Fakultät für Mathematik
Technische Universität München

Congestion Games: Equilibria, Networks, and Complexity

Clara Waldmann

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades einer

Doktorin der Naturwissenschaften (Dr. rer. nat.)

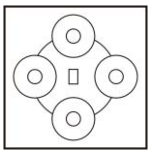
genehmigten Dissertation.

Vorsitzender:
Prof. Dr. Gregor Kemper

Prüfende der Dissertation:
1. Prof. Dr. Andreas S. Schulz
2. Prof. Dr. Max Klimm (Technische Universität Berlin)

Die Dissertation wurde am 06.12.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 24.03.2022 angenommen.

Für Opi



Abstract

This thesis studies pure Nash equilibria in atomic congestion games with increasing and decreasing resource cost functions. In the first part, we consider approximate equilibria in weighted congestion games with increasing resource cost functions, in particular polynomials with non-negative coefficients. We give the first super-constant lower bounds on the non-existence of approximate equilibria. For cost functions that are polynomials of degree d, we obtain a lower bound of $\Omega\Big(\frac{\sqrt{d}}{\ln d}\Big)$. For general cost functions, our lower bound depends on the number of players n and is lower bounded by $\Omega(\frac{n}{\ln n})$. By translating a Boolean circuit to an unweighted congestion game with polynomial cost functions, we are able to translate the lower bounds into hardness results. We show that it is NP-hard to decide whether a congestion game has an α-approximate equilibrium for any α below our non-existence bounds.

The second part studies the Price of Stability (PoS) in broadcast games with decreasing resource cost functions. We consider a generalization of fair cost allocation, where every edge has a constant total cost that is shared equally by all players. In our model, each edge has a concave increasing total cost that is shared equally by the players and the shares for each player are decreasing in the number of players. We give upper and lower bounds on the PoS. We explicitly compute a constant upper bound for the PoS in broadcast games with fair cost allocation. The same method shows constant bounds for quickly decreasing cost functions. For two special classes of cost functions, we give the first lower bounds. We conducted computational experiments enumerating small instances using methods from constraint programming. In the experiments, we found structures giving high lower bounds on the PoS. These instances are analyzed theoretically. We further study the complexity of computing good equilibria. We show that it is PLS-hard to compute an equilibrium for uniform network games. Computing the best equilibrium is shown to be NP-hard for multi- and broadcast games.

In the third part, we consider network games where no congestion is allowed and the players are restricted to use only shortest paths. This is the problem of finding disjoint shortest paths in undirected graphs. We give a polynomial-time algorithm for finding two disjoint shortest paths in graphs with non-negative edge lengths. This fills a gap in previous results by allowing zero length edges together with edges of positive length. As a subroutine, we give a polynomial-time algorithm for finding k disjoint paths in weakly acyclic mixed graphs containing directed and undirected edges.

Zusammenfassung

In der vorliegenden Arbeit untersuchen wir reine Nash Gleichgewichte in diskreten Auslastungsspielen (atomic congestion games) mit steigenden und fallenden Kostenfunktionen. Im ersten Teil geht es um approximative Gleichgewichte in gewichteten Spielen mit steigenden Kostenfunktionen, insbesondere Polynome mit nicht-negativen Koeffizienten. Wir zeigen erste untere Schranken für die Nicht-Existenz approximativer Gleichgewichte, die mit einem gewählten Parameter wachsen. Für Polynome vom Grad d zeigen wir eine Schranke von $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$. Für allgemeine Kostenfunktionen hängt die Schranke $\Omega(\frac{n}{\ln n})$ von der Spieleranzahl n ab. Wir zeigen, dass es für jedes α unter diesen Schranken NP-schwer ist, zu entscheiden, ob ein Auslastungsspiel ein α-approximatives Gleichgewicht besitzt. Dazu konstruieren wir aus einem booleschen Schaltkreis ein ungewichtetes Spiel mit polynomiellen Kostenfunktionen.

Im zweiten Teil der Arbeit geht es um den Price of Stability (PoS) in Broadcastspielen mit fallenden Kostenfunktionen. Unsere Kostenfunktionen sind eine Verallgemeinerung von gerechter Kostenverteilung, bei der jede Kante konstante Gesamtkosten hat, die gleichmäßig auf die Spieler aufgeteilt werden. In unserem Kostenmodell hat jede Kante konkav steigende Gesamtkosten, die gleichmäßig auf die Spieler aufgeteilt werden, wobei die Anteile für jeden Spieler mit der Spieleranzahl auf der Kante fallen. Wir zeigen obere und untere Schranken an den PoS. Insbesondere berechnen wir eine konkrete konstante obere Schranke für den PoS in Broadcastspielen mit gerechter Kostenverteilung. Mit der gleichen Methode erhalten wir konstante obere Schranken an den PoS für schnell fallende Kostenfunktionen. Weiterhin untersuchen wir untere Schranken an den PoS für zwei konkrete Klassen von Kostenfunktionen. Wir führen computergestütze Experimente durch, bei denen wir kleine Instanzen mit Methoden der Constraintprogrammierung aufzählen. Bei den Experimenten tauchen Strukturen auf, die große untere Schranken liefern. Diese Instanzen untersuchen wir theoretisch. Außerdem betrachten wir die Komplexität der Berechnung guter Gleichgewichte. Wir zeigen, dass es PLS-schwer ist, ein Gleichgewicht in gleichförmigen Netzwerkspielen zu berechnen. Die Berechnung des besten Gleichgewichts in Multi- und Broadcastspielen ist NP-schwer.

Der dritte Teil befasst sich mit Netzwerkspielen, in denen keine gemeinsame Nutzung von Resourcen erlaubt ist und die Spieler nur kürzeste Pfade verwenden dürfen. Wir untersuchen, ob es zulässige Profile gibt, also die Berechnung disjunkter kürzester Pfade in ungerichteten Graphen. Wir entwerfen einen polynomiellen Algorithmus, der das Problem für zwei Pfade in Graphen mit nicht-negativen Kantenlängen löst. Dieses Ergebnis schließt eine Lücke in der bisherigen Literatur, indem wir sowohl Kanten mit Länge null als auch Kanten mit positiver Länge erlauben. Als Unterprogramm zeigen wir einen polynomiellen Algorithmus für die Suche nach k disjunkten Pfaden in schwach azyklischen gemischten Graphen, die sowohl gerichtete als auch ungerichtete Kanten enthalten.

Contents

1 Introduction

Sharing resources with other participants is a common scenario in our daily lives. For example, the tenants of an apartment building share the bandwidth of their internet providers and the maintenance costs for the common rooms they use. Comparing these examples shows that sharing can have different effects on the participants. If many tenants have the same internet provider, they will experience *larger* latencies when trying to stream a movie simultaneously, compared to those who share their provider with only a few others. On the other hand, sharing the maintenance costs with more people results in a *smaller* share for each participant. Typically the tenants act selfishly and try to minimize their own latency and cost, which is, however, influenced by the choices of the other tenants.

The theoretical study of the interaction between selfish agents is called *game theory*. Situations where players non-cooperatively share common resources can be modeled by atomic **congestion games** introduced by Robert W. Rosenthal [Ros73]. A congestion game is given by a set of resources and a set of players. Every player has some strategies, which are subsets of the resources. Each resource is associated with a cost function, giving the cost for every player using it depending on the total load on the resource. Once all players chose one of their strategies, the cost for each player can be computed by summing the cost of the resources used by the player. The players want to minimize their individual costs by selecting appropriate strategies.

Figure 1.1: Choices of strategies of two players (A and B) in a congestion game with three resources ($\bigcirc, \square, \triangle$). a/b gives the cost of a resource. For one player the cost is a and for two players the cost is b for each of them.

Example 1. Figure 1.1 shows a congestion game with three resources $\bigcirc, \triangle$, and $\square$. There are two players A and B. A has two strategies $\{\bigcirc, \triangle\}$ and $\{\bigcirc, \square\}$. B has the two strategies $\{\triangle, \bigcirc\}$ and $\{\triangle, \square\}$. The resource costs are given in Figure 1.1 next to the resources. $\bigcirc$ incurs a cost of 1 for one player and a cost of 8 for two players. $\triangle$ has a cost of 1 for one player and a cost of 8 for two players. $\square$ has cost of 6 for one or player and a cost of 10 for two players. In Figure 1.1 A plays $\{\bigcirc, \square\}$ and B plays $\{\triangle, \square\}$. This incurs a cost of $1 + 10 = 11$ for both A and B. ⌟

There are two main classes of congestion games that differ in the behavior of the cost functions of the resources. Congestion games with **increasing** resource cost functions

model scenarios where sharing the same resource is disadvantageous, as in the example of using the cables of the same internet provider. On the other hand, congestion games with **decreasing** resource cost functions model scenarios where sharing resources is beneficial for the players, as in the example of sharing maintenance costs for common rooms.

A special case of increasing cost functions is the case where **no congestion** is allowed at all. Such games are important in cases where the resources can only be used by one player at a time. For example, the tenants of the apartment building may be free to choose their compartments in the cellar, but each compartment can only be used by one tenant. This thesis studies properties of congestion games with increasing (Chapter 3) and decreasing (Chapter 4) resource cost functions, and games where no congestion is allowed (Chapter 5).

