\boldsymbol{b}) = [\boldsymbol{b}_{-i}]_1 - \sum_{j \neq i} b_j(f(\boldsymbol{b})) \quad \text{for all } i \in N.$$

In an auction, the seller is usually interested in obtaining some revenue, so it makes sense to ask for substantial payments from the participants. As a result, there is money flowing from the participants to some third party. However, often decisions have to be made in a group that would like the money to stay among the participants. An example is a group of housemates that needs to decide who is allowed to use a shared car on a particular day. An example of a mechanism (due to Bailey (1997) and Cavallo (2006)) where less money is going to a third party is the following:

Example 7.1.2. We will take the same model as in Example 7.1.1 but change the payments a bit. Let

$$h_i(\boldsymbol{b}_{-i}) = [\boldsymbol{b}_{-i}]_1 - \frac{[\boldsymbol{b}_{-i}]_2}{n} \quad \text{for all } i \in N,$$

such that the payments are defined as

$$p_i(\boldsymbol{b}) = [\boldsymbol{b}_{-i}]_1 - \frac{[\boldsymbol{b}_{-i}]_2}{n} - \sum_{j \neq i} b_j(f(\boldsymbol{b})).$$

Without loss of generality, assume $b_1 \geq b_2 \geq b_3 \geq \ldots \geq b_n$. Then player 1 wins and pays $b_2 - \frac{b_3}{n}$. Player 2 receives $\frac{b_3}{n}$ and the remaining players receive $\frac{b_2}{n}$. The sum of payments in this example is only $2 \cdot \frac{b_2 - b_3}{n}$ compared to b_2 in Example 7.1.1.

Because it is a VCG mechanism it is truthful (Proposition 2.5.8). But, since there are now participants receiving money from the mechanism it does not satisfy NPT.

Certainly, the assumption that individuals or entities act self-interestedly applies in many economic settings, and because it simplifies analysis, it enables us to predict behavior in many economic situations. However, various empirical studies show that individuals do not act solely self-interestedly and that the self-interest hypothesis often does not apply (Andreoni and Miller, 2002; Charness and Rabin, 2002; Kahneman, 2011). The field of behavioral economics is specifically dedicated to studying how the decisions of individuals and institutions vary from what would be the 'rational' or 'optimal' choice.

If the preferences of individuals are fully aligned with the interests of the group, there is no need for a mechanism to guide the decision-making. Where being fully selfish is on one end of the spectrum, being fully altruistic is on the other end of the spectrum. Many people are somewhere in between: they care about themselves but are also partially altruistic towards others. Taking into account that some individuals are partially altruistic might change the incentives that the participants need to reveal their true preferences. Because the incentives come in the form of payments, we are interested in the question: What are the effects of (partial) altruism on the payments in mechanism design?

7.1.1 Our Contributions

The main contributions in this chapter are as follows:

1. We introduce a general utility model incorporating the other-regarding preferences of players. Our approach is to adapt the standard utility model by adding to each player's utility an extra term that represents their dispositions towards the other players.

2. By adding these other-regarding preferences, the utilities of the players become interdependent. As a consequence, the general class of VCG mechanisms cannot be straightforwardly applied to our setting. However, we are able to derive a characterization of truthful mechanisms in our new utility model with other-regarding preferences. The key to deriving our characterization is to exploit the specific form of the disposition functions.

3. Unfortunately, this characterization does not provide us with a "recipe" of how to obtain truthful mechanisms. We, therefore, establish a sufficient condition for truthfulness. We also derive sufficient and necessary conditions for when the resulting mechanisms satisfy the no positive transfer (NPT) and individual rationality (IR) properties. This also serves as a design template for our mechanisms.

4. We then address the question of how the payments can be redistributed among the players (while maintaining truthfulness) such that the overall payments are minimized. In general, we cannot expect that such redistribution mechanisms are strongly budget-balanced (i.e., the sum of the payments equals zero). We, therefore, use a relation of *individual dominance* between mechanisms, introduced by Guo, Markakis, et al. (2013), and provide a characterization of such redistribution mechanisms for our new utility model.

5. We then consider two specific models of altruism that are captured by our utility model with other-regarding preferences in combination with two natural social welfare objectives. We derive truthful mechanisms satisfying NPT and IR for all four settings. As it turns out, the altruistic dispositions of the players provide us with some additional flexibility in choosing the payments. A common property is that as the degree of altruism of a player increases, the designer needs to pay them less to have them reveal their private valuations.

6. We demonstrate the usefulness of our mechanisms by applying them to some fundamental problems in mechanism design: For the bilateral trade problem, we show that our truthful mechanism can be run without any subsidy (if the involved players are sufficiently altruistic), while this is impossible when using VCG payments. In case of the public-project problem, we show that our mechanism allows us to overcome some pathological deficiencies that are unavoidable in the standard setting. In fact, even a modest degree of altruism turns out to be sufficient to resolve the problem positively. Finally, we show that any mechanism that does not take altruism into account can be converted into a mechanism that does, and at the same time uses lower payments, where the gain is proportional to the altruism levels of the players.

Altogether, our results provide some evidence that altruism can only help in the mechanism design setting considered here. This is in contrast to some previous works (although in a purely strategic setting) showing that altruism may also have a negative impact on equilibrium outcomes (Buehler et al., 2011; Caragiannis, Kaklamanis, Kanellopoulos, Kyropoulou, and Papaioannou, 2010; Chen et al., 2014).

7.1.2 Related Work

There are different types of other-regarding preferences. In this chapter, we focus mainly on altruism. Other types are spite (in some sense the opposite of altruism), reciprocity (Kozlovskaya and Nicolò, 2019) and inequity aversion (Fehr and Schmidt, 1999). For more information on the different types of other-regarding preferences, we refer to Fehr and Fischbacher (2002).

There is a relatively small but steady interest in incorporating altruism into algorithmic game theory (see for instance (Apt and Schäfer, 2014; Buehler et al., 2011; Caragiannis, Kaklamanis, Kanellopoulos, Kyropoulou, and Papaioannou, 2010; Chen et al., 2014; De Marco and Morgan, 2011; Hoefer and Skopalik, 2013; Kerkmann and Rothe, 2021; Nguyen et al., 2016; Rahn and Schäfer, 2013; Rothe, 2021)). However, in mechanism design, there are only a few references we can point to.

Brandt and Weiß (2001) argue that in many situations, entities sacrifice a bit of profit to 'hurt' their competitors. They show that in the presence of spiteful bidders, the second-price auction is not truthful anymore and design equilibrium strategies for their model. Tang and Sandholm (2012) are concerned with finding mechanisms that maximize revenue in the presence of spiteful or altruistic bidders in an incomplete information setting. In their model, the other-regarding preferences have a specific shape, a constant times the utility of the other players, where the utility is a random variable depending on the distributions of the valuations. Interestingly, bidders have to pay even if the seller keeps the item and get subsidized when another player wins. Kucuksenel (2012) also models altruism according to a player-oriented model in a Bayesian setting. He characterizes a class of mechanisms that are *interim efficient*: they lead to utility-wise Pareto efficient outcomes. In, to our knowledge, unpublished work, Cavallo (2012) studies a regret-based model of altruism wherein participants are willing to give up to an α fraction of their personal utility for the good of the group. In an incomplete information and all-or-nothing setting, he proposes mechanisms that are strongly budget-balanced, i.e., the sum of payments is 0, when the players are 'mildly altruistic'.

A simple yet effective way to redistribute a large portion of the surplus on the payments can be done by the Bailey-Cavallo redistribution function (Bailey, 1997; Cavallo, 2006). Other ways to redistribute payments are studied by Guo and Conitzer (2009, 2010) and Moulin (2009). To be able to compare redistribution functions, we need an order. In the setting of non-deficit Groves mechanisms, Guo, Markakis, et al. (2013) define two partial orders and give characterizations of the maximal elements.

7.2 Preliminaries

We are given a finite set $N = \{1, \ldots, n\}$ of $n \geq 1$ players and a finite set A of alternatives to choose from. Each player $i \in N$ has a *private valuation function* $v_i : A \to \mathbb{R}$ which specifies their preferences over the set of alternatives A, independently of the other players' preferences. Note that the valuation function v_i is considered to be *private* information, i.e., only known to player i themselves. Given an alternative $a \in A$, we say that $v_i(a)$ is the *valuation* of player i for alternative a. We define V_i as the set of all possible valuation functions of player i. Unless stated otherwise, we assume that $V_i \subseteq \mathbb{R}^A$ is unrestricted and commonly known. Define $V = V_1 \times \cdots \times V_n$.

Suppose there is a central *designer* (e.g., principal, government) who wants to determine a socially desirable outcome, taking the preferences of the players into account. Each player $i \in N$ expresses their preferences over the available alternatives by reporting a valuation function $b_i \in V_i$ (not necessarily equal to their private valuation function v_i). The designer then utilizes a mechanism to decide on an outcome. A *(direct revelation) mechanism* $M = (f, \boldsymbol{p})$ is specified by a *social choice function* $f : V \to A$ and a vector of *payment functions* $\boldsymbol{p} = (p_1, \ldots, p_n)$ with $p_i : V \to \mathbb{R}$ for all $i \in N$. Given the reported valuation functions $\boldsymbol{b} = (b_1, \ldots, b_n)$, the mechanism determines an alternative $f(\boldsymbol{b})$ and for each player $i \in N$ a payment $p_i(\boldsymbol{b})$ to be made to the designer.

We assume that each player wants to maximize a given utility function. In the *standard utility model*, each player $i \in N$ has a quasi-linear utility function defined as $u_i^s(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b})$. The goal of the designer is to determine an alternative that maximizes a given *design objective* $D : V \times A \to \mathbb{R}$, i.e., $f(\boldsymbol{b}) \in \arg\max_{a \in A} D(\boldsymbol{b}, a)$. A commonly used design objective is to maximize the *social welfare*, i.e., the sum of the valuations of all players; formally, $D^{sw}(\boldsymbol{b}, a) = \sum_{i \in N} b_i(a)$. For any design objective considered in this paper, we assume that we can decompose $D(\boldsymbol{b}, a) = \sum_{i \in N} d_i(b_i, a)$ for functions $d_i : V_i \times A \rightarrow \mathbb{R}$. Then, we write $D_{-i}(\boldsymbol{b}, a) = D_{-i}(\boldsymbol{b}_{-i}, a) = \sum_{j \in N \setminus \{i\}} d_j(\boldsymbol{b}_j, a)$.

A mechanism $M = (f, \boldsymbol{p})$ is *truthful* if for every player $i \in N$, for any vector of reported valuations $\boldsymbol{b} \in V$, we have that $u_i(v_i, \boldsymbol{b}_{-i}) \geq u_i(b_i, \boldsymbol{b}_{-i})$. In other words, a truthful mechanism ensures that for each player i it is always at least as good to report their private valuation v_i than any other valuation, independent of what the other players report. Another desirable property of a mechanism is that it never makes payments to the players. A mechanism M satisfies the *no positive transfers (NPT)* property if for every player $i \in N$ and all $\boldsymbol{b} \in V$ we have that $p_i(\boldsymbol{b}) \geq 0$. Sometimes, we only require that the sum of payments is non-negative, i.e., $\sum_{i \in N} \boldsymbol{p}_i(\boldsymbol{b}) \geq 0$ for all $\boldsymbol{b} \in V$, then we call the mechanism *non-deficit*. Finally, every player should be guaranteed to receive a non-negative utility if they report their valuations truthfully. A mechanism M satisfies the *individual rationality (IR)* property if for every player $i \in N$ and for all reported valuations $\boldsymbol{b}_{-i} \in V_{-i}$ of the other players, $u_i(v_i, \boldsymbol{b}_{-i}) \geq 0$.

7.3 Modeling Other-Regarding Preferences

7.3.1 Utility Model with Other-Regarding Preferences

We propose a general utility model capturing that players may care about other players (both in the positive and negative sense).

Definition 7.3.1. Suppose we are given a function $g_i : \mathbb{R}^{n-1} \times \mathbb{R}^n \rightarrow \mathbb{R}$ for every player $i \in N$ modeling their *other-regarding preferences.* The utility $u_i^{g_i}$ of player $i \in N$ in the *utility model with other-regarding preferences* is then defined as

$$u_i^{g_i}(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) \ + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})).$$

Observe that the function g_i does not depend on the private valuations of the other players (which would be infeasible). This reflects the intuition that the other-regarding preferences of a player originate from *beliefs* about the experiences of others rather than from their true experiences.[1]

In the definition, the other-regarding preferences are allowed to depend on the payments. This makes it very general, but in proofs, this poses an extra challenge and so for some results below we make the following assumption.

[1] Intuitively, the function g_i of player i can be viewed as being dependent on the reported valuation functions $\boldsymbol{b}_{-i}$ of the other players and the payment functions $\boldsymbol{p}$. Formally, however, g_i only depends on the respective *values* of these functions under the outcome $(f(\boldsymbol{b}), \boldsymbol{p}(\boldsymbol{b}))$ determined by the mechanism $M = (f, \boldsymbol{p})$ when run on $\boldsymbol{b}$.

Assumption 7.3.2. *The other-regarding preferences g_i do not depend on the payments and therefore only depend on $\boldsymbol{b}_{-i}(f(\boldsymbol{b}))$ for all $i \in N$.*

We will see in Section 7.5 that there are natural models where the other-regarding preferences do not depend on the payments.

7.3.2 Characterization of Truthful Mechanisms

Theorem 7.3.3. *A mechanism $M = (f, \boldsymbol{p})$ is truthful in the utility model with other-regarding preferences if and only if it satisfies the following two conditions:*

1. *For every player $i \in N$ the difference between the other-regarding preferences g_i and the payment p_i only depends on the chosen alternative $f(\boldsymbol{b})$ and $\boldsymbol{b}_{-i}$ (but not on b_i itself), i.e., there is a function $\mu_i : A \times V_{-i} \to \mathbb{R}$ such that*

$$p_i(\boldsymbol{b}) - g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) = \mu_i(f(\boldsymbol{b}), \boldsymbol{b}_{-i}).$$

2. *The alternative chosen by M satisfies for every player $i \in N$ that*

$$f(b_i, \boldsymbol{b}_{-i}) \in \arg \max_{a \in A(\boldsymbol{b}_{-i})} (b_i(a) - \mu_i(a, \boldsymbol{b}_{-i})),$$

where $A(\boldsymbol{b}_{-i}) = \{f(b'_i, \boldsymbol{b}_{-i}) \, : \, b'_i \in V_i\}$ refers to the image of $f(\cdot, \boldsymbol{b}_{-i})$.

Proof: We first prove the if part. Consider a player $i \in N$ and fix $\boldsymbol{b}_{-i} \in V_{-i}$ arbitrarily. Define $\bar{a} = f(v_i, \boldsymbol{b}_{-i})$ and $a = f(b_i, \boldsymbol{b}_{-i})$ as the alternatives chosen by M when i reports their private valuation function v_i truthfully and when i reports an arbitrary valuation function b_i, respectively.

By the first condition of the statement, we have

$$u_i^{g_i}(v_i, \boldsymbol{b}_{-i}) = v_i(\bar{a}) - \mu_i(\bar{a}, \boldsymbol{b}_{-i}) \quad \text{and} \quad u_i^{g_i}(b_i, \boldsymbol{b}_{-i}) = v_i(a) - \mu_i(a, \boldsymbol{b}_{-i}). \tag{7.1}$$

By the second condition, the alternative $\bar{a}$ chosen by M for $(v_i, \boldsymbol{b}_{-i})$ satisfies

$$v_i(\bar{a}) - \mu_i(\bar{a}, \boldsymbol{b}_{-i}) \geq v_i(a) - \mu_i(a, \boldsymbol{b}_{-i}). \tag{7.2}$$

Combining (7.1) and (7.2) proves truthfulness.

Now we prove the only-if part of the first condition. Consider a player $i \in N$ and fix an arbitrary $\boldsymbol{b}_{-i} \in V_{-i}$. For notational convenience, we define $m_i(b_i, \boldsymbol{b}_{-i})$ as a shorthand for

$$m_i(b_i, \boldsymbol{b}_{-i}) = p_i(b_i, \boldsymbol{b}_{-i}) - g_i(\boldsymbol{b}_{-i}(f(b_i, \boldsymbol{b}_{-i})), \boldsymbol{p}(b_i, \boldsymbol{b}_{-i})),$$

for $b_i \in V_i$, The utility of player i can then be written as $u_i^{g_i}(b_i, \boldsymbol{b}_{-i}) = v_i(f(b_i, \boldsymbol{b}_{-i})) - m_i(b_i, \boldsymbol{b}_{-i})$. Suppose there are two valuation functions $b_i, b'_i \in V_i$

of player i such that $f(b_i, \boldsymbol{b}_{-i}) = f(b'_i, \boldsymbol{b}_{-i})$ and $m_i(b_i, \boldsymbol{b}_{-i}) < m_i(b'_i, \boldsymbol{b}_{-i})$. Then by identifying the private valuation function v_i of i with b'_i, we obtain

$$\begin{aligned} u_i^{g_i}(v_i, \boldsymbol{b}_{-i}) &= v_i(f(b'_i, \boldsymbol{b}_{-i})) - m_i(b'_i, \boldsymbol{b}_{-i}) \\ &< v_i(f(b_i, \boldsymbol{b}_{-i})) - m_i(b_i, \boldsymbol{b}_{-i}) = u_i^{g_i}(b_i, \boldsymbol{b}_{-i}), \end{aligned}$$

which contradicts the truthfulness of M. Thus $m_i(b_i, \boldsymbol{b}_{-i}) = m_i(b'_i, \boldsymbol{b}_{-i})$ whenever $f(b_i, \boldsymbol{b}_{-i}) = f(b'_i, \boldsymbol{b}_{-i})$. This proves the existence of a function μ_i only depending on $f(\boldsymbol{b})$ and $\boldsymbol{b}_{-i}$ as claimed.

Finally, we prove the only-if part of the second condition. Again, consider a player $i \in N$ and fix an arbitrary $\boldsymbol{b}_{-i} \in V_{-i}$. Suppose there is some $b_i \in V_i$ such that $f(b_i, \boldsymbol{b}_{-i})$ is not a maximizer of the expression. Let $a' \in A(\boldsymbol{b}_{-i})$ be such a maximizer, i.e.,

$$a' \in \arg \max_{a \in A(\boldsymbol{b}_{-i})} (b_i(a) - \mu_i(a, \boldsymbol{b}_{-i})).$$

By the definition of $A(\boldsymbol{b}_{-i})$, we have $a' = f(b'_i, \boldsymbol{b}_{-i})$ for some $b'_i \in V_i$. By identifying the private valuation function v_i of i with b_i and defining $\bar{a} = f(b_i, \boldsymbol{b}_{-i})$, we obtain

$$u_i^{g_i}(v_i, \boldsymbol{b}_{-i}) = b_i(\bar{a}) - \mu_i(\bar{a}, \boldsymbol{b}_{-i}) < b_i(a') - \mu_i(a', \boldsymbol{b}_{-i}) = u_i^{g_i}(b'_i, \boldsymbol{b}_{-i}),$$

which contradicts the truthfulness of M. □

It might be difficult to know what the other-regarding preferences look like for a specific player i. When making assumption 7.3.2 this is not very important. Even when we design a mechanism assuming different other-regarding preferences it is still truthful.

Corollary 7.3.4. *Let g_i, g'_i be other-regarding preferences for which Assumption 7.3.2 holds. Suppose we have a truthful mechanism with respect to g'_i. Then the mechanism is also truthful with respect to g_i.*

Proof: For a truthful mechanism with respect to g'_i, Theorem 7.3.3 implies that there exists a $\mu'_i(f(\boldsymbol{b}), \boldsymbol{b}_{-i})$ such that

$$p_i(\boldsymbol{b}) - g'_i(\boldsymbol{b}_{-i}) = \mu'_i(f(\boldsymbol{b}), \boldsymbol{b}_{-i}).$$

Let $\mu_i(f\boldsymbol{b}), \boldsymbol{b}_{-i}) = \mu'_i(f\boldsymbol{b}), \boldsymbol{b}_{-i}) - g_i(\boldsymbol{b}_{-i}) + g'_i(\boldsymbol{b}_{-i})$, then also

$$p_i(\boldsymbol{b}) - g_i(\boldsymbol{b}_{-i}) = \mu_i(f(\boldsymbol{b}), \boldsymbol{b}_{-i}),$$

showing that we still have a truthful mechanism. □

7.3.3 Design template

Theorem 7.3.3 gives a characterization of truthful mechanisms but does not provide us with a "recipe" of how to obtain such mechanisms for a given design objective D.

Theorem 7.3.5. *Fix a design objective D. A mechanism $M = (f, \boldsymbol{p})$ is truthful in the utility model with other-regarding preferences if the following two conditions are satisfied:*

1. $f(\boldsymbol{b}) \in \arg\max_{a \in A} D(\boldsymbol{b}, a)$.
2. *For every player $i \in N$ there exist functions $h_i, \gamma_i : V_{-i} \to \mathbb{R}$ such that*

$$p_i(\boldsymbol{b}) = h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})).$$

Proof: If $p_i(\boldsymbol{b}) = h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))$ then

$$p_i(\boldsymbol{b}) - g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) = h_i(\boldsymbol{b}_{-i}) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})).$$

Let

$$\mu_i(f(\boldsymbol{b}), \boldsymbol{b}_{-i}) = h_i(\boldsymbol{b}_{-i}) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})),$$

and we see that the conditions of Theorem 7.3.3 are satisfied. □

We are interested in knowing when a mechanism satisfies NPT and IR. For mechanisms following our recipe, we can quickly characterize what conditions the functions h_i have to satisfy.

Proposition 7.3.6. *Let $M = (f, \boldsymbol{p})$ be a mechanism as defined in Theorem 7.3.5. Then M satisfies NPT if and only if for every player $i \in N$ and any $\boldsymbol{b} \in V$,*

$$h_i(\boldsymbol{b}_{-i}) \geq \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})) - g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})).$$

Further, M satisfies IR if and only if for every player $i \in N$ and any $\boldsymbol{b} \in V$,

$$h_i(\boldsymbol{b}_{-i}) \leq \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(v_i, \boldsymbol{b}_{-i})) + v_i(f(v_i, \boldsymbol{b}_{-i})).$$

Proof: Immediate from the definitions of NPT and IR. □

If we insist both on NPT and IR the above proposition shows that in principle there is leeway in choosing h_i of about $v_i(f(v_i, \boldsymbol{b}_{-i})) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b}))$; however, recall that h_i may only depend on $\boldsymbol{b}_{-i}$ and we might thus be unable to exploit the full range. Further, Proposition 7.3.6 shows that if the valuation functions can be negative then we cannot guarantee both NPT and IR. In fact, the same holds if $g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b}))$ is allowed to be negative.

What effect does not knowing g_i exactly have on individual rationality? When making Assumption 7.3.2 this does not pose a problem, as long as we underestimate it.

Proposition 7.3.7. *Let g_i, g_i' be a other-regarding preferences for which Assumption 7.3.2 holds, such that $g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) \geq g_i'(\boldsymbol{b}_{-i}(f(\boldsymbol{b})))$ for all $\boldsymbol{b}$. Let $M = (f, \boldsymbol{p})$ be a mechanism as in Theorem 7.3.5 with respect to g_i' for player i that is individually rational. The mechanism with respect to g_i is also truthful and individually rational.*

Proof: We have

$$u_i(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})).$$

By assumption $v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) + g_i'(\boldsymbol{b}_{-i}(f(\boldsymbol{b})) \geq 0$. Observe that

$$u_i(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})) \geq v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) + g_i'(\boldsymbol{b}_{-i}(f(\boldsymbol{b})) \geq 0.$$

□

7.4 Minimizing Payments

As shown in the previous section, there is leeway in choosing h_i. There are situations in which extracting higher payments is preferred. Often, when a seller sells an item via an auction, they want to extract the highest payments possible. On the other hand, there are situations in which it is desirable to keep as much of the payments within the group of participants. For example, suppose siblings inherit a house from their parents, but only one of the siblings can live in the house. The parents wanted the house to go to the sibling desiring the house the most. They need a mechanism in which the siblings reveal their true preferences and the payments are used to buy the other siblings out. On top of that, they do not want to spend much on a third party. Especially in these situations, one can argue that higher altruism levels are more plausible as the players are more familiar with each other. In Section 7.6.3 we will use the results from this section to observe that incorporating altruism can significantly reduce the surplus of the payments of the mechanism.

Insisting on both IR and NPT, and wanting to use a mechanism that follows the recipe from Theorem 7.3.5, Proposition 7.3.6 restricts us to payment functions that satisfy

$$h_i(\boldsymbol{b}_{-i}) \geq \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})) - g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})).$$

In Theorem 7.3.5 the payments depend on the other-regarding preferences, which in turn can depend on the payments. To avoid issues with recursive definitions, we make Assumption 7.3.2 in this section.

Under Assumption 7.3.2, we can easily find the payment functions that minimize the sum of payments $\sum_i p_i(b) = \sum_i h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))$ while satisfying NPT:

$$h_i(\boldsymbol{b}_{-i}) = \sup_{b'_i} \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(b'_i, \boldsymbol{b}_{-i})) - g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))), \tag{7.3}$$

where we define $\boldsymbol{b}' = (b'_i, \boldsymbol{b}_{-i})$. If it is clear what b'_i is, we use this notation throughout this section.

The examples with the siblings inheriting a house and the housemates sharing a car show that there is no need to insist on NPT in general. In fact, in those situations, we would rather have the payments redistributed over the players so that as little money as possible is wasted on a third party. Unfortunately, in many settings it is impossible to have a *strongly budget-balanced* mechanism, i.e., the sum of payments is 0 (Mas-Colell, Whinston, Green, et al., 1995; Myerson and Satterthwaite, 1983). As we cannot aim for strong budget balance in general, we would like to minimize the amount we cannot redistribute.

Not insisting on NPT makes us much more flexible. However, we do not want to subsidize the mechanism, and thus we keep the requirement that the mechanism should be non-deficit, i.e., $\sum_i \boldsymbol{p}_i(\boldsymbol{b}) \geq 0$ for all $\boldsymbol{b}$.

Guo, Markakis, et al. (2013) characterized Groves mechanisms that are undominated in terms of the amount of money flowing from the mechanism to the auctioneer. We will extend this to the mechanisms with other-regarding preferences.

We say that a non-deficit mechanism with payment vector $\boldsymbol{p}$ *collectively dominates* a non-deficit mechanism with payment vector $\boldsymbol{p}'$ if for all $\boldsymbol{b} : \sum_i \boldsymbol{p}_i(\boldsymbol{b}) \leq \sum_i \boldsymbol{p}'_i(\boldsymbol{b})$ and there is at least one $\boldsymbol{b}$ for which this inequality is strict. Getting characterizations for payments that are collectively undominated is difficult. We can, however, relax this requirement a bit.

Definition 7.4.1. A non-deficit mechanism with payment vector $\boldsymbol{p}$ is said to *individually dominate* a non-deficit mechanism with payment vector $\boldsymbol{p}'$ if for all $\boldsymbol{b}$ and i

$$\boldsymbol{p}_i(\boldsymbol{b}) \leq \boldsymbol{p}'_i(\boldsymbol{b}),$$

and there is at least one $\boldsymbol{b}$ and i for which $\boldsymbol{p}_i(\boldsymbol{b}) < \boldsymbol{p}'_i(\boldsymbol{b})$.

Individual domination defines a partial order on mechanisms. The maximal elements in this order are interesting because they are the mechanisms in which no player can improve without making another player worse off.

We closely follow Guo, Markakis, et al. (2013) to characterize non-deficit and individually undominated mechanisms that follow the recipe of Theorem 7.3.5.

Lemma 7.4.2. *A mechanism $M = (f, \boldsymbol{p})$ satisying the recipe from Theorem 7.3.5 is non-deficit if and only if for all i and $\boldsymbol{b}_{-i}$*

$$h_i(\boldsymbol{b}_{-i}) \geq \sup_{b'_i} \sum_j (\gamma_j(\boldsymbol{b}'_{-j}) \cdot D_{-j}((\boldsymbol{b}'_{-j}, f(\boldsymbol{b}'))) - g_j(\boldsymbol{b}'_{-j}(f(\boldsymbol{b}')))) - \sum_{j \neq i} h_j(\boldsymbol{b}'_{-j}). \tag{7.4}$$

Proof: For a mechanism to be non-deficit we need $\sum_i \boldsymbol{p}_i(\boldsymbol{b}) \geq 0$ for any $\boldsymbol{b}$. Fix some $\boldsymbol{b}$ and an i. Suppose a mechanism satisfies equation 7.4 then we know that

$$\begin{aligned} h_i(\boldsymbol{b}_{-i}) &\geq \sup_{b'_i} \sum_j \big(\gamma_j(\boldsymbol{b}'_{-j}) \cdot D_{-j}((\boldsymbol{b}'_{-j}, f(\boldsymbol{b}'))) - g_j(\boldsymbol{b}'_{-j}(f(\boldsymbol{b}')))\big) - \sum_{j \neq i} h_j(\boldsymbol{b}'_{-j}) \\ &\geq \sum_j (\gamma_j(\boldsymbol{b}_{-j}) \cdot D_{-j}(\boldsymbol{b}_{-j}, f(\boldsymbol{b})) - g_j(\boldsymbol{b}_{-j}(f(\boldsymbol{b})))) - \sum_{j \neq i} h_j(\boldsymbol{b}_{-j}). \end{aligned}$$

Rewriting yields

$$\sum_j h_j(\boldsymbol{b}_{-j}) + g_j(\boldsymbol{b}_{-j}(f(\boldsymbol{b}))) - \gamma_j(\boldsymbol{b}_{-j}) \cdot D_{-j}(\boldsymbol{b}_{-j}, f(\boldsymbol{b})) \geq 0.$$

As the left hand side is exactly the sum of payments we have proved the if direction.

Suppose a mechanism is non-deficit, then $\sum_i \boldsymbol{p}_i(\boldsymbol{b}) \geq 0$ for all $\boldsymbol{b}$, thus also

$$\sum_j h_j(\boldsymbol{b}_{-j}) + g_j(\boldsymbol{b}_{-j}(f(\boldsymbol{b}))) - \gamma_j(\boldsymbol{b}_{-j}) \cdot D_{-j}(\boldsymbol{b}, f(\boldsymbol{b})) \geq 0.$$

Fix some arbitrary i, we can rewrite

$$h_i(\boldsymbol{b}_{-i}) \geq \sum_j \gamma_j(\boldsymbol{b}_{-j}) \cdot D_{-j}(\boldsymbol{b}, f(\boldsymbol{b})) - g_j(\boldsymbol{b}_{-j}(f(\boldsymbol{b}))) - \sum_{j \neq i} h_j(\boldsymbol{b}_{-j}),$$

and because this holds for any b_i, this also holds for the supremum. □

Theorem 7.4.3. *A mechanism $M = (f, \boldsymbol{p})$ following the recipe from Theorem 7.3.5 is individually undominated if and only if for all i and $\boldsymbol{b}_{-i}$*

$$h_i(\boldsymbol{b}_{-i}) = \sup_{b'_i} \sum_j \big(\gamma_j(\boldsymbol{b}'_{-j}) \cdot D_{-j}(\boldsymbol{b}'_{-j}(f(\boldsymbol{b}'))) - g_j(\boldsymbol{b}'_{-j}(f(\boldsymbol{b}')))\big) - \sum_{j \neq i} h_j(\boldsymbol{b}'_{-j}). \tag{7.5}$$

Proof: For both implications, we prove the contrapositive. Since an individually undominated mechanism must be non-deficit we know by the previous lemma that the equality must be greater or equal. Suppose it is a strict inequality for some $i, \tilde{\boldsymbol{b}}_{-i}$, i.e., there exists a $\delta > 0$ such that

$$\begin{aligned} h_i(\tilde{\boldsymbol{b}}_{-i}) - \Big(\sup_{b'_i} \sum_j \big(\gamma_j(\tilde{\boldsymbol{b}}'_{-j}) \cdot D_{-j}(\tilde{\boldsymbol{b}}'_{-j}, f(\tilde{\boldsymbol{b}}')) - g_j(\tilde{\boldsymbol{b}}'_{-j}(f(\tilde{\boldsymbol{b}}')))\big) \\ - \sum_{j \neq i} h_j(\tilde{\boldsymbol{b}}'_{-j})\Big) = \delta > 0. \end{aligned} \tag{7.6}$$

Define

$$h'_j(\boldsymbol{b}_{-j}) = \begin{cases} h_i(\tilde{\boldsymbol{b}}_{-i}) - \delta & \text{if } i = j \text{ and } \boldsymbol{b}_{-j} = \tilde{\boldsymbol{b}}_{-i} \\ h_j(\boldsymbol{b}_{-j}) & \text{otherwise,} \end{cases} \tag{7.7}$$

and let

$$\boldsymbol{p}'_i(\boldsymbol{b}) = h'_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})).$$

If we can verify that $\boldsymbol{p}'$ is non-deficit then it individually dominates $\boldsymbol{p}$. This is clear because we have shifted $h'_i(\tilde{\boldsymbol{b}})$ exactly by the amount of slack there was, namely δ, and the rest remained unchanged. As the mechanism with respect to h was non-deficit, it is also with respect to h'.