Congestion games have been extensively studied in the literature. In particular, the concept of equilibria is investigated. An **equilibrium** is a profile (a choice of strategies for every player) where no player can decrease her cost by unilaterally deviating to another strategy.

Example 2. The profile in Example 1 is not an equilibrium. Player A can decrease her cost from 11 to 9 by deviating to her other strategy $\{\bigcirc, \triangle\}$. ⌟

This model of a stable state is called *pure Nash equilibrium* after John F. Nash. Pure refers to the fact that every player chooses one of her strategies, in contrast to mixed strategies, which are probability distributions over pure strategies. In this thesis, we only consider *pure* profiles. Nash [Nas50] shows that every finite game (with finitely many players and finite strategies) has a mixed equilibrium. However, the **existence** of pure equilibria is not guaranteed for all games.

For congestion games, Rosenthal [Ros73] shows that pure equilibria exist in all unweighted games. In an **unweighted** congestion game, the total load on a resource is just the number of players using it (as in Example 1). Hence, all players contribute the same to the load. The other case, where every player has an associated weight and the total load of a resource is the sum of the weights of the players using the resource, is called a **weighted** congestion game. There are examples of weighted congestion games that do not have pure equilibria. Thus, research has focused on the relaxed notion of **approximate equilibria**. An approximate equilibrium is a profile where no player can decrease her cost by some factor. If this factor is 1, an approximate equilibrium is a pure Nash equilibrium. On the other hand, as the factor increases, some profiles may become stable so that approximate equilibria exist.

Example 3. Recall that the profile from Example 1 is not a pure Nash equilibrium. However, it is an approximate equilibrium with factor 2, as none of the players can halve her cost by deviating to her other strategy. ⌟

In the first part of this thesis, we study how large the factor has to be to guarantee the existence of approximate equilibria in weighted congestion games with increasing resource cost functions.

In cases where equilibria are guaranteed to exist, one is interested in their *quality*. The most commonly used measure is the social cost of a profile, which is the sum of all

player costs. Although the players try to minimize their cost individually, this does not necessarily lead to a profile with overall minimal social cost.

Example 4. The profile in Example 1 has minimal social cost of $11 + 11 = 22$ in the game. But as observed in Example 2, it is not an equilibrium. ⌟

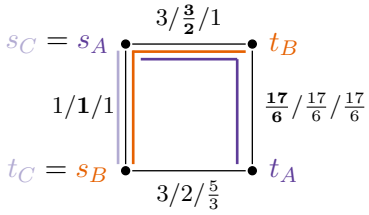
One of the research fields in algorithmic game theory investigates the gap between the social cost of equilibria and the minimal social cost achievable by any profile (not restricted to be an equilibrium). Two questions arise naturally: How large can the gap be in the worst case? How small can it be in the best case? The maximal ratio of the social cost of an equilibrium to the social cost of any profile is called the Price of Anarchy introduced by Koutsoupias and Papadimitriou [KP99] and Papadimitriou [Pap01]. It measures the worst-case gap. Schulz and Stier-Moses [SS03] and Anshelevich et al. [Ans+08a] introduce the counter-part called **Price of Stability**, which measures the best case, i.e., the minimal ratio of the social costs of an equilibrium and any profile.

Example 5. In Example 1, there are two equilibria. One is the profile $\sigma^*_{\max}$ where A plays $\{\bigcirc, \triangle\}$ and B plays $\{\triangle, \square\}$. Its social cost is $16 + 14 = 30$. The other is the profile $\sigma^*_{\min}$ where A plays $\{\bigcirc, \square\}$ and B plays $\{\triangle, \bigcirc\}$. The social cost is $14 + 9 = 23$. Recall from Example 4 that the minimal social cost in this game is 22. Hence, the Price of Anarchy is $\frac{30}{22}$ determined by $\sigma^*_{\max}$, and the Price of Stability in this game is given by $\sigma^*_{\min}$ and evaluates to $\frac{23}{22}$. ⌟

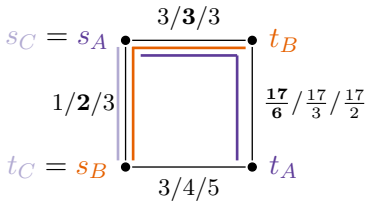
The Price of Anarchy of congestion games is fairly well understood, while for the Price of Stability much less is known. In particular, only one special case has been studied so far for congestion games with decreasing resource cost functions. This is the case of *fair cost allocation*, where every resource has a constant total cost that is shared equally by all players using the resource. In the second part of the thesis, we look at a more general class of decreasing cost functions, where each resource has a concave increasing total cost that is shared equally by all players, such that each share is decreasing in the number of players. Cost functions of this form arise in situations where **economies of scale** effects are active. In such situations, the total cost increases with the number of players, but the marginal increase gets smaller. For example, the supply of sweets for the common party room falls into this category. If more people come, we have to buy more sweets. However, our local sweets shop has a discount for selling large amounts. Thus buying one more package if we are already buying 10 packages will be cheaper than buying an additional package to just one package.

We study the Price of Stability in network games with decreasing cost functions with economies of scale effects as described above. **Network games** are special cases of congestion games where the resources are the edges of an underlying graph, and the strategies for each player are all paths between her source and sink node. Such games model, for example, routing traffic between different start and end locations in a road network.

Example 6. Figure 1.2 shows a network game with three players. The per-player cost is non-increasing (see Figure 1.2a) while the total costs on the edges are concave and non-decreasing (see Figure 1.2b). The shown profile is the one with minimal social cost



(a) The profile is not an equilibrium. Player A can decrease her cost from $4 + \frac{1}{3}$ to 4 by deviating to her other strategy.

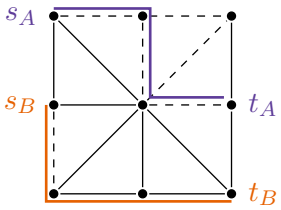


(b) The social cost of the profile is $7+\frac{5}{6}$.

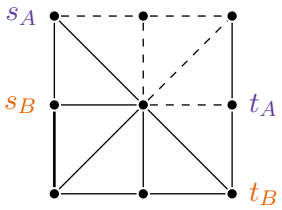
Figure 1.2: A network game with three players A, B, C. The respective source and sink nodes are labeled with s and t. The paths chosen in a strategy profile are highlighted with the respective colors. The labels on the edges give the resource cost functions, using the same notation as in Figure 1.1. In (a) the cost incurred to every player is given and in (b) the social cost is shown.

of $7 + \frac{5}{6}$. There is exactly one equilibrium which is the profile reached by letting A deviate to her other strategy. This equilibrium has social cost $3 + 3 + 3 = 9$. The Price of Stability is thus $\frac{9}{7+\frac{5}{6}} \approx 1.149$. ⌟

If we instead consider the *nodes* of the graph to be the resources and do not allow any congestion, then the players have to choose node disjoint paths. In the third part of the thesis, we further restrict the strategies to contain only the *shortest* paths between their source and sink node. In this setting, the concept of equilibria is not meaningful as it is not clear whether any valid profile exists. We thus ask: Are there node **disjoint shortest paths** (one for each player) in the network? See Figure 1.3 for two examples with a valid profile and without a valid profile.



(a) Two disjoint shortest paths.



(b) An instance without two disjoint shortest paths.

Figure 1.3: Network games where the resources are the nodes and no congestion is allowed. Further, the players can only use shortest paths to connect their source and sink. Solid edges have length 1 and dashed edges have length 0. In (a) there is a valid profile as shown with colors. In (b), where we changed the length of the thick edge from zero to one, there is no valid profile as both shortest paths use the center node.

1.1 Related Work

We give a very brief outline of the development of the questions studied in this thesis, focusing on atomic congestion games.

The origins of game theory as we know it today can be attributed to the influential book *Theory of Games and Economic Behavior* by von Neumann and Morgenstern [NM47], although the study of strategic behavior has been conducted before for specific economical settings. Most prominent are the works of Cournot [Cou38] and Pigou [Pig20]. Stable states have been one of the main concepts of consideration, right from the beginning. Cournot defines a notion of equilibria that is later (unknowingly) extended by Nash [Nas50] to more general games. His definition is what we now call a Nash equilibrium. Von Neumann [Neu28] shows that every zero-sum game has a mixed Nash equilibrium. In a mixed equilibrium players choose a probability distribution over their pure strategies. Nash [Nas50] extends the result and shows that all finite games have mixed equilibria. This is in contrast to the existence of pure Nash equilibria (equilibria). There are games (like matching pennies) that do not have pure equilibria. To overcome this issue, other solution concepts have been developed. The notion of α-approximate pure Nash equilibria (α-equilibria) is a natural relaxation. In an α-equilibrium, players only deviate if the decrease in their cost is substantial (measured by α). In the additive version of approximate equilibria, the difference of the costs is considered, while in the multiplicative version, the ratio of the costs is of interest. In this thesis, we use the multiplicative version. There are many other equilibria concepts that we do not discuss here. Examples are refinements of Nash equilibria like strong equilibria, or subgame perfect equilibria, and correlated equilibria as a generalization.