For the other implication. Suppose $\boldsymbol{p}$ is individually dominated by $\boldsymbol{p}'$, then there exist i and $\boldsymbol{b}$ such that $\boldsymbol{p}_i(\boldsymbol{b}) > \boldsymbol{p}'_i(\boldsymbol{b})$. In particular,

$$\begin{aligned}\boldsymbol{p}_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))\\ &> h'_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - \gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))\\ &= \boldsymbol{p}'_i(\boldsymbol{b}).\end{aligned}$$

As g_i and $\gamma_i(\boldsymbol{b}_{-i}) \cdot D_{-i}$ are the same on both sides of the inequality this actually implies $h_i(\boldsymbol{b}_{-i}) > h'_i(\boldsymbol{b}_{-i})$.

Write

$$\sigma = \sup_{b'_i} \sum_j \big(\gamma_j(\boldsymbol{b}_{-j}) \cdot D_{-j}((b'_i, \boldsymbol{b}_{-i}), f(b'_i, \boldsymbol{b}_{-i})) - g_j(\boldsymbol{b}_{-j}(f(b'_i, \boldsymbol{b}_{-i})))\big) - \sum_{j \neq i} h_j(\boldsymbol{b}'_{-j}).$$

But then

$$h'_i(\boldsymbol{b}_{-i}) - \sigma > h'_i(\boldsymbol{b}_{-i}) - \sigma \geq 0,$$

showing that (7.5) is not satisfied for $\boldsymbol{p}$.

□

7.5 A Case Study: Altruism

7.5.1 Two Altruism Models and Design Objectives

We consider two models of altruism that are instantiations of the utility model with other-regarding preferences. We assume that each player $i \in N$ is equipped with an *altruism level* $\alpha_i \in [0, 1]$ which interpolates between a ‘purely selfish’ ($\alpha_i = 0$) and a ‘fully altruistic’ ($\alpha_i = 1$) attitude.[2]

[2] Note that although our focus here is on altruism levels in the range $[0, 1]$, some results also hold for other cases such as spiteful players ($\alpha_i < 0$) or players that care about others more than about themselves ($\alpha_i > 1$), but we do not mention it explicitly.

Definition 7.5.1. Given an altruism level $\alpha_i \in [0,1]$ for every player $i \in N$, in the *welfare-oriented model* the utility $u_i^w : V \to \mathbb{R}$ of player $i \in N$ is defined as:

$$u_i^w(\mathbf{v}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) \ + \alpha_i \sum_{j \neq i} b_j(f(\boldsymbol{b})).$$

In the welfare-oriented model each player i receives a fraction of α_i of the reported valuations of all other players. Note that i fully cares about their own payment. Altruism here corresponds to a willingness to contribute to the creation of value in the form of valuations of alternatives.

Definition 7.5.2. Given an altruism level $\alpha_i \in [0,1]$ for every player $i \in N$, in the *omnistic model* the utility $u_i^o : V \to \mathbb{R}$ of a player $i \in N$ is given by:

$$u_i^o(\boldsymbol{b}) = v_i(f(\boldsymbol{b})) - p_i(\boldsymbol{b}) \ + \alpha_i \left(p_i(\boldsymbol{b}) + \sum_{j \neq i} b_j(f(\boldsymbol{b})) \right).$$

In the omnistic model each player i cares about every other player the same way as in the welfare-oriented model. The difference, however, is that player i perceives their payment p_i to the designer as being discounted by a fraction of $(1 - \alpha_i)$ (although they pay p_i eventually). Put differently, i enjoys a fraction of α_i of the payment p_i that the designer receives from them. This is also the reason why we call this the 'omnistic' model (omnes = all/everybody).

We derive mechanisms for these models with respect to the following two design objectives:

$$D^{sw}(\boldsymbol{b}, a) = \sum_{i \in N} b_i(a) \quad \text{and} \quad D^{ow}(\boldsymbol{b}, a) = \sum_{i \in N} \left(1 + \sum_{k \neq i} \alpha_k \right) b_i(a).$$

The design objective D^{sw} is the classical social welfare objective. In our context, it captures situations where the designer only cares about the sum of the individual valuations of the players, disregarding the positive perceptions that they receive from other players. Intuitively, here the utility functions serve merely as a means to model the positive attitudes of players towards others.

The design objective D^{ow} models situations in which the designer takes both the individual valuations of the players and their positive other-regarding preferences towards others into account. Note that this objective is equal to the sum of all valuations that the players receive (directly or indirectly). We refer to it as the *omnistic welfare* objective.

7.5.2 Mechanisms for Altruistic Players

We derive truthful mechanisms for the welfare-oriented and omnistic model (referred to as w and o for short) with respect to both the social and omnistic welfare

(m, D)	payment function and altruism-adjusted Clarke pivot rule
(w, D^{sw})	$p_i(\boldsymbol{b}) = h_i(\boldsymbol{b}_{-i}) - (1-\alpha_i)\sum_{j\neq i} b_j(f(\boldsymbol{b}))$ $h_i(\boldsymbol{b}_{-i}) = (1-\alpha_i+c_i)\sum_{j\neq i} b_j(a^{-i})$ $a^{-i} \in \arg\max_{a\in A} \sum_{j\neq i} b_j(a)$
(o, D^{sw})	$p_i(\boldsymbol{b}) = \frac{1}{1-\alpha_i} h_i(\boldsymbol{b}_{-i}) - \sum_{j\neq i} b_j(f(\boldsymbol{b}))$ $h_i(\boldsymbol{b}_{-i}) = (1-\alpha_i+c_i)\sum_{j\neq i} b_j(a^{-i})$ $a^{-i} \in \arg\max_{a\in A} \sum_{j\neq i} b_j(a)$
(w, D^{ow})	$p_i(\boldsymbol{b}) = h_i(\boldsymbol{b}_{-i}) - \sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i\right) b_j(f(\boldsymbol{b}))$ $h_i(\boldsymbol{b}_{-i}) = \sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i + c_i\right) b_j(a^{-i})$ $a^{-i} \in \arg\max_{a\in A} \sum_{j\neq i}\left(1+\sum_{k\neq j,i}\alpha_k - \alpha_i\sum_{k\neq i}\alpha_k\right) b_j(a)$
(o, D^{ow})	$p_i(\boldsymbol{b}) = h_i(\boldsymbol{b}_{-i}) - \frac{1}{1-\alpha_i}\sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i\right) b_j(f(\boldsymbol{b}))$ $h_i(\boldsymbol{b}_{-i}) = \frac{1}{1-\alpha_i}\sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i + c_i\right) b_j(a^{-i})$ $a^{-i} \in \arg\max_{a\in A} \sum_{j\neq i}\left(1+\sum_{k\neq j,i}\alpha_k - \alpha_i\sum_{k\neq i}\alpha_k\right) b_j(a)$

Table 7.1: Definition of the payment function of the AAVCG mechanisms and its altruism-adjusted Clarke pivot rule, depending on the altruism model and design objective. The parameter c_i can be fixed arbitrarily in the range $[0, \alpha_i]$

objective. In order to keep the presentation concise, we introduce the following generic definition of adjusted VCG mechanisms. The respective payment functions $\boldsymbol{p}$ are stated in Table 7.1.

Definition 7.5.3. Let $m \in \{w, o\}$ refer to an altruism model as defined above and let $D \in \{D^{sw}, D^{ow}\}$ be a design objective. A mechanism $M^{m,D} = (f, \boldsymbol{p})$ is called an *altruism-adjusted VCG mechanism (AAVCG) with respect to altruism model m and design objective D* if the following two conditions are satisfied:

1. $f(\boldsymbol{b}) \in \arg\max_{a\in A} D(\boldsymbol{b}, a)$.
2. For every player $i \in N$ and some function $h_i : V_{-i} \to \mathbb{R}$, the payment function $p_i(\boldsymbol{b})$ is defined as in Table 7.1.

Similarly, we give a generic definition of an altruism-adjusted Clarke pivot rule for these mechanisms. The respective definitions of the functions h_i are stated in Table 7.1.

Definition 7.5.4. We say that an AAVCG mechanism $M^{m,D} = (f, \boldsymbol{p})$ with respect to altruism model m and design objective D implements the *altruism-adjusted Clarke pivot rule* if for every player $i \in N$ there is some $c_i \in [0, \alpha_i]$ such that the function h_i is as defined in Table 7.1.

With the help of Theorem 7.3.5 and Proposition 7.3.6, we can show that the four AAVCG mechanisms as specified in Table 7.1 are truthful and satisfy NPT and IR.

Theorem 7.5.5. *Every AAVCG mechanisms $M^{m,D} = (f, \boldsymbol{p})$ with respect to altruism model m and design objective D is truthful. Further, $M^{m,D}$ satisfies NPT and IR if it implements the altruism-adjusted Clarke pivot rule.*

Proof: We will check for every combination that it fits the design template from Theorem 7.3.5.

For both models with the social welfare objective we use $\gamma_i(\boldsymbol{b}_{-i}) = 1$.

(w, sw)**:** We verify

$$\begin{aligned} p_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) - (1-\alpha_i)\sum_{j \neq i} b_j(f(\boldsymbol{b})) \\ &= h_i(\boldsymbol{b}_{-i}) + \alpha_i \sum_{j \neq i} b_j(f(\boldsymbol{b})) - \sum_{j \neq i} b_j(f(\boldsymbol{b})) \\ &= h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - 1 \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})). \end{aligned}$$

(o, sw)**:** For this combination we see

$$\begin{aligned} && p_i(\boldsymbol{b}) &= \frac{1}{1-\alpha_i} h_i(\boldsymbol{b}_{-i}) - \sum_{j \neq i} b_j(f(\boldsymbol{b})) \\ \iff && (1-\alpha_i) p_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) - (1-\alpha_i)\sum_{j \neq i} b_j(f(\boldsymbol{b})) \\ \iff && p_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) + \alpha_i p_i(\boldsymbol{b}) + \alpha_i \sum_{j \neq i} b_j(f(\boldsymbol{b})) - \sum_{j \neq i} b_j(f(\boldsymbol{b})) \\ \iff && p_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) - 1 \cdot \sum_{j \neq i} b_j(f(\boldsymbol{b})), \end{aligned}$$

and thus it fits the criteria of Theorem 7.3.5.

For both the models with the omnistic welfare we take $\gamma_i(\boldsymbol{b}_{-i}) = \frac{1}{1 + \sum_{k \neq i} \alpha_k}$.

(w, ow)**:** Then

$$\begin{aligned}
p_i(\boldsymbol{b}) &= h_i(\boldsymbol{b}_{-i}) - \sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i\right) b_j(f(\boldsymbol{b}))\\
&= h_i(\boldsymbol{b}_{-i}) + \alpha_i \sum_{j\neq i} b_j(f(\boldsymbol{b})) - \sum_{j\neq i}\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} b_j(f(\boldsymbol{b}))\\
&= h_i(\boldsymbol{b}_{-i}) + \alpha_i \sum_{j\neq i} b_j(f(\boldsymbol{b})) - \frac{1}{1+\sum_{k\neq i}\alpha_k}\sum_{j\neq i}\left(1+\sum_{k\neq j}\alpha_k\right) b_j(f(\boldsymbol{b}))\\
&= h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - \gamma_i(\boldsymbol{b}_{-i})D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})).
\end{aligned}$$

(o, ow) : Finally,

$$\begin{array}{rrl}
 & p_i(\boldsymbol{b}) = & \dfrac{1}{1-\alpha_i}h_i(\boldsymbol{b}_{-i}) - \displaystyle\sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i\right) b_j(f(\boldsymbol{b}))\\
\Longleftrightarrow & (1-\alpha_i)p_i(\boldsymbol{b}) = & h_i(\boldsymbol{b}_{-i}) - (1-\alpha)\displaystyle\sum_{j\neq i}\left(\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} - \alpha_i\right) b_j(f(\boldsymbol{b}))\\
 & & h_i(\boldsymbol{b}_{-i}) + \alpha_i p_i(\boldsymbol{b}) + \alpha_i \displaystyle\sum_{j\neq i} b_j(f(\boldsymbol{b})) -\\
\Longleftrightarrow & p_i(\boldsymbol{b}) = & \\
 & & \displaystyle\sum_{j\neq i}\frac{1+\sum_{k\neq j}\alpha_k}{1+\sum_{k\neq i}\alpha_k} b_j(f(\boldsymbol{b}))\\
 & & h_i(\boldsymbol{b}_{-i}) + \alpha_i\left(\displaystyle\sum_{j\neq i} b_j(f(\boldsymbol{b})) + p_i(\boldsymbol{b})\right) -\\
\Longleftrightarrow & p_i(\boldsymbol{b}) = & \\
 & & \dfrac{1}{1+\sum_{k\neq i}\alpha_k}\displaystyle\sum_{j\neq i}\left(1+\sum_{k\neq j}\alpha_k\right) b_j(f(\boldsymbol{b}))\\
\Longleftrightarrow & p_i(\boldsymbol{b}) = & h_i(\boldsymbol{b}_{-i}) + g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})), \boldsymbol{p}(\boldsymbol{b})) - \gamma_i(\boldsymbol{b}_{-i})D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})).
\end{array}$$

Hence, we see that all four combinations fit the criteria of Theorem 7.3.5 and so they are truthful. □

7.5.3 Discussion

We discuss a few main properties of the mechanisms introduced above.

First note that the two AAVCG mechanisms for the social welfare objective reduce to the standard VCG mechanism (Definition 2.5.7) if the players are entirely selfish, i.e., $\alpha_i = 0$ for all i. This is to be expected because in this case both the welfare-oriented model and the omnistic model reduce to the standard utility model.

Further, these mechanisms nicely capture the intuition that altruism counters the negative effect of egoistic predispositions: As the altruism level of a player i increases, the designer needs to pay them less to have them want to reveal the truth about their valuation functions. And in fact, as we would expect the players require no extra incentive at all when they are fully altruistic ($\alpha_i = 1$ for all i).

Observe that the altruism-adjusted Clarke pivot gives rise to a family of mechanisms (parametrized by $c_i \in [0, \alpha_i]$ for all i). The size of this set grows with the altruism levels α_i of the players. This flexibility can be exploited to extract smaller payments from the players.

The altruism-adjusted Clarke pivot rule has a particularly nice representation in the omnistic model with respect to the social welfare objective, i.e., for every $c_i \in [0, \alpha_i]$

$$p_i(\boldsymbol{b}) = \left(1 + \frac{c_i}{1 - \alpha_i}\right) \sum_{j \neq i} b_j(a^{-i}) - \sum_{j \neq i} b_j(f(\boldsymbol{b})), \tag{7.8}$$

where a^{-i} is as defined in Table 7.1.