Another way of guaranteeing the existence of equilibria is to restrict the class of games under consideration. Rosenthal [Ros73] introduces the class of unweighted congestion games and shows that they always have equilibria. His proof uses a potential function with the property that the change in the potential for any single player deviation captures exactly the change in the corresponding player's cost. Later, Monderer and Shapley [MS96] define the class of potential games that are games that allow for a potential function as the one from Rosenthal. In the same work, Monderer and Shapley show that every potential game is isomorphic to an unweighted congestion game. Until today the potential function of Rosenthal is the primary tool to show the existence of equilibria.

In some applications, it may not be the case that all players have the same influence on the congestion of a resource. Instead, there could be some players who have a higher demand. Milchtaich [Mil96] extends the model of unweighted congestion games to weighted games. He gives examples of weighted congestion games that do not have equilibria. In weighted games, one has to specify how the total cost of a resource is distributed to the players. In the above paper, the total cost is shared proportionally by the ratio of the weight of the player to the total weight on the resource. This is also the model considered in this thesis. Kollias and Roughgarden [KR11; KR15] define another way to share the total cost called Shapley cost sharing. In this model, every player pays the average marginal increase she is responsible for over all player orderings. They show that with this cost sharing method, equilibria exist in any weighted congestion game.

There are many variations of congestion games, each useful for different applications. We mention only a few. Some are special cases where the structure of the strategies is restricted. Two examples are singleton games, where every strategy contains exactly one resource, and matroid games, where the strategies are the bases of a matroid over the set of resources. Another important class of games is the class of network games, where the strategies of a player are all paths in an underlying graph connecting her source and sink node. The study of selfish agents routing some flow in a network is much older than the definition of congestion games. Pigou [Pig20] already considers a network of two parallel links. The first use cases of network games are for modeling traffic flow in road networks. Wardrop [War52] and Braess [Bra68] study situations where players only control an infinitesimally small portion of a flow, such that a single player does not have any influence on the congestion. Today such models are referred to as non-atomic congestion games. A first version of the discrete (atomic) case of network games was considered by Koutsoupias and Papadimitriou [KP99], where the network consists of parallel links between the common source and sink node. As network games model many applications, they received a considerable amount of attention in the literature. In Chapter 4, we are looking exclusively at network games.

Other variations of congestion games consider different player functions than the sum of the cost of used resources. One example are bottleneck congestion games, where the cost of a player is the maximum of the cost of the resources used. Another generalization is the non-anonymous setting, where the resource cost depend not only on the total weight of the players using it but also on the specific set of players and their identities. The case of player-specific cost functions is similar to the above setting, where now every player has her own cost function for using a resource depending on the total weight.

Research in algorithmic game theory focuses mainly on three questions:

- Existence of equilibria.
- Complexity of computing an equilibrium if it exists.
- Quality of equilibria.

We briefly overview the current results on these three questions for (un)weighted congestion games with increasing and decreasing resource cost functions for exact and approximate equilibria. This is meant as a rough comparison of the state of the art for the three parts of this thesis. Detailed discussions of the related literature for the specific settings are deferred to the respective parts.

Research has focused on two special cases of cost functions. For increasing costs the case of polynomials of degree d with non-negative coefficients is often considered and for decreasing costs, the case of fair cost allocation, where every resource has a constant total cost that is shared equally by all players.

Existence of Equilibria We have already discussed the existence of exact equilibria. In unweighted congestion games such equilibria always exist both for increasing and decreasing cost functions, as shown by the potential function of Rosenthal [Ros73]. On the other hand, there are weighted congestion games that do not have equilibria. Fotakis, Kontogiannis, and Spirakis [FKS04; FKS05] give an example of a network game with

simple increasing cost functions. They show, however, that network games with linear cost functions always have equilibria. For the special case of fair cost allocation of decreasing cost functions, Chen and Roughgarden [CR06; CR09] give an example of a three-player undirected multicast game (where all players have the same source) without equilibria. Anshelevich et al. [Ans+04; Ans+08a] show that under the same cost model, weighted congestion games with two players and symmetric network games always have equilibria.

When the approximation factor α is large enough, every game has an α-equilibrium. Research thus focuses on finding the smallest approximation factor α such that α-equilibria exist in any game. For increasing cost functions, the most studied case is that of polynomial cost functions. Caragiannis and Fanelli [CF19; CF21] show that d-equilibria exist in any weighted game with polynomial cost functions of degree d. On the other hand, there are examples of such games without α-equilibria, where α is a small constant. We study the non-existence of approximate equilibria in weighted congestion games with polynomial cost functions in Chapter 3. For fair cost allocation Chen and Roughgarden [CR06] show that $\mathcal{O}(\log w_{\max})$-equilibria exist in any weighted directed network game, where $w_{\max}$ is the maximal weight of any player. In the undirected case Albers and Lenzner [AL10; AL13] show that any system optimum is a $H(n)$-equilibrium in multicast games, where $H(n)$ denotes the n-th harmonic number.

Computation of Equilibria Using the potential defined by Rosenthal [Ros73] puts the problem of finding an equilibrium in unweighted congestion games in the setting of polynomial local search problems in PLS as defined by Johnson, Papadimitriou, and Yannakakis [JPY88]. Hence, one algorithm to find an equilibrium is to let the players iteratively deviate as long as this decreases the potential. This is called the standard improving dynamics. Fabrikant, Papadimitriou, and Talwar [FPT04] show that finding an equilibrium is PLS-complete for games with increasing cost functions. This means that there are examples where the standard improving dynamics may take exponentially many steps before converging to a local minimum. Their completeness result holds even for multi-commodity network games and symmetric general games, where every player has the same set of available strategies. Ackermann, Röglin, and Vöcking [ARV06; ARV08] show that this further holds for undirected networks with linear cost functions. For decreasing cost functions Syrgkanis [Syr10] shows that finding an equilibrium is still PLS-complete in directed network games with fair cost allocation. Recently, Bilò et al. [Bil+15; Bil+21] extend this result to undirected network games. There are classes of congestion games where polynomial-time algorithms are known. Examples include symmetric network games for which Fabrikant, Papadimitriou, and Talwar [FPT04] show that an equilibrium can be found by a min-cost flow computation. Another case are matroid congestion games introduced by Ackermann, Röglin, and Vöcking [ARV08]. They show that the matroid property is the maximal property on the structure of the strategies such that the improving dynamics where the players play best-responses, converges in polynomial time. Interestingly, their result holds for any kind of cost functions (increasing, decreasing, non-monotone).

Even the problem of finding an approximate equilibrium in unweighted congestion games with increasing cost functions is PLS-complete for any approximation factor, as shown by Skopalik and Vöcking [SV08]. However, there are special cases where α-equilibria can be computed in polynomial time. Chien and Sinclair [CS07; CS11] show that the improving dynamics takes polynomially many steps for symmetric congestion games with increasing cost functions that do not grow too fast. If the cost functions are polynomials of degree d Giannakopoulos, Noarov, and Schulz [GNS21] show how to compute $d^{d+o(d)}$-equilibria in polynomial time. For polynomially decreasing cost functions, i.e., where the cost is of the form $\frac{c_r}{x^\beta}$ for a fixed $\beta > 0$ and a constant c_r, Bilò et al. [Bil+21] develop a distributed algorithm whose running time depends on the approximation factor, the constant β, and the number of players. If the approximation factor is chosen to be large enough, the running time of their algorithm is polynomial.

A closely related question is the decision of whether a given weighted congestion game has an (approximate) equilibrium. Dunkel and Schulz [DS06; DS08] show that the decision for exact equilibria is NP-hard for weighted network games with increasing cost functions. In Chapter 3, we show that this is also true for approximate equilibria in general congestion games with increasing cost functions.

Although the focus is on computing any equilibrium, other profiles can be considered. In particular, when looking at the quality of equilibria (see later), one is interested in an equilibrium of minimal social cost. As an approximation to such a best equilibrium the global minimizer of the potential of Rosenthal [Ros73] has been considered. Chekuri et al. [Che+06; Che+07] show that computing the global minimizer of the potential is NP-hard for singleton games with fair cost allocation. Syrgkanis [Syr10] remarks that their proof can be used to show that computing the best equilibrium is also NP-hard. For increasing cost functions, Sperber [Spe09; Spe10] shows that computing the best equilibrium is NP-hard for symmetric network games on a directed series-parallel graph with at least three players. In Chapter 5, we are interested in finding a profile where no congestion occurs.

Quality of Equilibria The most common measure of the quality of an equilibrium is its social cost, that is, the sum of all player costs. One is interested in the loss (in terms of the social cost) due to the selfish behavior of players, compared to a centrally enforced social optimum. Comparing the social cost of equilibria to a social minimum is first considered in Koutsoupias and Papadimitriou [KP99]. They study the worst ratio of the social costs of an equilibrium to a social optimum. This ratio is later called *Price of Anarchy (PoA)* by Papadimitriou [Pap01]. In contrast to this pessimistic measure of the behavior of selfish agents, Schulz and Stier-Moses [SS03] and Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, and Roughgarden [Ans+08a] consider the best ratio of the social costs of an equilibrium to a social optimum. The term *Price of Stability (PoS)* for this ratio is introduced by Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, and Roughgarden [Ans+08a].

Considering other profiles than equilibria in the above definitions yields new quality measures of games. Replacing equilibria by α-equilibria gives the approximate PoA and

PoS. Taking the set of strong equilibria results in the strong PoA introduced by Andelman, Feldman, and Mansour [AFM07; AFM09]. For the sequential PoA introduced by Leme, Syrgkanis, and Tardos [LST12], the set of subgame perfect equilibria is considered. Kawase and Makino [KM12; KM13] restrict the equilibria to have minimal potential in unweighted congestion games and call the corresponding prices potential-optimal. In this thesis, we only consider the standard PoA and PoS using the set of all equilibria.

The Price of Anarchy is well understood for congestion games with increasing cost functions. Roughgarden [Rou09; Rou12; Rou15] gives exact bounds on the PoA for unweighted congestion games. Bhawalkar, Gairing, and Roughgarden [BGR10; BGR14] later extended his approach to weighted congestion games. For decreasing functions, Anshelevich et al. [Ans+08a] show that for the special case of fair cost allocation the PoA is as large as the number of players, even for unweighted network games.

On the other hand, knowledge of the Price of Stability is not as deep. For increasing functions, mostly the case of polynomial cost functions has been studied. In unweighted games with polynomials of degree d, Christodoulou and Gairing [CG13; CG16] give exact values of the PoS depending on d. They show that asymptotically the PoS in such games is $\Theta(d)$. For weighted games, Christodoulou et al. [Chr+18; Chr+19] give lower bound examples that have a PoS that is exponential in the degree d. This almost (up to a factor of 2^{d+1}) settles the PoS for weighted congestion games with polynomial cost functions. To the best of our knowledge, for decreasing functions, only the special case of fair cost allocation in network games has been studied so far. For unweighted games in directed networks, Anshelevich et al. [Ans+08a] show that the PoS is $H(n)$ the n-th harmonic number, where n is the number of players. For weighted games, they show the asymptotics $\Theta(\log W)$, where W is the sum of all player weights. An almost matching lower bound of $\Omega\left(\frac{\log W}{\log \log W}\right)$ is given by Albers [Alb08; Alb09] for undirected network games. Subsequent research focuses on special cases of undirected network games. We follow this line of work in Chapter 4.

Recently, the study of approximate PoA and PoS has emerged. In particular, the trade-off between the approximation factor and the corresponding PoS (see for example [CR06; Chr+19; GP20]).

Increasing vs Decreasing Cost Functions We conclude this introduction with a small discussion on the different behaviors of congestion games with increasing and decreasing cost functions. In the case of increasing cost functions, the players try to avoid sharing resources and hence spread over the available resources. For decreasing functions, on the other hand, the players are concentrating on a small part of the resources to benefit from a high amount of sharing. These two underlying forces are structurally very different. This is the main reason why results for one of the two settings can not easily be transferred to the other setting. We have seen in the discussion of the literature above that progress in both settings is quite unequal, and that research is mostly separated. It is exceptionally interesting that some of the methods are nevertheless applicable to both settings, most prominently the potential function of Rosenthal [Ros73].

1.2 Structure of the Thesis

Firstly, we fix the notation used in this thesis and briefly introduce some of the main concepts formally in Chapter 2. The rest of the thesis consists of three main parts. A detailed introduction to the problems considered and an overview of the relevant literature are given in each part.

Chapter 3 In the first part, we consider *weighted* congestion games with *increasing* resource cost functions. We study the existence of *approximate* equilibria in such games with *polynomial* and *general* cost functions. We give super-constant lower bounds on the approximation factor, for which approximate equilibria do not necessarily exist. Our proofs are constructive, that is, we explicitly construct games without approximate equilibria. Further, we study the question of deciding whether a game has an approximate equilibrium. We show that this decision is NP-hard for every approximation factor below the lower bounds for the non-existence.
▷ This part is based on joint work with George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, and Diogo Poças presented at *ICALP 2020* [Chr+20]. A full version of the conference paper is accepted for publication at *Mathematics of Operations Research*.

Chapter 4 The second part deals with *unweighted network* games with *decreasing* resource cost functions. We generalize the model of fair cost allocation of Anshelevich et al. [Ans+08a] to resource cost functions that are decreasing for every player while the total cost of the resource (the sum of the per-player costs) is increasing and concave. We study the *quality* and *computation* of *exact* equilibria in such games. The contributions of this part are the extensions of several results for fair cost allocation to our more general class of cost functions.
▷ This part is based on unpublished joint work with Yiannis Giannakopoulos and Marcus Kaiser.

Chapter 5 The last part of the thesis investigates a subclass of *network* games where the resources are the nodes, *no congestion* is allowed, and the strategies of the players are restricted to be *shortest* paths in the network. We are interested in finding a feasible profile for the players, i.e., disjoint shortest paths connecting the respective sources and sinks. We show that this problem can be decided in polynomial time for two players in networks with *non-negative* edge lengths. The main contribution is to allow edges of length zero together with edges of positive length.
▷ This part is based on joint work with Marinus Gottschau and Marcus Kaiser published in *Operations Research Letters*, Volume 47, Issue 1, 2019 [GKW19].

Note on Layout The overall layout of this thesis follows the template from Andre Richter[1] with minor modifications. All graphics appearing in the figures are either self-drawn or taken and adapted from the public domain database of vector graphics Openclipart[2]. The colors used in the figures are taken from www.ColorBrewer2.org by Cynthia A. Brewer. We use the colorblind safe, print-friendly, and photocopy safe diverging palette with 4 colors[3] to make the thesis accessible to many people and in various forms (viewed on a screen, or printed in color or black and white).

[1] https://github.com/TUM-LIS/tum-dissertation-latex
[2] https://openclipart.org/
[3] https://colorbrewer2.org/#type=diverging&scheme=PuOr&n=4

2 Preliminaries

In this chapter we introduce terminology, notation and concepts used in this thesis. More specialized notation for single chapters is introduced there.

2.1 Basic Set Notation

We denote by $\mathbb{N}$ the set of natural numbers (including 0) and by $\mathbb{N}_{\geq a}$ the subset of natural numbers that are at least a. $\mathbb{Q}$ and $\mathbb{R}$ denote the sets of rational and real numbers.

For a set M we write 2^M for the *power set* of M containing all subsets of M. The set of subsets of size k is denoted by $\binom{M}{k}$. We write $|M|$ for the *cardinality* of M. The *symmetric difference* of two sets A and B is denoted by $A \triangle B$ and defined as $A \triangle B = (A \setminus B) \cup (B \setminus A)$.

A *binary relation* R on a set M is a set of pairs of elements of M, i.e., $R \subseteq M \times M$. We often use the infix notation $u\,R\,v$ for $(u, v) \in R$. A binary relation is *reflexive*, if $u\,R\,u$ for all elements $u \in M$. Two relations R and S on the same set M can be composed. We write $S \circ R = \{(u, w) \in M \times M | \exists v \in M : u\,R\,v \wedge v\,S\,w\}$ for the *composition* of R and S.

2.2 Congestion Games

The main concept of this thesis and the common subject of interest in all chapters are congestion games. We fix notation used in this thesis, following the standard notation from the literature.

A weighted (atomic) *congestion game* is given by a set of *resources* R and a set of *players* P. Every player $p \in P$ has a *weight* $w_p \in \mathbb{R}_{>0}$ and a set of available *strategies* $S_p \subseteq 2^R$ each consisting of a subset of the resources. If every player has weight 1, the game is called an *unweighted* congestion game. Each resource r has a *cost function* $c_r : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, giving the cost incurred to a player using r depending on the total weight of players using the resource. In particular, we assume $c_r(0) = 0$. If the cost functions are polynomials of maximum degree d with non-negative coefficients we say the game is a *polynomial congestion game*.

We distinguish two classes of congestion games by the type of the resource cost functions. If all of the cost functions are non-decreasing, we keep the name congestion game. If on the other hand, all cost functions are decreasing, we refer to such games as *cost sharing games*.

Given a congestion game we are interested in *strategy profiles*, which are vectors of strategies chosen by the players. We denote a profile by $\sigma \in \times_{p \in P} S_p$ and the strategy

chosen by player p in a profile σ by σ_p. Often we fix the strategies of all but one player, and denote by σ_{-p} the part of the profile σ excluding the strategy of player p. We obtain a full strategy profile by adding a strategy $s \in S_p$ for player p. This profile is written as (s, σ_{-p}) for any $s \in S_p$.