Note that by choosing $c_i = 0$ for every i, the resulting AAVCG mechanism reduces to the standard VCG mechanism with the Clarke pivot rule. In particular, this means that for this setting the standard VCG mechanism (not taking care of any other-regarding preferences) is truthful.

7.6 Impact of Altruism

First, we apply our altruism-adjusted VCG mechanism derived in the previous section to classical problems in mechanism design, bilateral trade, and funding of a public project. Using standard VCG for these problems results in a mechanism with undesirable properties. If the players are sufficiently altruistic, these undesirable properties get resolved. Next, we show that we can convert a standard VCG mechanism into an altruism-adjusted VCG mechanism with smaller payments. The more altruistic the players, the lower the payments.

7.6.1 Bilateral Trade

A buyer is interested in some object and values it at v_b, while some seller has the object and values it at v_s. We want a mechanism in which the players reveal their true preferences. In a mechanism design context this is modeled as follows:

(i) The set of alternatives is $A = \{\textit{trade}, \textit{no-trade}\}$.

(ii) The buyer has valuation function v_b defined as

$$\begin{aligned} v_b(\mathit{trade}) &= v_b \\ v_b(\mathit{no\text{-}trade}) &= 0. \end{aligned}$$

(iii) The seller has valuation function v_s defined as

$$\begin{aligned} v_s(\mathit{trade}) &= -v_s \\ v_s(\mathit{no\text{-}trade}) &= 0. \end{aligned}$$

The seller has a negative value for the trade because they will give away the item when a trade happens. By using standard VCG and insisting on no payments when there is no trade we will see that the mechanism needs to be subsidized, which is undesirable. Requiring that no payments are made when there is no trade implies that if $v_b < v_s$, then $p_s((v_b, v_s)) = h_s(v_b) - 0 = 0$ and $p_b((v_b, v_s)) = h_b(v_s) - 0 = 0$ implying that $h_s(v_b) = h_b(v_s) = 0$. But when a trade happens i.e., if $v_b \geq v_s$, then $p_s((v_b, v_s)) = 0 - v_b = -v_b$ and $p_b((v_b, v_s)) = 0 - -v_s = v_s$. Hence the seller receives v_b, while the buyer pays v_s. Thus, if $v_b > v_s$ the mechanism needs to be subsidized. It is of course very undesirable that a third party has to subsidize the mechanism to make the players reveal their true preferences.

In contrast, suppose the buyer has an altruism level of $-\alpha_b$ (the seller has a negative valuation for trading, so the negative altruism level of the buyer reflects the belief that the seller loses more value than what they report), and the seller of α_s, then using the AAVCG mechanism in the welfare-oriented model we deduce in the same way that $h_s = h_b = 0$ but now we charge the buyer $(1 + \alpha_b)v_s$ and the seller gets $(1 - \alpha_s)v_b$ if the trade happens. Hence, if

$$(1 + \alpha_b)v_s \geq (1 - \alpha_s)v_b,$$

the mechanism runs without the need of subsidizing it. And the greater the altruism levels of the players, the more likely that this happens.

7.6.2 Funding a Public Project

In the *public project problem* a contractor (e.g., government) considers undertaking a public project (e.g., building a bridge) at a commonly known cost C. Each player $i \in N$ (e.g., citizen) reports a value b_i that the realization of the project is worth to them (not necessarily equal to their private value v_i). Given the bids $\boldsymbol{b} = (b_1, \dots, b_n)$, the contractor determines whether the project is realized and what the contribution $p_i(\boldsymbol{b})$ of every player $i \in N$ is. Here the project is realized if and only if it can be *funded* by the players, i.e., $\sum_{i \in N} p_i(\boldsymbol{b}) \geq C$.

This models a very realistic situation. It would be desirable that the theory of mechanism design provides us with a mechanism that ensures that the project is undertaken when it should be undertaken, i.e., when the actual value created by the realization of the project is at least C; formally, $\sum_{i \in N} v_i \geq C$. This is precisely what a truthful mechanism maximizing social welfare would achieve. Unfortunately, the only instances of the public project problem that can be solved in the standard mechanism design setting are trivial ones.

Formally, the *public project problem* in a mechanism design context can be defined as follows (after Clarke (1971)):

(i) The set of alternatives is $A = \{yes, no\}$.

(ii) The set of players consists of $N = \{1, \ldots, n\}$ and a special player 0, representing the contractor.

(iii) Player $i = 0$ has a singleton valuation set $V_i = \{v_i\}$ with $v_i(yes) = -C$ for some $C \in \mathbb{R}^+$ and $v_i(no) = 0$.

(iv) Every player $i \in N$ has a valuation set V_i such that for every $v_i \in V_i$ we have $v_i(yes) = w_i$ for some $w_i \in \mathbb{R}^+$ and $v_i(no) = 0$.

(v) The design objective is the social welfare D^{sw}.

Note that player $i = 0$ is essentially a dummy player as they do not have any choice other than reporting their valuation function truthfully. Also, their valuation is $-C$ (reflecting that the realization of the project incurs a cost to them). In particular, they cannot be asked to contribute anything to the project.

Why standard mechanism design fails. The VCG mechanism with the Clarke pivot rule is known to be the only truthful mechanism that satisfies individual rationality (given the social welfare design objective); see, e.g., Nisan et al., 2007. In order to understand how this mechanism determines the payments in the public project problem, we need the following concept: Given the bids $\boldsymbol{b} \in V$, a player $i \in N$ is called *pivotal* if $\sum_j b_j \geq C$ but $\sum_{j \neq i} b_j < C$; otherwise, i is *non-pivotal*. In other words, a pivotal player is essential to make the project fundable.

The following proposition characterizes when a project is funded:

Proposition 7.6.1. ((Nisan et al., 2007)). *Using the VCG mechanism with the Clarke pivot rule, the public project in the standard utility model is funded if and only if*

1. $b_i \geq C$ *for some* $i \in N$ *and* $b_j = 0$ *for all* $j \in N$, $j \neq i$, *or*

2. $\sum_{i \in N} b_i = C$.

Proof: The payment of each player i is $C - \sum_{j \neq i} b_j$ if i is pivotal and 0 otherwise.

The if part for the first condition follows because in this case every player i with non-zero b_i is pivotal and thus

$$\sum_{i \in N} p_i(\boldsymbol{b}) = \sum_{i \in N} \left(C - \sum_{j \neq i} b_j \right) = nC - (n-1) \sum_{i \in N} b_i = C.$$

The if part of the second condition follows because the pivotal player i pays C.

For the only-if part we assume that both conditions do not hold. Only pivotal players have positive payment, thus if there are none then the project cannot be funded. Hence, we may assume there is at least one pivotal player i. Their payment is $C - \sum_{j \neq i} b_j$. To have the project funded the remaining players should pay at least $\sum_{j \neq i} b_j$. As condition 1 does not hold, we know that this is a positive amount. A player pays never more than their bid, but to reach $\sum_{j \neq i} b_j$ all players should pay their bid. This only happens if every player is pivotal, which is impossible by our assumption that condition 2 does not hold. □

As a consequence, a public project is only funded if there is exactly one player who benefits from it, or if there is no benefit at all but just a break-even between the value created and the investment costs incurred.

We next show how altruism helps to escape the above dilemma. More specifically, we consider the omnistic model and use the VCG mechanism with the altruism-adjusted Clarke pivot rule (7.8) with $c_i = \alpha_i$ for all i.

Proposition 7.6.2. *Let N_p be the set of pivotal players, and N_n the set of non-pivotal players. Using the VCG mechanism with the altruism-adjusted Clarke pivot rule (choosing $c_i = \alpha_i$ for all i), the public project in the omnistic model is funded if and only if*

$$\sum_{i \in N_p} \left(C - \sum_{j \neq i} b_j \right) + \sum_{i \in N_n} \frac{\alpha_i}{1 - \alpha_i} \left(\sum_{j \neq i} b_j - C \right) \geq C.$$

Proof: The payment is given by

$$p_i(\boldsymbol{b}) = \frac{1}{1 - \alpha_i} \left(\sum_{j \neq i} v_j(a^{-i}) + v_0(a^{-i}) \right) - \sum_{j \neq i} v_j(f(\boldsymbol{b})) - v_0(f(\boldsymbol{b})).$$

If i is a pivotal player and $f(\boldsymbol{b}) = \mathit{yes}$ then

$$p_i(\boldsymbol{b}) = 0 - \sum_{j \neq i} b_j + C = C - \sum_{j \neq i} b_j,$$

while if i is non-pivotal then

$$\begin{aligned} p_i(\boldsymbol{b}) &= \frac{1}{1-\alpha_i}\left(\sum_{j\neq i} b_j - C\right) - \sum_{j\neq i} b_j + C \\ &= \frac{\alpha_i}{1-\alpha_i}\left(\sum_{j\neq i} b_j - C\right). \end{aligned}$$

Hence the sum of the payments is

$$\sum_i p_i(\boldsymbol{b}) = \sum_{i\in N_p}\left(C - \sum_{j\neq i} b_j\right) + \sum_{i\in N_n}\frac{\alpha_i}{1-\alpha_i}\left(\sum_{j\neq i} b_j - C\right),$$

and this has to be at least C to have the project funded. □

What does Proposition 7.6.2 tell us? First of all, we see that the project is more likely to be fully funded when it is more profitable for the group to undertake it. For the pivotal players, the higher the contribution of the other players, the higher $C - \sum_{j\neq i} b_j$, and for the non-pivotal players it is clear that the more profitable the project is the higher $\sum_{j\neq i} b_j - C$. This relation is rather satisfying, especially if one compares it with the results for the standard utility model (where, paradoxically, the larger the net benefits of the project are, the less likely it is that it will be funded).

Secondly, we observe that altruism of non-pivotal players *always* has a positive effect on the likelihood of funding the project and this effect is amplified when the altruism levels of the (non-pivotal) players with small valuation is large. This effect becomes even more apparent if one considers uniform altruism levels ($\alpha_i = \alpha$ for all i) and only non-pivotal players. The condition of Proposition 7.6.2 simplifies to

$$\alpha\left(\sum_{j\in N} b_j - C\right) \geq \frac{C}{n-1}.$$

Also, note that the number of players n is positively related to the likelihood of funding the project.

7.6.3 Minimizing Payments

In this section, we assume that Assumption 7.3.2 holds. Given a mechanism where we do not take altruism into account but where players are actually altruistic, we will transform it into a mechanism with a smaller total payment. Consider welfare-oriented other-regarding preferences. Let every player i have an altruism level of α_i. Define $\alpha = \min_i \alpha_i$. Suppose we have some non-deficit mechanism satisfying the requirements of Theorem 7.3.5 with $\gamma = 1$ without taking the other-regarding preferences into account, i.e., $g_i = 0$ (one can think of standard

VCG). Hence, we have $h_i(\boldsymbol{b}_{-i})$ such that for $p_i(v) = h_i(\boldsymbol{b}_{-i}) - D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))$ the mechanism is non-deficit. Lemma 7.4.2 implies that

$$h_i(\boldsymbol{b}_{-i}) \geq \sup_{b'_i} \sum_j D_{-j}(\boldsymbol{b}'_{-j}, f(\boldsymbol{b}')) - \sum_{j \neq i} h_j(\boldsymbol{b}'_{-i}). \tag{7.9}$$

Let $g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) = \alpha_i \sum_{j \neq i} b_j(f(\boldsymbol{b}))$. And let $g'_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) = \alpha \sum_{j \neq i} b_j(f(\boldsymbol{b}))$. First note that $g'_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) = \alpha \cdot D_{-i}(\boldsymbol{b}_{-i}, f(\boldsymbol{b}))$ and also observe that by choice of α it holds that $g'_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) \leq g_i(\boldsymbol{b}_{-i}(f(\boldsymbol{b})))$. This will allow us to invoke Proposition 7.3.7 below.

We manipulate equation (7.9) by multiplying both sides by $(1 - \alpha)$

$$\begin{aligned}(1-\alpha)h_i(\boldsymbol{b}_{-i}) &\geq \sup_{b'_i} \sum_j (1-\alpha) D_{-j}(\boldsymbol{b}'_{-j}, f(\boldsymbol{b}')) - \sum_{j \neq i} (1-\alpha) h_j(\boldsymbol{b}'_{-i}) \\ &= \sup_{b'_i} \sum_j D_{-j}(\boldsymbol{b}'_{-j}, f(\boldsymbol{b}')) - g'_j(\boldsymbol{b}'_{-j}(f(\boldsymbol{b}'))) - \sum_{j \neq i} (1-\alpha) h_j(\boldsymbol{b}'_{-i}).\end{aligned}$$

Thus, substituting $(1 - \alpha)h_i$ by h'_i, shows that h'_i satisfies Lemma 7.4.2 and so the mechanism with respect to h'_i is non-deficit.

Because $g'_i \leq g_i$ Proposition 7.3.7 tells us that our mechanism is also IR and truthfulness follows from Theorem 7.3.5.

We take a look at the payments. First, observe that

$$\begin{aligned}p'_i(\boldsymbol{b}) &= h'_i(\boldsymbol{b}_{-i}) + g'_j(\boldsymbol{b}_{-i}(f(\boldsymbol{b}))) - D_{-j}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})) \\ &= (1-\alpha)h_i(\boldsymbol{b}_{-i}) - (1-\alpha)D_{-j}(\boldsymbol{b}_{-i}, f(\boldsymbol{b})) = (1-\alpha)p_i(\boldsymbol{b}).\end{aligned}$$

And so if $\alpha > 0$, every player pays or receives a factor $(1 - \alpha)$ less, and also the total payment $\sum_i p'_i(\boldsymbol{b}) = (1 - \alpha) \sum_{-i} p_i(\boldsymbol{b})$ is reduced by a factor $(1 - \alpha)$.

We can take any VCG mechanism that does not take altruism into account and convert it into one that does and automatically has payments that are reduced by a factor $(1 - \alpha)$.

Example 7.6.3. (Multi-Unit Auction). Consider a multi-unit auction with k identical goods. At most one item will be allocated to each player[3].

Standard VCG without other-regarding preferences charges the highest losing bid to all players. We can capture this in the welfare-oriented model with the classical social welfare objective by setting $\alpha_i = 0$ for all $i \in N$ and $h_i(\boldsymbol{b}_{-i}) = \sum_{j=1}^k [\boldsymbol{b}_{-i}]_j$. Let $\alpha = \min_{i \in N} \alpha_i$ where α_i is the altruism level of player i. If we require NPT we can take $c_i = 0$ for all i to minimize payments and obtain

$$h_i(\boldsymbol{b}_{-i}) = (1 - \alpha) \sum_{j=1}^k [\boldsymbol{b}_{-i}]_j.$$

[3] Note that this is different from the multi-unit auction in Chapter 6 where multiple items could be allocated to a single bidder.

As also the second part of the payment (see Table 7.1) is multiplied by $(1-\alpha)$ the total payments are a factor $(1-\alpha)$ lower than in standard VCG without altruism.