A profile σ induces a *congestion* $n_\sigma(r)$ on a resource r, which is the total weight of players using r. It is formally given by

$$n_\sigma(r) = \sum_{\substack{p \in P: \\ r \in \sigma_p}} w_p.$$

With this congestion the *cost of player* p in a profile σ is defined as

$$C_p(\sigma) = \sum_{r \in \sigma_p} c_r(n_\sigma(r)).$$

The sum of all player costs in a profile σ is called the *social cost* of σ and denoted by $C(\sigma)$. A profile with minimal social cost is a *social optimum*. It will be denoted by $\mathrm{OPT}(\mathcal{G})$ or just OPT if the instance $\mathcal{G}$ is clear from the context. Formally,

$$\mathrm{OPT}(\mathcal{G}) \in \operatorname*{arg\,min}_{\sigma \text{ profile in } \mathcal{G}} C(\sigma).$$

Other profiles of interest are profiles where no player has an incentive to unilaterally deviate. A profile σ is a *pure Nash equilibrium (equilibrium)*, if for all players $p \in P$ and all strategies $s \in S_p$ the inequality

$$C_p(\sigma) \leq C_p((s, \sigma_{-p})) \tag{2.1}$$

holds. Otherwise, if there is a player p with a strategy $s' \in S_p \setminus \{\sigma_p\}$ where

$$C_p(\sigma) > C_p((s', \sigma_{-p})), \tag{2.2}$$

we call s' an *improving move* for p in σ. In this thesis we only consider pure strategies and hence we refer to pure Nash equilibria just with equilibria.

The following process is called *improving-dynamics*: Begin with any profile σ. As long as there is a player with an improving move, let her deviate. If this process terminates, the final profile is an equilibrium by definition.

Since equilibria are not guaranteed to exist for weighted games, we study a relaxed version. A profile σ is an *α-approximate pure Nash equilibrium (α-equilibrium)* for some $\alpha \geq 1$, if for all players $p \in P$ and strategies $s \in S_p$ the inequality

$$C_p(\sigma) \leq \alpha \cdot C_p((s, \sigma_{-p})) \tag{2.3}$$

holds. Otherwise, a strategy $s' \in S_p \setminus \{\sigma_p\}$ is an *α-improving move* for p in σ, if

$$C_p(\sigma) > \alpha \cdot C_p((s', \sigma_{-p})). \tag{2.4}$$

Note that for $\alpha = 1$ these definitions correspond to the definitions of equilibria (compare to (2.1)) and improving move (compare to (2.2)). To distinguish this case clearly from $\alpha > 1$, we refer to a 1-equilibrium as *exact* equilibrium.

In Chapter 3, we study α-equilibria in weighted congestion games with increasing cost functions and in Chapter 4, we focus on exact equilibria in unweighted network games with decreasing cost functions.

Network Games A special case of congestion games are *network (congestion) games*. A network game is given by a graph $G = (V, E)$, that can be directed or undirected. For every player p there is a source node $s_p \in V$ and a sink node $t_p \in V$. The resources are the edges (or arcs) of the graph and the set of available strategies for a player p are all (directed or undirected) $s_p - t_p$ paths in G. (See Section 2.3 for graph definitions.) If all players have the same sink node, we call the network game a *multicast* (or single-sink) network game. To clearly distinguish this case from the general case where all players have their own source and sink node, we call the latter a *general* network game.

At the end of Chapter 3, we briefly consider network games on directed graphs. In Chapter 4, we focus on network games on undirected graphs. For the main part, we look at a subclass of multicast games. The topic of Chapter 5 can be seen as another special case of network games, where now the strategy set of every players is restricted to contain only shortest $s_p - t_p$ paths.

Potential Function Rosenthal [Ros73] introduces the following *potential* function for *unweighted* congestion games. For a strategy profile σ define the potential $\Phi(\sigma)$ as

$$\Phi(\sigma) = \sum_{r \in R} \sum_{i=1}^{n_\sigma(r)} c_r(i).$$

Note that for unweighted games the congestion $n_\sigma(r)$ is just the number of players using r in σ. Rosenthal shows that this is an exact potential. This means that for two strategy profiles differing only in a deviation of one player, the change in the potential is exactly the change in the cost of that player. For a profile σ in an unweighted congestion game and another profile $\sigma' = (s', \sigma_{-i})$ where player i changed her strategy to $s' \in S_i$, we have

$$\Phi(\sigma) - \Phi(\sigma') = C_i(\sigma) - C_i(\sigma').$$

The proof follows directly from the definitions of the player cost and the potential. Remarkably, this holds for both increasing and decreasing cost functions.

Using this key property of the potential, Rosenthal establishes the existence of equilibria in unweighted congestion games. Equilibria are exactly the local minimizers of the potential w.r.t. single player deviations. That is

$$\text{a profile } \sigma \text{ is an equilibrium } \iff \forall p \in P \forall s' \in S_p : \Phi(\sigma) \leq \Phi((s', \sigma_{-p})).$$

This also shows that the improving-dynamics in unweighted games always terminates. This potential is the main tool until today to show the existence of equilibria and to discuss the quality of equilibria. In Chapter 4, we follow the line of existing work and use Φ to derive upper bounds on the quality of equilibria in undirected network games.

Games having an exact potential are called potential games. Monderer and Shapley [MS96] show that the set of potential games and the set of unweighted congestion games are the same (up to isomorphism). This immediately creates the need to find similar concepts for *weighted* games. There has been progress on designing *approximate* potential functions for weighted games with the property, that local minimizers are approximate equilibria. In Section 3.6, we show that the social cost is an approximate potential for any weighted congestion game with non-decreasing cost functions.

Quality of Equilibria We are interested in the quality of equilibria in terms of the social cost. Equilibria are stable states in a system with selfish players, where every player is satisfied and does not want to deviate. However, from a centralized perspective they may not be optimal in terms of social cost. While a centralized authority wants to enforce a state which has smallest possible total cost, this may incur a high cost for some players.

Two quantities have been introduced in the literature to measure the difference of system optima and equilibria. The Price of Anarchy (PoA) measures how far apart the social cost of an equilibrium can be from a system optimum. The term was coined by Papadimitriou [Pap01] while the concept has been studied before by Koutsoupias and Papadimitriou [KP99]. We define the PoA of an instance $\mathcal{G}$ of an *unweighted* game as

$$\mathrm{PoA}(\mathcal{G}) = \max_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\mathrm{OPT}(\mathcal{G}))}.$$

We are interested in the highest possible PoA over a class of instances. We overload notation and define

$$\mathrm{PoA}(\mathcal{I}) = \sup_{\mathcal{G} \in \mathcal{I}} \mathrm{PoA}(\mathcal{G})$$

to be the PoA over a class $\mathcal{I}$ of instances of a game.

The PoA is very pessimistic as it measures the worst thing happening when introducing selfish players to a system. In contrast to that, the Price of Stability (PoS) has later been studied to consider the best thing that can happen when introducing selfish players. The term is introduced by Anshelevich et al. [Ans+08a] while the concept has been studied before by Schulz and Stier-Moses [SS03] and Anshelevich et al. [Ans+03]. The PoS of an instance $\mathcal{G}$ of an *unweighted* game is defined as

$$\mathrm{PoS}(\mathcal{G}) = \min_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\mathrm{OPT}(\mathcal{G}))}.$$

We refer to an equilibrium of minimal social cost in an instance as *the best equilibrium* if it is a global minimizer among all equilibria or *one of the best* otherwise.

Similar to the PoA we are interested in the PoS over a class of instances and write

$$\mathrm{PoS}(\mathcal{I}) = \sup_{\mathcal{G} \in \mathcal{I}} \mathrm{PoS}(\mathcal{G})$$

for the PoS over the class of instances $\mathcal{I}$.

In Chapter 4, we study network games parametrized by the number of players. For every $n \in \mathbb{N}$ we define $\mathcal{N}_n$ to be the set of network games with at most n players. For this class we abbreviate $\mathrm{PoA}(\mathcal{N}_n)$ by $\mathrm{PoA}(n)$ and $\mathrm{PoS}(\mathcal{N}_n)$ by $\mathrm{PoS}(n)$. When we are looking at subclasses of network games (like multicast, or broadcast games), we also use $\mathrm{PoA}(n)$ and $\mathrm{PoS}(n)$ to denote the PoA and PoS over the respective subclass.

2.3 Graphs

We use standard terminology and notation for undirected and directed graphs as can be found for example in [Die17].

Undirected Graphs An *undirected graph* $G = (V, E)$ consists of a (finite) set of *nodes* V and a set of *edges* $E \subseteq \binom{V}{2}$. Every edge e consists of two nodes, which we refer to as the *endpoints* of e. We say a node is *incident* to an edge, if it is one of the endpoints of the edge. Two edges are *incident* if they share a common endpoint. Further, two nodes u and v are *adjacent*, if $\{u, v\}$ is an edge in E. In this case, we also say that u is a *neighbor* of v. Since the edges are undirected, then also v is a neighbor of u.

The main relevant concept in graphs is that of a *path* connecting two nodes. A $u - v$ path is a sequence of nodes $(v_0, v_1, \ldots, v_k)$ such that all nodes are different, $v_0 = u$, $v_k = v$, and every pair of consecutive nodes are neighbors of each other, i.e., $\{v_{i-1}, v_i\} \in E$. We identify a $u - v$ path simultaneously by its sequence of nodes and by its sequence of edges. A sequence of nodes $(v_0, v_1, \ldots, v_k, v_0)$, where $(v_0, v_1, \ldots, v_k)$ is a path and $\{v_k, v_0\} \in E$, is called a *cycle*. Two nodes u and v are said to be *connected* in G, if there exists a $u - v$ path in G. A set of nodes is said to be connected, if every pair of nodes in the set is connected. If the whole set of nodes V is connected, we say the graph G is connected.