Not insisting on NPT (but still requiring the mechanism to be non-deficit) allows us to do better. Already in the standard VCG setting, we can apply the Bailey-Cavallo redistribution function (Bailey, 1997; Cavallo, 2006). This corresponds to

$$h_i(\boldsymbol{b}_{-i}) = \left(\sum_{j=1}^{k} [\boldsymbol{b}_{-i}]_j - \frac{k}{n}[\boldsymbol{b}_{-i}]_{k+1} \right),$$

significantly reducing the sum of payments to

$$\sum_i \boldsymbol{p}_i(\boldsymbol{b}) = \sum_{j=1}^{k} [\boldsymbol{b}]_j - k \cdot [\boldsymbol{b}]_{k+1}.$$

Applying the theory from this subsection we can take

$$h_i(\boldsymbol{b}_{-i}) = (1-\alpha) \left(\sum_{j=1}^{k} [\boldsymbol{b}_{-i}]_j - \frac{k}{n}[\boldsymbol{b}_{-i}]_{k+1} \right),$$

and still have a non-deficit, individually rational and truthful mechanism saving a factor $(1-\alpha)$ on the payments:

$$\sum_{i \in N} \boldsymbol{p}_i(\boldsymbol{b}) = (1-\alpha) \left(\sum_{j=1}^{k} [\boldsymbol{b}]_j - k \cdot [\boldsymbol{b}]_{k+1} \right).$$

It is even true that the last mechanism is individually undominated. Taking $b'_i = [\boldsymbol{b}_{-i}]_{k+1}$ in the supremum of equation (7.5) will result in equality. For example for $k=1$ and $i=1$ we have

$$\sum_j D_{-j}(\boldsymbol{b}_{-1}, [\boldsymbol{b}_{-1}]_2) = (1-\alpha)(n-1)[\boldsymbol{b}_{-1}]_1$$

and

$$\sum_{j \neq 1} h_j(\boldsymbol{b}'_{-j}) = (1-\alpha) \left((n-2)[\boldsymbol{b}'_{-j}]_1 + \frac{[\boldsymbol{b}'_{-j}]_2}{n} \right).$$

Note that $[\boldsymbol{b}_{-j}]_1 = [\boldsymbol{b}_{-1}]_1$ and $[\boldsymbol{b}'_{-j}]_2 = [\boldsymbol{b}_{-1}]_2$. Taking the difference is equal to $h_1(\boldsymbol{b}_{-1})$. This equality can be verified for all k and i.

7.7 Conclusion

We have introduced a general utility model that incorporates other-regarding preferences with the specific aim of modeling altruism. In addition to characterizing truthful mechanisms in this model, we provide a recipe for obtaining them. With this recipe, we propose two different ways of modeling altruism and give case studies in which these models allow us to explain how the classic mechanism design problems of bilateral trade and funding of a public project become feasible. Of course, more other-regarding preferences are conceivable, and we hope that our framework gives an easy tool to come up with truthful mechanisms with respect to those other-regarding preferences.

Following Guo, Markakis, et al. (2013), we have characterized what individually undominated mechanisms look like in our model. Moreover, we have shown that any VCG mechanism not taking altruism into account can be converted into a truthful mechanism that does take altruism into account with lower payments.

Under some assumptions, we have seen that it is possible to have truthful mechanisms based on the 'wrong' other-regarding preferences. Because the other-regarding preferences are an integral part of the characterization of individually undominated mechanisms, it is not possible to compare mechanisms with different other-regarding preferences. It would be nice to be able to characterize mechanisms that are undominated irrespective of the exact other-regarding preferences, as long as the mechanisms are truthful and non-deficit.

Another interesting next step is to see if it is possible to come up with mechanisms that are strongly budget-balanced. A starting point is looking into how Cavallo (2012) leverages the flexibility that altruism gives for this purpose.

Finally, on a conceptual level, we hope that there will be more research into how altruism can be incorporated into mechanism design.

Bibliography

Aggarwal, Charu C and Philip S Yu (2009). "A Survey of Uncertain Data Algorithms and Applications". In: *IEEE Transactions on Knowledge and Data Engineering* 21.5, pages 609–623 (cited on page 66).

Ahuja, Ravindra K, Kurt Mehlhorn, James Orlin, and Robert E Tarjan (1990). "Faster algorithms for the shortest path problem". In: *Journal of the ACM* 37.2, pages 213–223 (cited on page 61).

Amici, Raul, Marco Bonola, Lorenzo Bracciale, Antonello Rabuffi, Pierpaolo Loreti, and Giuseppe Bianchi (2014). "Performance assessment of an epidemic protocol in VANET using real traces". In: *Procedia Computer Science* 40, pages 92–99 (cited on page 90).

Andreoni, James and John Miller (2002). "Giving according to GARP: An experimental test of the consistency of preferences for altruism". In: *Econometrica* 70.2, pages 737–753 (cited on pages 6, 164).

Apt, Krzysztof R and Guido Schäfer (2014). "Selfishness level of strategic games". In: *Journal of Artificial Intelligence Research* 49, pages 207–240 (cited on page 166).

Avrachenkov, Konstantin and Nelly Litvak (2006). "The Effect of New Links on Google Pagerank". In: *Stochastic Models* 22.2, pages 319–331 (cited on page 37).

Awerbuch, Baruch, Yossi Azar, Yossi Richter, and Dekel Tsur (2006). "Tradeoffs in worst-case equilibria". In: *Theoretical Computer Science* 361.2-3, pages 200–209 (cited on page 104).

Bader, David A., Shiva Kintali, Kamesh Madduri, and Milena Mihail (2007). "Approximating betweenness centrality". In: *Algorithms and Models for the Web-Graph*, pages 124–137 (cited on page 84).

Bailey, Martin J (1997). “The demand revealing process: to distribute the surplus”. In: *Public Choice* 91.2, pages 107–126 (cited on pages 164, 167, 186).

Ball, Michael O. (1986). “Computational complexity of network reliability analysis: An overview”. In: *IEEE Transactions on Reliability* 35.3, pages 230–239 (cited on pages 65, 67).

Barabási, Albert-László and Réka Albert (1999). “Emergence of scaling in random networks”. In: *Science* 286.5439, pages 509–512 (cited on page 87).

Bauer, Reinhard, Gianlorenzo D’Angelo, Daniel Delling, Andrea Schumm, and Dorothea Wagner (2012). “The Shortcut Problem - Complexity and Algorithms”. In: *Journal of Graph Algorithms and Applications* 16.2, pages 447–481 (cited on page 37).

Beek, Andries van, Ruben Brokkelkamp, and Guido Schäfer (2022). “Corruption in Auctions: Social Welfare Loss in Hybrid Multi-Unit Auctions”. In: *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. Edited by Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor. International Foundation for Autonomous Agents and Multiagent Systems, pages 1283–1291. DOI: 10.5555/3535850.3535993 (cited on page 9).

Bellman, Richard (1958). “On a routing problem”. In: *Quarterly of applied mathematics* 16.1, pages 87–90 (cited on page 61).

Bergamini, Elisabetta, Pierluigi Crescenzi, Gianlorenzo D’Angelo, Henning Meyerhenke, Lorenzo Severini, and Yllka Velaj (2018). “Improving the Betweenness Centrality of a Node by Adding Links”. In: *ACM Journal of Experimental Algorithmics* 23 (cited on pages 33, 37).

Bilò, Davide, Luciano Gualà, and Guido Proietti (2012). “Improved approximability and non-approximability results for graph diameter decreasing problems”. In: *Theoretical Computer Science* 417, pages 12–22 (cited on page 37).

Bilò, Vittorio (2018). “A unifying tool for bounding the quality of non-cooperative solutions in weighted congestion games”. In: *Theory of Computing Systems* 62.5, pages 1288–1317 (cited on pages 103, 104, 111).

Birmpas, Georgios, Evangelos Markakis, Orestis Telelis, and Artem Tsikiridis (2019). “Tight Welfare Guarantees for Pure Nash Equilibria of the Uniform Price Auction”. In: *Theory of Computing Systems* 63.7, pages 1451–1469. DOI: 10.1007/s00224-018-9889-7 (cited on pages 133, 136, 137, 148, 149).

Boldi, Paolo, Francesco Bonchi, Aristides Gionis, and Tamir Tassa (July 2012). “Injecting uncertainty in graphs for identity obfuscation”. In: *Proceedings of the VLDB Endowment* 5.11, pages 1376–1387. DOI: 10.14778/2350229.2350254 (cited on page 66).

Bonchi, Francesco, Aristides Gionis, Francesco Gullo, and Antti Ukkonen (2014). “Distance oracles in edge-labeled graphs”. In: *Proceedings of the 17th International Conference on Extending Database Technology*, pages 547–558 (cited on page 66).

Borgs, Christian, Michael Brautbar, Jennifer Chayes, and Brendan Lucier (2014). "Maximizing social influence in nearly optimal time". In: *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 946–957 (cited on page 66).

Brandes, Ulrik (2001). "A faster algorithm for betweenness centrality". In: *Journal of Mathematical Sociology* 25.2, pages 163–177 (cited on pages 66, 84).

Brandt, Felix and Gerhard Weiß (2001). "Antisocial agents and Vickrey auctions". In: *International Workshop on Agent Theories, Architectures, and Languages*. Springer, pages 335–347 (cited on page 166).

Briest, Patrick (2008). "Uniform Budgets and the Envy-Free Pricing Problem". In: *Automata, Languages and Programming*, pages 808–819 (cited on page 38).

Briest, Patrick, Parinya Chalermsook, Sanjeev Khanna, Bundit Laekhanukit, and Danupon Nanongkai (2010). "Improved Hardness of Approximation for Stackelberg Shortest-Path Pricing". In: *Internet and Network Economics*, pages 444–454 (cited on page 38).

Briest, Patrick, Martin Hoefer, and Piotr Krysta (Apr. 1, 2012). "Stackelberg Network Pricing Games". In: *Algorithmica* 62.3, pages 733–753. DOI: 10.1007/s00453-010-9480-3 (cited on page 37).

Brokkelkamp, Ruben, Sjir Hoeijmakers, and Guido Schäfer (2022). "Greater Flexbility in Mechanism Design Through Altruism". In: *International Symposium on Algorithmic Game Theory*. Springer (cited on page 9).

Brokkelkamp, Ruben, Sven C. Polak, Guido Schäfer, and Yllka Velaj (2019). "Approximate Pricing in Networks: How to Boost the Betweenness and Revenue of a Node". In: *30th International Symposium on Algorithms and Computation, ISAAC 2019, December 8-11, 2019, Shanghai University of Finance and Economics, Shanghai, China*. Edited by Pinyan Lu and Guochuan Zhang. Volume 149. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 13:1–13:15. DOI: 10.4230/LIPIcs.ISAAC.2019.13 (cited on page 7).

Brokkelkamp, Ruben, Raymond van Venetië, Mees J. de Vries, and Jan Westerdiep (2020). "PACE Solver Description: tdULL". In: *15th International Symposium on Parameterized and Exact Computation, IPEC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)*. Edited by Yixin Cao and Marcin Pilipczuk. Volume 180. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 29:1–29:4. DOI: 10.4230/LIPIcs.IPEC.2020.29 (cited on page 10).

Brotcorne, Luce, F. Cirinei, Patrice Marcotte, and Gilles Savard (May 1, 2011). "An exact algorithm for the network pricing problem". In: *Discrete Optimization* 8.2, pages 246–258. DOI: 10.1016/j.disopt.2010.09.003 (cited on page 37).

Buehler, Russell, Zachary Goldman, David Liben-Nowell, Yuechao Pei, Jamie Quadri, Alexa Sharp, Sam Taggart, Tom Wexler, and Kevin Woods (2011). "The price of civil society". In: *International Workshop on Internet and Network Economics*. Springer, pages 375–382 (cited on page 166).

Caragiannis, Ioannis, Christos Kaklamanis, Panagiotis Kanellopoulos, Maria Kyropoulou, Brendan Lucier, Renato Paes Leme, and Éva Tardos (2015). "Bounding the inefficiency of outcomes in generalized second price auctions". In: *Journal of Economic Theory* 156, pages 343–388 (cited on page 108).

Caragiannis, Ioannis, Christos Kaklamanis, Panagiotis Kanellopoulos, Maria Kyropoulou, and Evi Papaioannou (2010). "The impact of altruism on the efficiency of atomic congestion games". In: *International Symposium on Trustworthy Global Computing*. Springer, pages 172–188 (cited on page 166).

Carlson, James A, Arthur Jaffe, and Andrew Wiles (2006). *The Millennium Prize Problems* (cited on page 19).

Cavallo, Ruggiero (2006). "Optimal decision-making with minimal waste: Strategyproof redistribution of VCG payments". In: *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 882–889 (cited on pages 164, 167, 186).

— (2012). "Efficient Auctions with Altruism". Published online at `http://rc.ftml.net.user.fm/papers/cavallo-altru.pdf` (cited on pages 167, 187).

Chalermsook, Parinya, Bundit Laekhanukit, and Danupon Nanongkai (2013). "Independent Set, Induced Matching, and Pricing: Connections and Tight (Subexponential Time) Approximation Hardnesses". In: *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 370–379. DOI: `10.1109/FOCS.2013.47` (cited on pages 38, 53, 54).

Charness, Gary and Matthew Rabin (2002). "Understanding social preferences with simple tests". In: *The quarterly journal of economics* 117.3, pages 817–869 (cited on pages 6, 164).

Chehreghani, M. H. (2014). "An efficient algorithm for approximate betweenness centrality computation". In: *The Computer Journal* 57.9, pages 1371–1382 (cited on page 84).

Chen, Po-An, Bart De Keijzer, David Kempe, and Guido Schäfer (2014). "Altruism and its impact on the price of anarchy". In: *ACM Transactions on Economics and Computation* 2.4, pages 1–45 (cited on page 166).

Cheng, Yu-Rong, Ye Yuan, Lei Chen, and Guo-Ren Wang (2015). "Threshold-based shortest path query over large correlated uncertain graphs". In: *Journal of Computer Science and Technology* 30.4, pages 762–780 (cited on page 66).

Cheng, Yurong, Ye Yuan, Guoren Wang, Baiyou Qiao, and Zhiqiong Wang (2014). "Efficient sampling methods for shortest path query over uncertain graphs". In: *International Conference on Database Systems for Advanced Applications*, pages 124–140 (cited on pages 65, 66, 67, 87, 93).

Christodoulou, George, Annamária Kovács, Alkmini Sgouritsa, and Bo Tang (2016). "Tight Bounds for the Price of Anarchy of Simultaneous First-Price Auctions". In: *ACM Transactions on Economics and Computation* 4.2. DOI: `10.1145/2847520` (cited on page 136).

Clarke, Edward H (1971). "Multipart pricing of public goods". In: *Public choice*, pages 17–33 (cited on page 31).

Cole, Richard, José R Correa, Vasilis Gkatzelis, Vahab Mirrokni, and Neil Olver (2015). “Decentralized utilitarian mechanisms for scheduling games”. In: *Games and Economic Behavior* 92, pages 306–326 (cited on page 104).

Conway, R.W., W.L Maxwell, and L.W. Miller (1967). *Theory of Scheduling.* Addison Wesley, Reading, Mass. (cited on page 103).

Cook, Stephen A (1971). “The complexity of theorem-proving procedures”. In: *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158 (cited on page 20).

Cormen, Thomas H, Charles E Leiserson, Ronald L Rivest, and Clifford Stein (2009). *Introduction to algorithms.* MIT press (cited on pages 11, 19).

Correa, José R and Maurice Queyranne (2012). “Efficiency of equilibria in restricted uniform machine scheduling with total weighted completion time as social cost”. In: *Naval Research Logistics (NRL)* 59.5, pages 384–395 (cited on page 104).

Costa, Camila F., Mario A. Nascimento, José A.F. Macêdo, Yannis Theodoridis, Nikos Pelekis, and Javam Machado (2015). “Optimal time-dependent sequenced route queries in road networks”. In: *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–4 (cited on page 90).

Craddock, Cameron et al. (2013). “The neuro bureau preprocessing initiative: Open sharing of preprocessed neuroimaging data and derivatives”. In: *Frontiers in Neuroinformatics.* DOI: 10.3389/conf.fninf.2013.09.00041 (cited on pages 62, 96).

Crescenzi, Pierluigi, Gianlorenzo D’Angelo, Lorenzo Severini, and Yllka Velaj (2015). “Greedily improving our own centrality in a network”. In: *Proceedings of 14th International Symposium on Experimental Algorithms.* Volume 9125, pages 43–55 (cited on page 37).

— (2016). “Greedily Improving Our Own Closeness Centrality in a Network”. In: *ACM Trans. Knowl. Discov. Data* 11.1, 9:1–9:32 (cited on page 37).