If every edge of the graph is associated with a length (or cost) $\ell(e) \in \mathbb{R}$, then the *length* of a path is the sum of the lengths of its edges. A *shortest* $u - v$ path in G w.r.t. ℓ is a $u - v$ path in G of minimal length induced by the edge lengths given by ℓ.

A path is a cycle-free way to connect two nodes. One generalization of paths, are cycle-free connections between a set of nodes. Those are called trees. A graph $G = (V, E)$ is a *tree*, if G is connected and does not contain any cycles. In a tree there is exactly one path between any pair of nodes. A graph (not necessarily connected) without any cycles is said to be *acyclic*.

Sometimes, we are only interested in a part of a graph $G = (V, E)$. A graph $G' = (V', E')$ containing only a subset of the nodes ($V' \subseteq V$) or a subset of the edges ($E' \subseteq E$) is called a *subgraph* of G. A special subgraph of a connected graph $G = (V, E)$ is a *spanning tree*. That is, a subgraph $G' = (V', E')$ that is a tree and contains all nodes of G, i.e., $V' = V$. Other types of subgraphs can be obtained by specifying the set of nodes. A subgraph $G' = (V', E')$ of $G = (V, E)$ is *induced* by V', if E' contains exactly those edges of E that have both endpoints in V'. We write $G[V']$ for the subgraph of G induced by the node set $V' \subseteq V$.

Directed Graphs In contrast to undirected graphs, where the edges are bidirectional, the edges in directed graphs are unidirectional. To clearly distinguish the case of bi- and unidirectional edges, we refer to the latter as *arcs*. A *directed graph* $G = (V, A)$ is given by a set of nodes V and a set of (directed) arcs $A \subseteq V \times V$. An arc $a = (u, v)$ is directed from u to v. We denote by $\text{tail}(a) = u$ the *tail* of a and by $\text{head}(a) = v$ the *head* of a. Similarly to before, two nodes u and v are neighbors, if there is an arc connecting both. From the direction of the arc, we get two types of neighbors. For an arc (u, v), u is an *in-neighbor* of v and v is an *out-neighbor* of u. We denote by $N_A^-(v) = \{u \in V : (u, v) \in A\}$ the set of all in-neighbors of v w.r.t. the arc set A. Similarly, $N_A^+(v) = \{w \in V : (v, w) \in A\}$ denotes the set of all out-neighbors of v w.r.t. A. Sometimes we need the in- and outgoing arcs for a node. We define $\delta_A^-(v) = \{a \in A : \text{head}(a) = v\}$ and $\delta_A^+(v) = \{a \in A : \text{tail}(a)\}$.

Most of the definitions for undirected graphs can be naturally adapted to directed graphs. A directed $u \to v$ path is a sequence of nodes $(v_0, v_1, \ldots, v_k)$ such that all nodes are different, $v_0 = u$, $v_k = v$, and there is an arc between every pair of consecutive nodes, i.e., $(v_{i-1}, v_i) \in A$. As before, we identify a directed path with its sequence of nodes and its sequence of arcs. A *directed cycle* is a sequence of nodes $(v_0, v_1, \ldots, v_k, v_0)$, where $(v_0, v_1, \ldots, v_k)$ is a directed path and $(v_k, v_0) \in A$.

As counter-part to undirected acyclic graphs, *directed acyclic graphs* (*dag* for short) are directed graphs without any directed cycles. A directed acyclic graph induces a *topological ordering* of its nodes, that is a linear order such that every arc goes from left to right, i.e., for an arc $(v, w) \in A$ it holds $v < w$ in the topological ordering. A linear ordering where all arcs go from right to left, i.e., for $(v, w) \in A$ we have $v > w$, is called a *reverse* topological ordering. There may be several topological orderings for a directed acyclic graph.

A *multigraph* is a (directed or undirected) graph where there can be multiple edges or arcs between the same nodes. The definitions of path and cycle naturally extend to multigraphs.

In Chapter 3, we only consider directed graphs and in Chapter 4, we mainly focus on undirected graphs. In Chapter 5, we look at both directed and undirected graphs and a mix of both: mixed graphs, that contain both undirected edges and directed arcs. The definitions and notation for mixed graphs are given in Chapter 5.

2.4 Running Time and Complexity Classes

We very briefly introduce the concept of running time and the complexity classes considered in this thesis. We do not give a full introduction to complexity theory. In particular, we assume the reader to be familiar with (non-)deterministic Turing machines. For the first parts, the descriptions used here and more details can be found in the book [AB09]. For local search problems, we refer to the original papers by Johnson, Papadimitriou, and Yannakakis [JPY88] and Schäffer and Yannakakis [SY91].

Running Time The running time of an algorithm measures how many elementary operations are needed until the algorithm stops. We are in particularly interested in the worst-case running time, that is, the largest number of operations needed for any input of size at most n. The size of the input is the number of bits needed to represent the necessary information. Often we are only interested in the asymptotic behavior of the running time and do not consider the exact number of operations.

We use the following notation for the asymptotic behavior of functions $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ for large arguments. We write $f \in \mathcal{O}(g)$, if there are constants $c \in \mathbb{R}_{>0}$ and $n_0 \in \mathbb{N}$ such that $\forall n \geq n_0 : f(n) \leq cg(n)$. This means that the asymptotic behavior of f is upper bounded by the asymptotic behavior of g, up to some constant factor. On the other hand, we introduce $f \in \Omega(g)$ for the reverse relation, that is, f is asymptotically lower bounded by g. Formally, $f \in \Omega(g)$, if there are constants $c \in \mathbb{R}_{>0}$ and $n_0 \in \mathbb{N}$ such that $\forall n \geq n_0 : f(n) \geq cg(n)$. If we have both $f \in \mathcal{O}(g)$ and $f \in \Omega(g)$, we write $f \in \Theta(g)$.

The *running time* of an algorithm is a function $f : \mathbb{N} \to \mathbb{N}$, where $f(n)$ gives the maximal number of elementary operations needed to stop for an input of size at most n. An algorithm runs in *polynomial time*, if $f \in \mathcal{O}(p)$, for some polynomial p.

Decision Problems Every YES/NO-question is a decision problem. Often they are stated in the form: given an instance of some problem, does there exist a solution with some desirable properties? For example, given a congestion game, does it have an equilibrium? Or, given a Boolean formula, does it have a satisfying assignment?

Both of the above examples are members of the class NP. This class contains exactly those decision problems for which it is easy to check whether some solution has the desired properties. For congestion games this means, given any strategy profile it can be efficiently checked whether this profile is an equilibrium. For a Boolean formula one can efficiently check whether some assignment of the variables is satisfying. Here easy and efficiently mean, by an algorithm (a Turing machine) that takes time polynomial in the size of the input instance. Formally, the class NP contains all languages that can be decided in polynomial time by a non-deterministic Turing machine. Languages that can be decided in polynomial time by deterministic Turing machines are collected in the class P. We say, an instance of a decision problem is a YES-instance, if and only if the answer to the corresponding question is "Yes". Otherwise, the instance is a NO-instance. For example the Boolean formula $x \wedge y$ is a YES-instance to the satisfiability problem, since there is a satisfying assignment (set both variables x and y to TRUE).

We give two examples of problems in NP, that will be used later in the thesis.

Example 7. CIRCUIT SATISFIABILITY is the following decision problem: Given a Boolean circuit with n input bits and one output bit, is there an assignment for the input bits such that the circuit outputs 1?

The size of the instance is the number of input bits plus the number of gates in the circuit. As for Boolean formulas, we observe that the problem is indeed in NP. Given an assignment of the input bits, we can efficiently check whether the circuit outputs 1 by just evaluating the circuit. ⌟

Example 8. An instance of *Exact-Cover by 3-Sets (X3C)* is given by a ground set X and a family $\mathcal{S}$ of 3-element subsets of X. The question is, whether there exists an exact cover of X, i.e., is there a $\mathcal{C} \subseteq \mathcal{S}$ such that every element of X appears in exactly one of the sets in $\mathcal{C}$?

The size of an instance is the number of elements in X plus the size of $\mathcal{S}$. Given a subset $\mathcal{C}$ of $\mathcal{S}$ we can efficiently check for every element of X, whether it is contained in exactly one of the sets in $\mathcal{C}$. Hence, X3C is in NP. ⌟

Some problems in NP seem to be more difficult than others. Karp [Kar72] introduced a notion of reducibility among problems in NP. A *polynomial-time (Karp) reduction* from a problem $\mathcal{P} \in$ NP to another problem $\mathcal{Q} \in$ NP consists of a polynomial-time computable function φ, that maps instances I of $\mathcal{P}$ to instances of $\mathcal{Q}$ such that I is a YES-instance for $\mathcal{P}$ if and only if $\varphi(I)$ is a YES-instance for $\mathcal{Q}$. A problem is said to be *NP-complete*, if it is a member of NP and there is a polynomial-time reduction from any other problem in NP to the complete problem. If there is a reduction from any problem

in NP to some problem $\mathcal{P}$, but that problem is not necessarily a member of NP, then $\mathcal{P}$ is said to be *NP-hard*.