Crescenzi, Pierluigi and Viggo Kann, editors (2000). *A compendium of NP optimization problems.* URL: https://www.csc.kth.se/tcs/compendium (cited on page 20).

Czumaj, Artur and Berthold Vöcking (2007). “Tight bounds for worst-case equilibria”. In: *ACM Transactions on Algorithms* 3.1, pages 1–17 (cited on page 104).

D’Angelo, Gianlorenzo, Martin Olsen, and Lorenzo Severini (2019). “Coverage centrality maximization in undirected networks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence.* Volume 33. 01, pages 501–508 (cited on page 37).

Das, Kousik, Sovan Samanta, and Madhumangal Pal (2018). “Study on centrality measures in social networks: a survey”. In: *Social network analysis and mining* 8.1, pages 1–11 (cited on page 33).

de Keijzer, Bart, Evangelos Markakis, Guido Schäfer, and Orestis Telelis (2013). “Inefficiency of Standard Multi-unit Auctions”. In: *Proceedings of the European*

Symposium on Algorithms, pages 385–396 (cited on pages 133, 135, 136, 143, 144, 148, 149).

De Marco, Giuseppe and Jacqueline Morgan (2011). "Altruistic behavior and correlated equilibrium selection". In: *International Game Theory Review* 13.04, pages 363–381 (cited on page 166).

Dekker, Wilco and Tom Kreling (Jan. 28, 2022). *Hoe werd Nederland een belastingparadijs (en hoe komen we er vanaf)?* De Volkskrant. URL: `https://www.volkskrant.nl/nieuws-achtergrond/hoe-werd-nederland-een-belastingparadijs-en-hoe-komen-we-er-vanaf~b53dce58/` (cited on page 2).

Delling, Daniel, Peter Sanders, Dominik Schultes, and Dorothea Wagner (2009). "Engineering route planning algorithms". In: *Algorithmics of large and complex networks*. Springer, pages 117–139 (cited on page 61).

Demaine, Erik D. and Morteza Zadimoghaddam (2010). "Minimizing the Diameter of a Network Using Shortcut Edges". In: *Proceedings of 12th Scandinavian Symposium and Workshop on Algorithm Theory (SWAT)*. Volume 6139, pages 420–431 (cited on page 37).

Di Martino, Adriana, Clare Kelly, Rebecca Grzadzinski, Xi-Nian Zuo, Maarten Mennes, María Mairena, Catherine Lord, Francisco Castellanos, and Michael Milham (Dec. 2010). "Aberrant striatal functional connectivity in children with autism". In: *Biological Psychiatry* 69.9, pages 847–56. DOI: `10.1016/j.biopsych.2010.10.029` (cited on pages 62, 96).

Dijkstra, Edsger W. (Dec. 1959). "A Note on Two Problems in Connexion with Graphs". In: *Numer. Math.* 1.1, pages 269–271. DOI: `10.1007/BF01386390` (cited on pages 15, 41).

Eppstein, David (1998). "Finding the k shortest paths". In: *SIAM Journal on Computing* 28.2, pages 652–673 (cited on page 66).

Erdős, Paul and Alfréd Rényi (1959). "On random graphs". In: *Publicationes Mathematicae Debrecen* 6, pages 290–297 (cited on page 87).

Fehr, Ernst and Urs Fischbacher (2002). "Why social preferences matter–the impact of non-selfish motives on competition, cooperation and incentives". In: *The economic journal* 112.478, pages C1–C33 (cited on page 166).

Fehr, Ernst and Klaus M Schmidt (1999). "A theory of fairness, competition, and cooperation". In: *The quarterly journal of economics* 114.3, pages 817–868 (cited on page 166).

Feige, Uriel (1998). "A Threshold of $\ln n$ for Approximating Set Cover". In: *J. of the ACM* 45.5, pages 634–652 (cited on pages 44, 49).

Feldman, Michal, Hu Fu, Nick Gravin, and Brendan Lucier (2013). "Simultaneous auctions are (almost) efficient". In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 201–210 (cited on page 137).

Feldman, Michal, Brendan Lucier, and Noam Nisan (2016). "Correlated and coarse equilibria of single-item auctions". In: *Proceedings of the International*

Conference on Web and Internet Economics (WINE). Springer, pages 131–144 (cited on pages 134, 137, 152, 153, 155).

Ford Jr, Lester R (1956). *Network flow theory*. Technical report. Rand Corp Santa Monica Ca (cited on page 61).

Freeman, Linton C. (1977). "A Set of Measures of Centrality Based on Betweenness". In: *Sociometry* 40.1, pages 35–41 (cited on pages 2, 13, 33, 66).

Gairing, Martin, Thomas Lücking, Marios Mavronicolas, and Burkhard Monien (2010). "Computing Nash equilibria for scheduling on restricted parallel links". In: *Theory of Computing Systems* 47.2, pages 405–432 (cited on page 104).

Gamzu, Iftah and Danny Segev (2010). "A Sublogarithmic Approximation for Highway and Tollbooth Pricing". In: *Proceedings of 37th International Colloquium Conference on Automata, Languages and Programming*. Berlin, Heidelberg, pages 582–593 (cited on page 38).

García Domínguez, Luis, Jim Stieben, José Luis Pérez Velázquez, and Stuart Shanker (Oct. 2013). "The imaginary part of coherency in autism: Differences in cortical functional connectivity in preschool children". In: *PLOS ONE* 8.10, pages 1–13. DOI: `10.1371/journal.pone.0075941` (cited on pages 62, 96).

Garey, Michael R and David S Johnson (1979). *Computers and intractability*. Volume 174. freeman San Francisco (cited on page 20).

Geisberger, Robert, Peter Sanders, and Dominik Schultes (2008). "Better approximation of betweenness centrality". In: *Proceedings of the Meeting on Algorithm Engineering and Expermiments*. San Francisco, California, pages 90–100 (cited on page 84).

Ghosh, Joy, Hung Q. Ngo, Seokhoon Yoon, and Chunming Qiao (2007). "On a routing problem within probabilistic graphs and its application to intermittently connected networks". In: *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 1721–1729 (cited on page 62).

Graham, Daniel A. and Robert C. Marshall (1987). "Collusive Bidder Behavior at Single-Object Second-Price and English Auctions". In: *Journal of Political Economy* 95.6, pages 1217–1239 (cited on page 135).

Graham, Ronald Lewis, Eugene Leighton Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan (1979). "Optimization and approximation in deterministic sequencing and scheduling: a survey". In: *Annals of discrete mathematics*. Volume 5. Elsevier, pages 287–326 (cited on page 101).

Grandoni, Fabrizio and Thomas Rothvoss (Jan. 1, 2016). "Pricing on Paths: A PTAS for the Highway Problem". In: *SIAM Journal on Computing* 45.2, pages 216–231. DOI: `10.1137/140998846` (cited on page 38).

Groves, Theodore (1973). "Incentives in teams". In: *Econometrica: Journal of the Econometric Society*, pages 617–631 (cited on page 31).

Guo, M., Vangelis Markakis, Krzysztof Apt, and Vincent Conitzer (2013). "Undominated Groves Mechanisms". In: *Journal of Artificial Intelligence Research* 46, pages 129–163 (cited on pages 165, 167, 173, 187).

Guo, Mingyu and Vincent Conitzer (2009). “Worst-case optimal redistribution of VCG payments in multi-unit auctions”. In: *Games and Economic Behavior* 67.1, pages 69–98 (cited on page 167).

— (2010). “Optimal-in-expectation redistribution mechanisms”. In: *Artificial Intelligence* 174.5-6, pages 363–381 (cited on page 167).

Guruswami, Venkatesan, Jason D. Hartline, Anna R. Karlin, David Kempe, Claire Kenyon, and Frank McSherry (2005). “On Profit-maximizing Envy-free Pricing”. In: *Proceedings of Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Philadelphia, PA, USA, pages 1164–1173 (cited on page 38).

Han, Kai, Fei Gui, Xiaokui Xiao, Jing Tang, Yuntian He, Zongmai Cao, and He Huang (2019). “Efficient and effective algorithms for clustering uncertain graphs”. In: *Proceedings of the VLDB Endowment* 12.6, pages 667–680 (cited on page 66).

Hannan, James (1957). “Approximation to Bayes risk in repeated play”. In: *Contributions to the Theory of Games* 3.2, pages 97–139 (cited on page 26).

Harsanyi, John C. (1967). “Games with Incomplete Information Played by “Bayesian” Players, Parts I, II and III.” In: *Management Science* 14.3, 159–182, 320–334, and 486–502 (cited on pages 29, 135).

Hart, Peter E, Nils J Nilsson, and Bertram Raphael (1968). “A formal basis for the heuristic determination of minimum cost paths”. In: *IEEE transactions on Systems Science and Cybernetics* 4.2, pages 100–107 (cited on page 61).

Heydenreich, Birgit, Rudolf Müller, and Marc Uetz (2007). “Games and mechanism design in machine scheduling—an introduction”. In: *Production and operations management* 16.4, pages 437–454 (cited on page 107).

Hoefer, Martin and Alexander Skopalik (2013). “Altruism in atomic congestion games”. In: *ACM Transactions on Economics and Computation* 1.4, pages 1–21 (cited on page 166).

Hoeffding, W. (1963). “Probability Inequalities for Sums of Bounded Random Variables”. In: *J. American Stat. Association* 58, pages 13–30 (cited on page 86).

Hoeijmakers, Sjir (2014). *How Altruism Changes Mechanism Design.* MSc thesis, Tilburg University, The Netherlands (cited on page 9).

Hoeksma, Ruben and Marc Uetz (2019). “The price of anarchy for utilitarian scheduling games on related machines”. In: *Discrete Optimization* 31, pages 29–39. DOI: `https://doi.org/10.1016/j.disopt.2018.08.001` (cited on pages 103, 104, 108, 109, 127).

Horowitz, Ellis and Sartaj Sahni (1976). “Exact and approximate algorithms for scheduling nonidentical processors”. In: *Journal of the ACM* 23.2, pages 317–327 (cited on page 106).

Hua, Ming and Jian Pei (2010). “Probabilistic path queries in road networks: Traffic uncertainty aware path selection”. In: *Proceedings of the 13th International Conference on Extending Database Technology*, pages 347–358 (cited on page 62).

Ibarra, Oscar H and Chul E Kim (1977). “Heuristic algorithms for scheduling independent tasks on nonidentical processors”. In: *Journal of the ACM* 24.2, pages 280–289 (cited on pages 101, 107).

Immorlica, Nicole, Li (Erran) Li, Vahab S. Mirrokni, and Andreas S. Schulz (Apr. 2009). “Coordination mechanisms for selfish scheduling”. In: *Theoretical Computer Science* 410.17, pages 1589–1598. DOI: `10.1016/j.tcs.2008.12.032` (cited on page 107).

Ingraham, Allan (2005). “A Test for Collusion between a Bidder and an Auctioneer in Sealed-Bid Auctions”. In: *Contributions in Economic Analysis & Policy* 4. DOI: `10.2202/1538-0645.1448` (cited on page 130).

Ishakian, Vatche, Dóra Erdös, Evimaria Terzi, and Azer Bestavros (2012). “A Framework for the Evaluation and Management of Network Centrality”. In: *Proceedings of 12th SIAM International Conference on Data Mining*, pages 427–438 (cited on page 37).

Jin, Ruoming, Lin Liu, Bolin Ding, and Haixun Wang (2011). “Distance-constraint reachability computation in uncertain graphs”. In: *Proceedings of the VLDB Endowment* 4.9, pages 551–562 (cited on pages 65, 66, 67).

Johnson, Donald B. (1977). “Efficient algorithms for shortest paths in sparse networks”. In: *Journal of the ACM* 24.1, pages 1–13 (cited on page 66).

Kahneman, Daniel (2011). *Thinking, fast and slow*. Macmillan (cited on pages 6, 164).

Karp, Richard M., Michael Luby, and Neal Madras (1989). “Monte-Carlo approximation algorithms for enumeration problems”. In: *Journal of Algorithms* 10.3, pages 429–448 (cited on pages 64, 66, 71, 74, 78).

Ke, Xiangyu, Arijit Khan, and Leroy Lim Hong Quan (2019). “An in-depth comparison of s-t reliability algorithms over uncertain graphs”. In: *Proceedings of the VLDB Endowment* 12.8, pages 864–876. DOI: `10.14778/3324301.3324304` (cited on page 66).

Keown, Christopher, Patricia Shih, Aarti Nair, Nick Peterson, Mark Mulvey, and Ralph-Axel Müller (2013). “Local functional overconnectivity in posterior brain regions is associated with symptom severity in autism spectrum disorders”. In: *Cell Reports* 5, pages 567–572. DOI: `10.1016/j.celrep.2013.10.003` (cited on page 96).

Kerkmann, Anna Maria and Jörg Rothe (2021). “Altruism in coalition formation games”. In: *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 347–353 (cited on page 166).

Khan, A., F. Bonchi, A. Gionis, and F. Gullo (2014). “Fast reliability search in uncertain graphs”. In: *Proceedings of the 17th International Conference on Extending Database Technology*, pages 535–546 (cited on pages 62, 66).

Khan, Arijit, Francesco Bonchi, Francesco Gullo, and Andreas Nufer (2018). “Conditional reliability in uncertain graphs”. In: *IEEE Transactions on Knowledge and Data Engineering* 30.1, pages 2078–2092 (cited on pages 65, 67).

Khan, Arijit, Yuan Ye, and Lei Chen (2018). *On Uncertain Graphs*. Volume 10. Synthesis Lectures on Data Management. Morgan & Claypool Publishers (cited on pages 65, 67).

Kojaku, Sadamori and Naoki Masuda (2019). “Constructing networks by filtering correlation matrices: A null model approach”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 475.2231, page 20190578. DOI: `10.1098/rspa.2019.0578` (cited on page 96).

Koutsoupias, Elias and Christos Papadimitriou (1999). “Worst-Case Equilibria”. In: *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, pages 404–413 (cited on pages 102, 104, 133).

Kozlovskaya, Maria and Antonio Nicolò (2019). “Public good provision mechanisms and reciprocity”. In: *Journal of Economic Behavior & Organization* 167, pages 235–244 (cited on page 166).

Kucuksenel, Serkan (2012). “Behavioral mechanism design”. In: *Journal of Public Economic Theory* 14.5, pages 767–789 (cited on page 166).

Kuiper, Alex, Michel Mandjes, Jeroen de Mast, and Ruben Brokkelkamp (2021). “A flexible and optimal approach for appointment scheduling in healthcare”. In: *Decision Sciences*. DOI: `https://doi.org/10.1111/deci.12517` (cited on page 10).

Labbe, Martine, Patrice Marcotte, and Gilles Savard (Dec. 1998). “A Bilevel Model of Taxation and Its Application to Optimal Highway Pricing”. In: *Management Science* 44.12, pages 1608–1622 (cited on page 37).

Lanciano, Tommaso, Francesco Bonchi, and Aristides Gionis (2020). “Explainable classification of brain networks via contrast subgraphs”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Virtual Event, CA, USA. DOI: `10.1145/3394486.3403383` (cited on page 96).

Lejour, Arjan M., Maarten van ‘t Riet, and Jan Möhlmann (2019). *Doorsluisland NL doorgelicht*. CPB Policy Brief (cited on page 2).

Lengwiler, Yvan and Elmar Wolfstetter (2010). “Auctions and corruption: An analysis of bid rigging by a corrupt auctioneer”. In: *Journal of Economic Dynamics and Control* 34.10, pages 1872–1892. DOI: `10.1016/j.jedc.2010.03.002` (cited on pages 130, 135, 136, 162).

Lengwiler, Yvan and Elmar G. Wolfstetter (2000). *Auctions and Corruption*. CESifo Working Paper Series 401. CESifo. URL: `https://ideas.repec.org/p/ces/ceswps/%5C_401.html` (cited on page 135).

Leskovec, Jure and Andrej Krevl (June 2014). *SNAP Datasets: Stanford Large Network Dataset Collection*. `http://snap.stanford.edu/data` (cited on page 97).