Both example problems Circuit Satisfiability [Pap94] and X3C [GJ90] are NP-complete. We use Circuit Satisfiability in Section 3.4 to show NP-hardness of several questions related to the existence of approximate equilibria in polynomial congestion games. In Section 4.7.3, X3C is used as starting point in reductions showing that computing special equilibria in multi- and broadcast games is NP-hard.

Counting Problems Another type of problems that arise in some applications is the question of *how many* solutions do exist? Problems of this type are collected in the class #P (sharp P) defined by Valiant [Val79]. Informally, a function $f : \{0,1\}^* \to \mathbb{N}$ is in #P if $f(x)$ counts all accepting paths of a polynomial-time nondeterministic Turing machine for input x. A function f is #P-complete, if it is a member of #P and any other function in #P can be computed by a polynomial-time Turing machine with an oracle for f. We refer to the original paper [Val79] and the book [AB09] for the details.

Many counting versions of NP-completes problems give rise to #P-complete problems. In particular, if a (Karp) reduction from a decision problem A to a decision problem B is *parsimonious*, that is, it maintains the number of solutions (it induces a one-to-one mapping between the solution sets), then the #P-hardness of A can immediately be transferred to B.

Example 9. The counting version of Circuit Satisfiability is #P-complete as can be seen by observing that the Cook-Levin reduction gives a one-to-one mapping from certificates to satisfying assignments. ⌟

Interestingly, there are also #P-complete problems, whose corresponding decision problems are easy. Valiant [Val79] shows that counting the number of perfect matchings in a bipartite graph (or equivalently, computing the permanent of a matrix) is #P-complete. But deciding whether a bipartite graph has a perfect matching can be done efficiently for example by a computation of a determinant.

We use the counting version of Circuit Satisfiability in Section 3.5 to show #P-hardness of counting the number of α-equilibria in weighted polynomial congestion games by giving a parsimonious reduction from Circuit Satisfiability.

Local Search Problems In contrast to decision problems, where the question is whether there *exists* some feasible solution, we are often in the situation where several feasible solutions exist, and we want to find a *locally optimal* one. To study the complexity of local search algorithms Johnson, Papadimitriou, and Yannakakis [JPY88] introduce the class of *polynomial-time local search problems (PLS)*. The key property of a problem in PLS is that there is a polynomial-time algorithm to decide if a solution is a local optimum and if not, give a better solution in the neighborhood. This immediately suggests a standard algorithm to solve such local search problems. Start with any feasible solution and do local improvements until we reach a local optimum. Each step can be done in polynomial time, but the question remains how many steps this algorithm may have to take before terminating. Johnson, Papadimitriou, and Yannakakis [JPY88]

show that there is a problem where this standard algorithm takes exponentially many steps in the worst case. They further define a reduction among problems ins PLS and give a complete problem for the class PLS. Later Schäffer and Yannakakis [SY91] show PLS-completeness of a variety of local search problems. Their reductions preserve the property that the standard algorithm may take exponentially many steps.

We give the formal definitions of PLS and PLS-reduction from [JPY88] and [SY91]. An instance I of a *local search problem* $\mathcal{P}$ is associated with a set of feasible solutions $\mathcal{F}(I)$. For every feasible solution S there is a set $\mathcal{N}(S, I) \subseteq \mathcal{F}(I)$ of feasible solutions which are called *neighbors* of S. Further, every feasible solution S has a measure $\mu(S, I) \in \mathbb{R}_{\geq 0}$. A solution S is a *local optimum*, if there is no better solution in its neighborhood. If μ is to be maximized that means S is a local optimum, if $\mu(S', I) \leq \mu(S, I)$ for every $S' \in \mathcal{N}(S, I)$ in the neighborhood of S. If μ is to be minimized the inequality should hold in the other direction. The task is to find a local optimum for a given instance.

A local search problem $\mathcal{P}$ is in PLS, if all of the following hold.

(i) For every instance I, some feasible solution in $\mathcal{F}(I)$ can be computed in polynomial time.

(ii) The measure of a feasible solution can be computed in polynomial time.

(iii) For every feasible solution S, it can be decided in polynomial time whether S is a local optimum, and if not, a better solution in $\mathcal{N}(S, I)$ can be computed in polynomial time.

Example 10. Max Cut([SY91]) is a problem in PLS. An instance of Max Cut is given by an undirected graph (V, E) with positive edge weights $w \in \mathbb{R}_{>0}^{E}$. A feasible solution is a partition of the nodes $V = V_1 \dot{\cup} V_2$. The measure is the sum of the weights of edges crossing the cut

$$\mu(V_1, V_2) = \sum_{e \in E \cap (V_1 \times V_2)} w(e)$$

which is to be maximized. The neighbors of a partition consists of all partitions reached by moving a single node from one side of the partition to the other side.

We can compute some feasible solution by just setting $V_1 = V$ and $V_2 = \emptyset$. The measure of a partition can be computed in polynomial time. Since we can compute the measure of all neighbors by looking at every vertex individually, also the third point is satisfied. ⌟

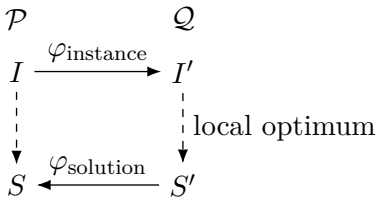


Figure 2.1: Schematic drawing of a PLS-reduction from $\mathcal{P}$ to $\mathcal{Q}$. An instance I of $\mathcal{P}$ is transformed to an instance I' of $\mathcal{Q}$ by the function $\varphi_{\text{instance}}$ and any local optimum of I' is mapped to a local optimum of I by $\varphi_{\text{solution}}$.

A *PLS-reduction* from a problem $\mathcal{P} \in \text{PLS}$ to another problem $\mathcal{Q} \in \text{PLS}$ consists of two polynomial-time computable functions $\varphi_{\text{instance}}$ and $\varphi_{\text{solution}}$. Where $\varphi_{\text{instance}}$ maps instances of $\mathcal{P}$ to instances of $\mathcal{Q}$ and $\varphi_{\text{solution}}$ maps solutions of $\varphi_{\text{instance}}(I)$ to solutions of I. The main property is that both functions maintain local optima. If $S_{\mathcal{Q}}$ is a local optimum of $\varphi_{\text{instance}}(I)$, then $\varphi_{\text{solution}}(S_{\mathcal{Q}}, I)$ is a local optimum of I. Note that the reverse direction need not hold, i.e., not every local optimum of I has to appear as a local optimum of $\varphi_{\text{instance}}(I)$. Figure 2.1 shows a schematic drawing of a PLS-reduction.

As for the class NP, there are problems that are *complete* for PLS. That is they are member of PLS, and there is a PLS-reduction from any other problem in PLS to the complete problem. Schäffer and Yannakakis [SY91] show that Max Cut is PLS-complete. We use this problem in Section 4.7.2 as starting point of a PLS-reduction to show that finding an equilibrium in a class of network games is PLS-complete.

2.5 Modeling Languages and Solution Methods

We briefly introduce the two languages used in this thesis to model optimization and decision problems. For our theoretical results we use linear programs and the main tool of duality. For the computational experiments we use satisfiability modulo theories (SMT). Notation and details for linear programs can be found in [Sch00]. For a detailed description of SMT and SMT solvers we refer to [KS16].

Linear Programming A *linear program (LP)* is an optimization problem of a linear function over a set of points defined by linear inequalities. The canonical form of an LP is

$$\max \quad c^\top x \tag{2.5}$$
$$\text{s.t.} \quad Ax \leq b \tag{2.6}$$
$$x \geq 0, \tag{2.7}$$

where $x \in \mathbb{R}^n$ are the variables, $c \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$ are vectors, and $A \in \mathbb{R}^{(m \times n)}$ is a coefficient matrix. The linear function $x \mapsto c^\top x$ is called the *objective function* and for a vector x the value $c^\top x$ is the *objective (function) value* of x. The linear constraints (2.6) and (2.7) define a polyhedron $\{x \in \mathbb{R}^n : Ax \leq b \wedge x \geq 0\}$. This polyhedron is called the *feasible region* of the LP and a point x in this polyhedron is a *feasible solution* of the LP. If the feasible region is empty, the LP is said to be infeasible. A feasible solution with maximal objective function value is called an *optimal solution* and the maximal objective function value is called the *optimal value* of the LP. If the objective function is unbounded on the feasible region, the LP is called *unbounded*. Otherwise, the LP is *bounded*. A bounded LP can have several optimal solutions but there is always at least one optimal solution that is a vertex of the feasible region.

To find the optimal value of a bounded LP, one can thus compute the objective function value for all vertices of the feasible region and take the maximum. A vertex x of an n-dimensional feasible region is specified by n linearly independent constraints

that are *active* at x, i.e., x satisfies them with equality. Thus, a vertex is the solution of a system of linear equations derived from n of the inequality constraints defining the feasible region. In the context of linear programming, the vertices of the feasible region are called *basic feasible solutions*.