Levin, Leonid Anatolevich (1973). “Universal sequential search problems”. In: *Problemy peredachi informatsii* 9.3, pages 115–116 (cited on page 20).

Li, Xiaodong, Reynold Cheng, Yixiang Fang, Jiafeng Hu, and Silviu Maniu (Mar. 2018). “Scalable evaluation of k-NN queries on large uncertain graphs”.

In: *Proceedings of the 21st International Conference on Extending Database Technology*. Vienna, Austria, pages 181–192. DOI: `10.5441/002/edbt.2018.17`. URL: `https://hal.archives-ouvertes.fr/hal-01955609` (cited on page 66).

Liben-Nowell, David and Jon Kleinberg (2003). "The link prediction problem for social networks". In: *Proceedings of the Twelfth ACM International Conference on Information and Knowledge Management*. New Orleans, LA, USA, pages 556–559. DOI: `10.1145/956863.956972` (cited on page 66).

Lucier, Brendan and Renato Paes Leme (2011). "GSP auctions with correlated types". In: *Proceedings of the 12th ACM conference on Electronic commerce*, pages 71–80 (cited on page 108).

Luxen, Dennis and Christian Vetter (2011). "Real-time routing with OpenStreetMap data". In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 513–516 (cited on page 90).

Madden, Samuel (2004). *Intel lab data*. URL: `http://db.csail.mit.edu/labdata/labdata.html` (cited on page 95).

Mahmoody, Ahmad, Charalampos E. Tsourakakis, and Eli Upfal (2016). "Scalable Betweenness Centrality Maximization via Sampling". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1765–1773 (cited on pages 66, 85).

Markakis, Evangelos and Orestis Telelis (2015). "Uniform Price Auctions: Equilibria and Efficiency". In: *Theory of Computing Systems* 57.3, pages 549–575. DOI: `10.1007/s00224-014-9537-9` (cited on page 137).

Mas-Colell, Andreu, Michael Dennis Whinston, Jerry R Green, et al. (1995). *Microeconomic theory*. Volume 1. Oxford university press New York, pages 861–862 (cited on page 173).

Masuda, Naoki, Sadamori Kojaku, and Yukie Sano (July 2018). "Configuration model for correlation matrices preserving the node strength". In: *Physical Review E* 98 (1), page 012312. DOI: `10.1103/PhysRevE.98.012312` (cited on page 96).

McAfee, Randolph and John McMillan (1992). "Bidding Rings". In: *American Economic Review* 82.3, pages 579–99 (cited on page 135).

Menezes, Flavio M. and Paulo Klinger Monteiro (2006). "Corruption and auctions". In: *Journal of Mathematical Economics* 42.1, pages 97–108. DOI: `10.1016/j.jmateco.2005.04.002` (cited on pages 130, 135, 136).

Meyerson, Adam and Brian Tagiku (2009). "Minimizing average shortest path distances via shortcut edge addition". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 272–285 (cited on page 37).

Moreira-Matias, Luis, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas (2013). "Predicting taxi–passenger demand using streaming data".

In: *IEEE Transactions on Intelligent Transportation Systems* 14.3, pages 1393–1402 (cited on page 90).

Moulin, Hervé (2009). "Almost budget-balanced VCG mechanisms to assign multiple objects". In: *Journal of Economic theory* 144.1, pages 96–119 (cited on page 167).

Myerson, Roger B (1981). "Optimal auction design". In: *Mathematics of operations research* 6.1, pages 58–73 (cited on page 29).

Myerson, Roger B and Mark A Satterthwaite (1983). "Efficient mechanisms for bilateral trading". In: *Journal of economic theory* 29.2, pages 265–281 (cited on page 173).

Nash, J.F. (1951). "Non-cooperative Games". In: *Annals of Mathematics* 54.2, pages 286–295 (cited on page 25).

Nemhauser, George L., Laurence A. Wolsey, and Marshall L. Fisher (1978). "An analysis of approximations for maximizing submodular set functions–I". In: *Mathematical Programming* 14.1, pages 265–294 (cited on page 46).

Nguyen, Nhan-Tam, Anja Rey, Lisa Rey, Jörg Rothe, and Lena Schend (2016). "Altruistic hedonic games". In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 251–259 (cited on page 166).

Nisan, N., T. Roughgarden, E. Tardos, and V.V. Vazirani (2007). *Algorithmic Game Theory*. Cambridge University Press. URL: `https://books.google.nl/books?id=YCu2alSw0w8C` (cited on pages 11, 182).

Noonan, Sarah, Frank Haist, and Ralph-Axel Müller (Feb. 2009). "Aberrant functional connectivity in autism: Evidence from low-frequency BOLD signal fluctuations". In: *Brain Research* 1262, pages 48–63. DOI: `10.1016/j.brainres.2008.12.076` (cited on page 96).

NOS (Nov. 6, 2014). *Nederland is een aantrekkelijk belastingland*. NOS (cited on page 2).

— (Mar. 26, 2019). *Nederland is een belastingparadijs*. NOS (cited on page 2).

Olmstead, Larry (Apr. 1993). "2 Managers Held in Bidding Scheme at School Agency". In: *The New York Times*. URL: `https://www.nytimes.com/1993/04/21/nyregion/2-managers-held-in-bidding-scheme-at-school-agency.html` (cited on pages 5, 130).

Olsen, Martin and Anastasios Viglas (2014). "On the approximability of the link building problem". In: *Theoretical Computer Science* 518, pages 96–116 (cited on page 37).

OpenStreetMap contributors (2020). *Planet dump retrieved from https://planet.osm.org*. `https://www.openstreetmap.org` (cited on page 90).

Papagelis, Manos, Francesco Bonchi, and Aristides Gionis (2011). "Suggesting Ghost Edges for a Smaller World". In: *Proceedings of 20th ACM International Conference on Information and Knowledge Management*, pages 2305–2308 (cited on page 37).

Parotsidis, Nikos, Evaggelia Pitoura, and Panayiotis Tsaparas (2015). "Selecting Shortcuts for a Smaller World". In: *Proceedings of SIAM International Conference on Data Mining*. Chapter 4, pages 28–36 (cited on page 37).

Perumal, Senni, Prithwish Basu, and Ziyu Guan (2013). "Minimizing Eccentricity in Composite Networks via Constrained Edge Additions". In: *Military Communications Conference*, pages 1894–1899 (cited on page 37).

Pfeiffer, Joseph John and Jennifer Neville (2011). "Methods to determine node centrality and clustering in graphs with uncertain structure". In: *Fifth International AAAI Conference on Weblogs and Social Media*, pages 590–593 (cited on pages 63, 84, 96).

Piorkowski, Michal, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser (2009). *CRAWDAD Dataset epfl/mobility (v. 2009-02-24)*. `https://crawdad.org/epfl/mobility/20090224/cab` (cited on page 90).

Polak, Sven Carel (2014). *Algorithms for the Network Analysis of Bilateral Tax Treaties*. MSc. thesis, University of Amsterdam, The Netherlands (cited on pages 2, 7, 33).

Postema, Merel et al. (Dec. 2019). "Altered structural brain asymmetry in autism spectrum disorder in a study of 54 datasets". In: *Nature Communications* 10. DOI: `10.1038/s41467-019-13005-8` (cited on page 96).

Potamias, Michalis, Francesco Bonchi, Aristides Gionis, and George Kollios (2010). "K-nearest neighbors in uncertain graphs". In: *Proceedings of the VLDB Endowment* 3.1-2, pages 997–1008 (cited on pages 65, 66, 67).

Ptošek, Vit, Lukáš Rapant, and Jan Martinovič (2018). *Floating Car Data Collection for Processing and Benchmarking*. Version 1.0. URL: `https://doi.org/10.5281/zenodo.2250119` (cited on page 90).

Rahn, Mona and Guido Schäfer (2013). "Bounding the Inefficiency of Altruism through Social Contribution Games". In: *Web and Internet Economics*, pages 391–404 (cited on pages 103, 104, 166).

Riet, Maarten van 't and Arjan M. Lejour (2013). *Nederland belastingparadijs? Nederland doorsluisland!* CPB Policy Brief (cited on page 2).

— (2014). *Ranking the Stars: Network Analysis of Bilateral Tax Treaties*. CPB Discussion Paper (cited on page 59).

— (2018). "Optimal tax routing: Network analysis of FDI diversion". In: *International Tax and Public Finance* 25.5, pages 1321–1371 (cited on pages 2, 33, 59).

Riondato, Matteo and Evgenios M. Kornaropoulos (2016). "Fast approximation of betweenness centrality through sampling". In: *Data Mining and Knowledge Discovery* 30.2, pages 438–475 (cited on pages 66, 96, 97).

Roch, Sébastien, Gilles Savard, and Patrice Marcotte (2005). "An approximation algorithm for Stackelberg network pricing". In: *Networks* 46.1, pages 57–67 (cited on page 37).

Rothe, Jörg (2021). "Thou shalt love thy neighbor as thyself when thou playest: altruism in game theory". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Volume 35. 17, pages 15070–15077 (cited on page 166).

Roughgarden, Tim (2015). "Intrinsic Robustness of the Price of Anarchy". In: *Journal of the ACM* 62.5. DOI: `10.1145/2806883` (cited on pages 27, 28, 108, 133, 136).

— (2016). *Twenty lectures on algorithmic game theory*. Cambridge University Press (cited on page 26).

Roughgarden, Tim, Vasilis Syrgkanis, and Eva Tardos (2017). "The price of anarchy in auctions". In: *Journal of Artificial Intelligence Research* 59, pages 59–101 (cited on pages 102, 136).

Saha, Arkaprava, Ruben Brokkelkamp, Yllka Velaj, Arijit Khan, and Francesco Bonchi (2020). *Our Code and Data*. `https://github.com/ArkaSaha/MPSP-Centrality` (cited on page 87).

— (2021). "Shortest Paths and Centrality in Uncertain Networks". In: *Proceedings of the VLDB Endowment* 14.7, pages 1188–1201. DOI: `10.14778/3450980.3450988` (cited on page 8).

Supekar, Kaustubh, Lucina Q. Uddin, Amirah Khouzam, Jennifer Phillips, William D. Gaillard, Lauren E. Kenworthy, Benjamin E. Yerys, Chandan J. Vaidya, and Vinod Menon (2013). "Brain hyperconnectivity in children with autism and its links to social deficits". In: *Cell Reports* 5.3, pages 738–747. DOI: `https://doi.org/10.1016/j.celrep.2013.10.001` (cited on page 96).

Syrgkanis, Vasilis (2014). "Efficiency of mechanisms in complex markets". PhD thesis. Cornell University (cited on page 146).

Syrgkanis, Vasilis and Eva Tardos (2013). "Composable and efficient mechanisms". In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 211–220 (cited on pages 133, 136, 143).

Tang, Pingzhong and Tuomas Sandholm (2012). "Optimal auctions for spiteful bidders". In: *Twenty-Sixth AAAI Conference on Artificial Intelligence* (cited on page 166).

Tang, Youze, Xiaokui Xiao, and Yanchen Shi (2014). "Influence maximization: Near-optimal time complexity meets practical efficiency". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pages 75–86 (cited on page 66).

The Wall Street Journal (Aug. 1999). "State Governments Consider Reopening Bidding for Berlin Airport". In: *The Wall Street Journal*. URL: `https://www.wsj.com/articles/SB935028690392278172` (cited on pages 5, 129).

Tzourio-Mazoyer, Nathalie, Brigitte Landeau, Dimitri Papathanassiou, Fabrice Crivello, Olivier Etard, Nicolas Delcroix, Bernard Mazoyer, and Marc Joliot (2002). "Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain". In: *NeuroImage* 15.1, pages 273–289. DOI: `https://doi.org/10.1006/nimg.2001.0978` (cited on page 96).

Valiant, Leslie (Aug. 1979). "The Complexity of Enumeration and Reliability Problems". In: *SIAM Journal on Computing* 8.3, pages 410–421. DOI: `10.1137/0208032` (cited on pages 41, 65, 67, 68).

Vickrey, William (1961). "Counterspeculation, auctions, and competitive sealed tenders". In: *The Journal of finance* 16.1, pages 8–37 (cited on page 31).

Waard, P. de (Oct. 14, 2011). *Nederland belastingparadijs voor veel multinationals*. De Volkskrant (cited on page 2).

Wang, Chenxu and Ziyuan Lin (2019). "An efficient approximation of betweenness centrality for uncertain graphs". In: *IEEE Access* 7, pages 61259–61272 (cited on pages 63, 84, 96).

Williamson, David P. and David B. Shmoys (2011). *The Design of Approximation Algorithms*. Cambridge University Press (cited on pages 11, 22, 35).

Wu, Baoning and Brian D. Davison (2005). "Identifying link farm spam pages". In: *Proceedings of 14th International Conference on World Wide Web*, pages 820–829 (cited on page 37).

Yen, Jin Y. (1971). "Finding the k shortest loopless paths in a network". In: *Management Science* 17.11, pages 712–716 (cited on pages 70, 73).

Yuan, Ye, Lei Chen, and Guoren Wang (2010). "Efficiently answering probability threshold-based shortest path queries over uncertain graphs". In: *International Conference on Database Systems for Advanced Applications*, pages 155–170 (cited on pages 65, 66, 67).

Zhang, Long, Yuzhong Zhang, Donglei Du, and Qingguo Bai (2019). "Improved price of anarchy for machine scheduling games with coordination mechanisms". In: *Optimization Letters* 13.4, pages 949–959 (cited on pages 104, 109, 124, 125).

Zou, Lei, Peng Peng, and Dongyan Zhao (2011). "Top-K possible shortest path query over a large uncertain graph". In: *International Conference on Web Information Systems Engineering*, pages 72–86 (cited on pages 63, 64, 65, 66, 67, 70, 71, 75, 78, 87, 88, 100).

Samenvatting

De vijf hoofdstukken in deze thesis zijn verbonden door de volgende vraag:

Hoe dichtbij komt het?

In Hoofdstuk 3 bestuderen we twee prijsproblemen in netwerken. Gegeven is een gerichte graaf met kosten op de lijnen, een verzameling met grondstoffen en een speciaal punt u. Elke grondstof bestaat uit een stroom die getransporteerd moet worden van een beginpunt naar een eindpunt in het netwerk. De stroom gebruikt hiervoor het kortste pad en als er meerdere kortste paden zijn, verdeelt de stroom zich uniform over deze paden. Het speciale punt u kan de kosten van een aantal van zijn uitgaande lijnen aanpassen. Daarmee heeft het invloed op de kortste paden in het netwerk en indirect op hoe de stromen door het netwerk gaan. Het punt u is of geïnteresseerd in het maximaliseren van de stroom die door het punt u heen gaat, of in het maximaliseren van de inkomsten van zijn uitgaande lijnen. De inkomsten van u van een uitgaande lijn is de hoeveelheid stroom die door de lijn gaat, vermenigvuldigt met de kosten van de lijn. De inkomsten van u is de som van de inkomsten van zijn uitgaande lijnen. We bewijzen dat het probleem niet alleen NP-moeilijk is maar ook dat de optimale oplossing moeilijk is te benaderen voor beide optimalisatiedoelen. Ondanks deze negatieve resultaten voor het algemene probleem ontwikkelen we efficiënte optimale en benaderingsalgoritmen voor speciale gevallen. *Hoe dichtbij komt het?* We laten zien dat de benaderingsalgoritmen de best mogelijke benaderingsfactoren hebben, gegeven de negatieve resultaten.

In Hoofdstuk 4 kijken we naar het Meest Waarschijnlijk Kortste Pad (MWKP) in een onzeker netwerk. We modelleren een onzeker netwerk door aan iedere lijn een kans te geven waarmee die lijn bestaat. We laten zien dat het berekenen van de kans dat een gegeven pad het kortste pad is een #P-moeilijk probleem is. Om het MWKP te vinden, ontwikkelen we een Monte Carlo-achtig algoritme dat met grote waarschijnlijkheid snel het MWKP vindt. Gebaseerd op deze notie van een kortste pad definiëren we een nieuw soort betweenness centrality en we geven een gerandomiseerd algoritme dat hem snel kan berekenen. *Hoe dichtbij komt het?* Met hoge kans produceert ons algoritme het MWKP en we doen uitgebreide experimenten om de prestaties van het algoritme in de praktijk te testen.

In Hoofdstuk 5 bekijken we het roosteren van taken, elke met een verwerkingstijd, op een verzameling van machines, elk met een snelheid. De tijd die het duurt om een taak te verwerken op een machine is de verwerkingstijd van de taak gedeeld door de snelheid van de machine. Een simpel algoritme bekijkt de taken van klein naar groot en roostert iedere taak op de machine die zorgt voor de laagste doorlooptijd (nadat de korte taken al geroosterd zijn). De benaderingsfactor van dit algoritme en de prijs van anarchie van de speltheoretische variant, waarin iedere taak wordt beheerd door een speler die de doorlooptijd van zijn eigen taak wil minimaliseren, van dit probleem zijn hetzelfde. Het exact bepalen van de prijs van anarchie is een moeilijk probleem en vereist nieuwe technieken. *Hoe dichtbij komt het?* We beschrijven een techniek gebaseerd op een primal-dual methode die de beste bekende grens kan herbewijzen en de potentie heeft om tot sterkere grenzen te komen. We ontwikkelen ook instanties die een hogere ondergrens hebben voor een vast aantal taken en machines.

In Hoofdstuk 6 bestuderen we een gesloten eerste prijs veiling voor meerdere identieke objecten. Er is een corrupte veilingmeester die de winnende bieders benadert met het aanbod om hun bod te verlagen tot het hoogste verliezende bod. Ter compensatie verwacht hij een steekpenning die gelijk is aan een γ-fractie van de winst. We laten zien dat het bestuderen van deze veiling equivalent is aan het bestuderen van een γ-hybride veiling waarin de prijs een convexe combinatie is van het bod en het hoogste verliezende bod. Daarnaast bekijken we ook veilingen waarin de prijs die de bieders moeten betalen ten minste een γ-fractie is van de hoogste winnende biedingen. Voor deze veilingen bekijken we het verlies in het sociale welzijn dat wordt veroorzaakt door de corrupte veilingmeester. *Hoe dichtbij komt het?* We leiden matchende boven- en ondergrenzen op de grof gecorreleerde prijs van anarchie af. Vervolgens doen we de aanname dat spelers niet mogen overbieden en bewijzen meer grenzen op de prijs van anarchie voor verschillende equilibrium concepten en speciale versies van de veiling.

In Hoofdstuk 7 ontwerpen we waarheidsgetrouwe mechanismen voor spelers die (gedeeltelijk) altruïstisch zijn. We breiden het standaard nutsmodel uit met functies die de voorkeuren voor de andere spelers beschrijven. Door de uitbreiding kan het VCG mechanisme niet worden toegepast. We geven een karakterisatie van waarheidsgetrouwe mechanismen in het nieuwe model. Vervolgens beschrijven we een recept voor waarheidsgetrouwe mechanismen en gebruiken het om twee specifieke modellen met altruïsme te definiëren. Voor spelers met altruïstische motieven zijn er lagere prijzen nodig om ze de waarheid te laten vertellen. We laten onder andere zien dat met de altruïstische modellen het probleem van het financieren van een publiek project mogelijk is waar dit zonder altruïsme tegen obstakels oploopt. Verder kijken we naar wat er mogelijk is bij het herverdelen van de betalingen als de spelers altruïstische motieven hebben. *Hoe dichtbij komt het?* We laten zien dat mechanismen die geen rekening houden met altruïstische motieven kunnen worden getransformeerd in mechanismen die dat wel doen en er tegelijkertijd voor zorgen dat er minder geld de deelnemersgroep verlaat.

Summary

The five chapters in this thesis are connected by the following question:

How Close Does It Get?

In Chapter 3, we study two pricing problems in networks. We are given a directed graph with edge costs, a set of commodities, and a designated node u. Each commodity has a flow demand that needs to be transported from a source node to a destination node, and it uses shortest paths to do so. If there are multiple shortest paths, the flow splits uniformly. The node u can change the cost of at most a given number of its outgoing edges, which affects the shortest paths in the network and, indirectly, the flows. The node u is interested in either maximizing the flow that goes through it or maximizing the revenue it earns on the flow going through its outgoing edges. The revenue on an outgoing edge of u is equal to the amount of flow going through it times the cost of that edge. The revenue of u is the sum of the revenues of its outgoing edges. We prove that the problem is not only NP-hard but also highly inapproximable in general for both objectives. However, we develop efficient optimal and approximation algorithms for special cases. *How close does it get?* We show that the guarantees of the approximation algorithms are best possible given our inapproximability results.

In Chapter 4, we look into the notion of the Most Probable Shortest Path (MPSP) in an uncertain network. We model an uncertain network by assigning a probability to each edge with which that edge is available. We show that computing the probability that a path is the shortest path is #P-hard. To compute the MPSP we develop a sampling-based Monte Carlo-type algorithm to quickly find the MPSP. Based on this notion of shortest path, we also define

a new betweenness centrality measure and give a sampling-based algorithm for computing it. *How close does it get?* We show that with high probability, our algorithm returns the MPSP and we conduct extensive experiments to assess its performance in practice.

In Chapter 5, we consider the problem of related machine scheduling in which a set of jobs, each with a processing time, must be scheduled on a set of machines, each with a speed. The time it takes to process a job on a machine is the processing time of the job divided by the speed of the machine. The greedy algorithm that schedules the jobs from shortest to longest on the machine on which it completes earliest has an approximation guarantee equal to the price of anarchy of the game-theoretic version of this problem. In the game, each job is controlled by a player that is interested in minimizing the completion time of their own job. Finding a tight bound on the price of anarchy is a difficult problem and requires new techniques. *How close does it get?* We outline a technique based on a primal-dual method that is able to recover the best-known bound and has the potential to yield better bounds. Further, we provide better lower-bound instances for a fixed number of jobs and machines.

In Chapter 6, we study a first-price multi-unit auction in which a corrupt auctioneer approaches winning bidders with the offer to lower their bid to the highest losing bid in exchange for a bribe that is equal to a γ-fraction of the gains. We show that this auction is equivalent to a γ-hybrid auction in which the payments are a convex combination of first-price and second-price payments. We also consider a more general γ-approximate first-price auction where the payments recover at least a γ-fraction of the first-price payments. We study the social welfare loss that is caused by the corrupt auctioneer as a function of γ. *How close does it get?* We derive tight coarse correlated price of anarchy bounds if players are allowed to overbid. Then, we make a no-overbidding assumption and prove more (almost) tight bounds on the price of anarchy for various equilibrium concepts and specific versions of the auction.

In Chapter 7, we study the problem of designing truthful mechanisms for players that are (partially) altruistic. We model this by extending the standard utility model with other-regarding preferences. Unfortunately, VCG cannot be applied here anymore. We derive characterizations of truthful mechanisms in the new model, exploiting the specific form of the other-regarding preferences. Next, we give a recipe for truthful mechanisms and use it to define two specific models of altruism. Because of the altruistic dispositions, smaller payments are needed to incentivize participants to reveal their true preferences. We show that by using one of our altruistic models, the public project problem can be resolved for moderate altruism levels. Further, we look into redistributing the surplus among the participants in the presence of altruistic players. *How close does it get?* We show that mechanisms that do not take altruism into account can be transformed into ones that do and, in the process, make sure that less money leaves the group of participants.

Titles in the ILLC Dissertation Series:

ILLC DS-2016-01: **Ivano A. Ciardelli**
Questions in Logic

ILLC DS-2016-02: **Zoé Christoff**
Dynamic Logics of Networks: Information Flow and the Spread of Opinion

ILLC DS-2016-03: **Fleur Leonie Bouwer**
What do we need to hear a beat? The influence of attention, musical abilities, and accents on the perception of metrical rhythm

ILLC DS-2016-04: **Johannes Marti**
Interpreting Linguistic Behavior with Possible World Models

ILLC DS-2016-05: **Phong Lê**
Learning Vector Representations for Sentences - The Recursive Deep Learning Approach

ILLC DS-2016-06: **Gideon Maillette de Buy Wenniger**
Aligning the Foundations of Hierarchical Statistical Machine Translation

ILLC DS-2016-07: **Andreas van Cranenburgh**
Rich Statistical Parsing and Literary Language

ILLC DS-2016-08: **Florian Speelman**
Position-based Quantum Cryptography and Catalytic Computation

ILLC DS-2016-09: **Teresa Piovesan**
Quantum entanglement: insights via graph parameters and conic optimization

ILLC DS-2016-10: **Paula Henk**
Nonstandard Provability for Peano Arithmetic. A Modal Perspective

ILLC DS-2017-01: **Paolo Galeazzi**
Play Without Regret

ILLC DS-2017-02: **Riccardo Pinosio**
The Logic of Kant's Temporal Continuum

ILLC DS-2017-03: **Matthijs Westera**
Exhaustivity and intonation: a unified theory

ILLC DS-2017-04: **Giovanni Cinà**
Categories for the working modal logician

ILLC DS-2017-05: **Shane Noah Steinert-Threlkeld**
Communication and Computation: New Questions About Compositionality

ILLC DS-2017-06: **Peter Hawke**
The Problem of Epistemic Relevance

ILLC DS-2017-07: **Aybüke Özgün**
Evidence in Epistemic Logic: A Topological Perspective

ILLC DS-2017-08: **Raquel Garrido Alhama**
Computational Modelling of Artificial Language Learning: Retention, Recognition & Recurrence

ILLC DS-2017-09: **Miloš Stanojević**
Permutation Forests for Modeling Word Order in Machine Translation

ILLC DS-2018-01: **Berit Janssen**
Retained or Lost in Transmission? Analyzing and Predicting Stability in Dutch Folk Songs

ILLC DS-2018-02: **Hugo Huurdeman**
Supporting the Complex Dynamics of the Information Seeking Process

ILLC DS-2018-03: **Corina Koolen**
Reading beyond the female: The relationship between perception of author gender and literary quality

ILLC DS-2018-04: **Jelle Bruineberg**
Anticipating Affordances: Intentionality in self-organizing brain-body-environment systems

ILLC DS-2018-05: **Joachim Daiber**
Typologically Robust Statistical Machine Translation: Understanding and Exploiting Differences and Similarities Between Languages in Machine Translation

ILLC DS-2018-06: **Thomas Brochhagen**
Signaling under Uncertainty

ILLC DS-2018-07: **Julian Schlöder**
Assertion and Rejection

ILLC DS-2018-08: **Srinivasan Arunachalam**
Quantum Algorithms and Learning Theory

ILLC DS-2018-09: **Hugo de Holanda Cunha Nobrega**
Games for functions: Baire classes, Weihrauch degrees, transfinite computations, and ranks

ILLC DS-2018-10: **Chenwei Shi**
Reason to Believe

ILLC DS-2018-11: **Malvin Gattinger**
New Directions in Model Checking Dynamic Epistemic Logic

ILLC DS-2018-12: **Julia Ilin**
Filtration Revisited: Lattices of Stable Non-Classical Logics

ILLC DS-2018-13: **Jeroen Zuiddam**
Algebraic complexity, asymptotic spectra and entanglement polytopes

ILLC DS-2019-01: **Carlos Vaquero**
What Makes A Performer Unique? Idiosyncrasies and commonalities in expressive music performance

ILLC DS-2019-02: **Jort Bergfeld**
Quantum logics for expressing and proving the correctness of quantum programs

ILLC DS-2019-03: **András Gilyén**
Quantum Singular Value Transformation & Its Algorithmic Applications

ILLC DS-2019-04: **Lorenzo Galeotti**
The theory of the generalised real numbers and other topics in logic

ILLC DS-2019-05: **Nadine Theiler**
Taking a unified perspective: Resolutions and highlighting in the semantics of attitudes and particles

ILLC DS-2019-06: **Peter T.S. van der Gulik**
Considerations in Evolutionary Biochemistry

ILLC DS-2019-07: **Frederik Möllerström Lauridsen**
Cuts and Completions: Algebraic aspects of structural proof theory

ILLC DS-2020-01: **Mostafa Dehghani**
Learning with Imperfect Supervision for Language Understanding

ILLC DS-2020-02: **Koen Groenland**
Quantum protocols for few-qubit devices

ILLC DS-2020-03: **Jouke Witteveen**
Parameterized Analysis of Complexity

ILLC DS-2020-04: **Joran van Apeldoorn**
A Quantum View on Convex Optimization

ILLC DS-2020-05: **Tom Bannink**
Quantum and stochastic processes

ILLC DS-2020-06: **Dieuwke Hupkes**
Hierarchy and interpretability in neural models of language processing

ILLC DS-2020-07: **Ana Lucia Vargas Sandoval**
On the Path to the Truth: Logical & Computational Aspects of Learning

ILLC DS-2020-08: **Philip Schulz**
Latent Variable Models for Machine Translation and How to Learn Them

ILLC DS-2020-09: **Jasmijn Bastings**
A Tale of Two Sequences: Interpretable and Linguistically-Informed Deep Learning for Natural Language Processing

ILLC DS-2020-10: **Arnold Kochari**
Perceiving and communicating magnitudes: Behavioral and electrophysiological studies

ILLC DS-2020-11: **Marco Del Tredici**
Linguistic Variation in Online Communities: A Computational Perspective

ILLC DS-2020-12: **Bastiaan van der Weij**
Experienced listeners: Modeling the influence of long-term musical exposure on rhythm perception

ILLC DS-2020-13: **Thom van Gessel**
Questions in Context

ILLC DS-2020-14: **Gianluca Grilletti**
Questions & Quantification: A study of first order inquisitive logic

ILLC DS-2020-15: **Tom Schoonen**
Tales of Similarity and Imagination. A modest epistemology of possibility

ILLC DS-2020-16: **Ilaria Canavotto**
Where Responsibility Takes You: Logics of Agency, Counterfactuals and Norms

ILLC DS-2020-17: **Francesca Zaffora Blando**
Patterns and Probabilities: A Study in Algorithmic Randomness and Computable Learning

ILLC DS-2021-01: **Yfke Dulek**
Delegated and Distributed Quantum Computation

ILLC DS-2021-02: **Elbert J. Booij**
The Things Before Us: On What it Is to Be an Object

ILLC DS-2021-03: **Seyyed Hadi Hashemi**
Modeling Users Interacting with Smart Devices

ILLC DS-2021-04: **Sophie Arnoult**
Adjunction in Hierarchical Phrase-Based Translation

ILLC DS-2021-05: **Cian Guilfoyle Chartier**
A Pragmatic Defense of Logical Pluralism

ILLC DS-2021-06: **Zoi Terzopoulou**
Collective Decisions with Incomplete Individual Opinions

ILLC DS-2021-07: **Anthia Solaki**
Logical Models for Bounded Reasoners

ILLC DS-2021-08: **Michael Sejr Schlichtkrull**
Incorporating Structure into Neural Models for Language Processing

ILLC DS-2021-09: **Taichi Uemura**
Abstract and Concrete Type Theories

ILLC DS-2021-10: **Levin Hornischer**
Dynamical Systems via Domains: Toward a Unified Foundation of Symbolic and Non-symbolic Computation

ILLC DS-2021-11: **Sirin Botan**
Strategyproof Social Choice for Restricted Domains

ILLC DS-2021-12: **Michael Cohen**
Dynamic Introspection

ILLC DS-2021-13: **Dazhu Li**
Formal Threads in the Social Fabric: Studies in the Logical Dynamics of Multi-Agent Interaction

ILLC DS-2022-01: **Anna Bellomo**
Sums, Numbers and Infinity: Collections in Bolzano's Mathematics and Philosophy

ILLC DS-2022-02: **Jan Czajkowski**
Post-Quantum Security of Hash Functions

ILLC DS-2022-03: **Sonia Ramotowska**
Quantifying quantifier representations: Experimental studies, computational modeling, and individual differences

How Close Does It Get?

Ruben Brokkelkamp