Duality. If one is only interested in bounding the value of an LP, one can make use of duality. The following two linear programs are called *dual* to each other:

$$\begin{aligned} \max \quad & c^\top x && \text{(primal)} \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned} \qquad\qquad \begin{aligned} \min \quad & b^\top y && \text{(dual)} \\ \text{s.t.} \quad & A^\top y \geq c \\ & y \geq 0. \end{aligned}$$

The dual of an LP has a variable for every inequality in the primal, thus $y \in \mathbb{R}^m$. *Strong duality* shows that the optimal values of the primal and the dual are the same, if both feasible regions are non-empty. To bound the optimal value of the primal, it suffices to show that both feasible regions are non-empty, i.e., by giving feasible solutions. Then, the dual objective function value of any dual feasible solution is an upper bound on the optimal value of the primal and thus an upper bound on the objective function value for any primal feasible solution. This is called *weak duality* of linear programs. More details can be found in the book [Sch00].

In Section 4.6, we determine the optimal value of an LP by looking at all basic feasible solutions. In Section 4.5.2, we use weak duality to show one of the key inequalities. We set up an LP and give feasible solutions to the primal and dual such that the corresponding objective function values are the left- and right-hand side of the inequality.

Algorithms. There are several algorithms to solve LPs. The most used algorithm in practice is the Simplex algorithm invented by Dantzig in 1947 (see [Dan90]). This algorithm follows the approach of looking at the basic feasible solutions of the LP. It goes from vertex to vertex of the feasible region as long as the objective function value increases. Since there are only finitely many vertices, this process terminates. But there are examples [KM72] where the number of vertices visited by the Simplex algorithm is exponential in the input size. The Simplex algorithm is thus not efficient in the worst-case. However, in practice it works very well. In contrast to that, there are polynomial-time algorithms to solve LPs, that are not useful in practice. The extension of the ellipsoid method for non-linear programs (where the constraints or the objective function are non-linear) to linear programs by Khachiyan [Kha80] is the first.

Satisfiability Modulo Theories The constraints of an LP are a conjunction of several linear inequalities. For some applications it is useful to extend this model by allowing more Boolean combinations of linear inequalities like disjunction or implication. For example, one might have to model a constraint of the form "If $x \leq 3$, then $y + z \geq 5$ or $y \leq 7$". Such combinations of a Boolean structure on top of an underlying theory can be expressed with the language of *satisfiability modulo theories (SMT)*. In this thesis, we consider only quantifier-free formulas over the linear fragment of the theory of reals

(QF_LRA)[1]. Such formulas model Boolean combinations of linear inequalities of real variables as in the example above. For a detailed introduction to SMT we refer to the book [KS16].

In contrast to optimizing some objective function as in LPs, here the question is whether there is an assignment for the variables such that the formula is satisfied. One can, however, approximate the optimal value of an LP by solving several satisfiability problems. Each time we add a new constraint of the form "objective function value > previous value". We use this method in Section 4.6.5 to lower bound the optimal value of an LP with additional Boolean combinations of linear inequalities.

SMT Solvers. State-of-the-Art SMT solvers combine solvers for checking the satisfiability of Boolean formulas (SAT solvers) with decision procedures for conjunctions in the underlying theory. In our case of QF_LRA, the latter are feasibility questions of LPs, i.e., the existence of a point satisfying a set of linear inequalities. The framework of such SMT solvers is *DPLL modulo theories (DPLL(T))* introduced by Tinelli [Tin02]. It is an extension of the DPLL framework for SAT named after its inventors Davis, Putnam, Loveland, and Logemann. The key idea is to use a SAT solver for the outer Boolean formula where the inner inequalities are abstractly represented as Boolean variables. If the SAT solver finds a satisfying assignment for those Boolean variables, the corresponding inequalities (whose assignment is set to TRUE) are handed to the feasibility checker. Using a variant of the Simplex algorithm known as the general Simplex algorithm, the checker tries to find an assignment for the real variables satisfying all of the given inequalities. If there is an assignment, the formula is satisfiable and an assignment has been found. Otherwise, a new satisfying assignment for the outer Boolean variables has to be found. To prevent the same assignment from appearing again, the negation of the current conjunction is added as new clause. Several improvements on this procedure have been developed by exchanging more information between the SAT solver and the feasibility checker. In our computational experiments in Section 4.6.5, we use the SMT solver OpenSMT2 [Hyv+16]. It is the winner of the 15th International Satisfiability Modulo Theories Competition (SMT-COMP 2020)[2] in the Single Query Track of the QF_LRA division.

2.6 Weighted Harmonic Mean

In Section 4.6, we use as tool the weighted harmonic mean of some values. We introduce the notation and some properties used later. The *weighted harmonic mean* of the *elements* $r_1, \dots, r_n \in \mathbb{R}_{>0}$ with *weights* $w_1, \dots, w_n \in \mathbb{R}_{\geq 0}$ is defined as

$$\mathcal{H}((w_1, r_1), \dots, (w_n, r_n)) = \frac{w_1 + \dots + w_n}{\frac{w_1}{r_1} + \dots + \frac{w_n}{r_n}}.$$

We observe the usual properties of a mean.

[1] see https://smtlib.cs.uiowa.edu/logics.shtml
[2] https://smt-comp.github.io/2020/

Observation 1.

(i) The mean is between the minimal and maximal element, irregardless of the chosen weights:

$$\min\{r_1, \dots, r_n\} \leq \mathcal{H}((w_1, r_1), \dots, (w_n, r_n)) \leq \max\{r_1, \dots, r_n\}.$$

(ii) The mean of n elements can be recursively computed by computing the means of two elements:

$$\mathcal{H}((w_1, r_1), (w_2, r_2), (w_3, r_3)) = \mathcal{H}((w_1, r_1), (w_2 + w_3, \mathcal{H}((w_2, r_2), (w_3, r_3)))).$$

We show two monotonicity properties of the weighted harmonic mean. First, we show that decreasing the weight of the smaller element or increasing the weight and the value of the larger element increases the weighted harmonic mean.

Lemma 1. *For elements $r_1, r_2, r_2' \in \mathbb{R}_{\geq 0}$ with $r_1 \leq r_2 \leq r_2'$ and weights $w_1, w_1', w_2, w_2' \in \mathbb{R}_{\geq 0}$ with $w_1 \geq w_1'$ and $w_2 \leq w_2'$ the weighted harmonic means satisfy*

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}((w_1', r_1), (w_2, r_2)) \tag{2.8}$$

and

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}((w_1, r_1), (w_2', r_2')). \tag{2.9}$$

Proof. We have

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}((w_1', r_1), (w_2, r_2)) \iff w_2\left(\frac{1}{r_1} - \frac{1}{r_2}\right)(w_1' - w_1) \leq 0.$$

As $w_2 \geq 0$ we look at the remaining factors on the right hand side. Since $r_1 \leq r_2$ the second factor is non-negative and as $w_1 \geq w_1'$ the third factor is negative. Thus, the right hand side is satisfied and (2.8) holds.

For (2.9), we similarly compute

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}((w_1, r_1), (w_2', r_2')) \impliedby w_2'\left(\frac{1}{r_2'} - \frac{1}{r_2}\right)(w_1 + w_2') \leq 0,$$

where we used $w_2 \leq w_2'$. As $r_2 \leq r_2'$ the right hand side is satisfied and (2.9) holds. □

The next lemma shows other conditions under which the mean increases. They arise in the computations done in Section 4.6.

Lemma 2. *For elements $r_1, r_1', r_2 \in \mathbb{R}_{\geq 0}$ with*

$$r_1 \leq r_2 \qquad \text{and} \qquad r_1 \leq r_1'$$

and weights $w_1, w_1', w_2, w_2' \in \mathbb{R}_{\geq 0}$ where

$$w_2' = x w_2 \qquad \text{and} \qquad w_1' \leq x w_1$$

for some $x \in \mathbb{R}_{\geq 0}$, the weighted harmonic means satisfy

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}((w_1', r_1'), (w_2', r_2))$$

Proof. From the definition of the weighted harmonic mean, we see that multiplying the weights with the same factor does not change the mean

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) = \mathcal{H}((xw_1, r_1), (xw_2, r_2)).$$

With (2.8) we get the relation

$$\mathcal{H}((xw_1, r_1), (xw_2, r_2)) \leq \mathcal{H}((w_1', r_1), (w_2', r_2)).$$

From the definition we also have that increasing one of the elements increases the mean, which finally gives the inequality from the statement

$$\mathcal{H}((w_1, r_1), (w_2, r_2)) \leq \mathcal{H}\big((w_1', r_1), (w_2', r_2)\big) \leq \mathcal{H}((w_1', r_1'), (w_2', r_2)).$$

□

3 Existence and Complexity of Approximate Equilibria in Weighted Congestion Games

3.1 Introduction

We begin with the situation where using the same resource is disadvantageous for the players.

Consider a group of mathematicians and a group of computer scientists who want to offer some cookies to their visitors. Each group needs small cookies to be served with a cup of tea and large cookies to be served on their own. There are 4 local bakeries that produce cookies of different sizes, shapes, and flavors. In Figure 3.1, we see that bakery 1 produces small, round, classic chocolate chip cookies. Bakery 2 produces large, round, plain shortbread. Bakery 3 produces small, rectangular, plain shortbread. Finally, bakery 4 produces large, rectangular, classic chocolate chip cookies. Both of the groups

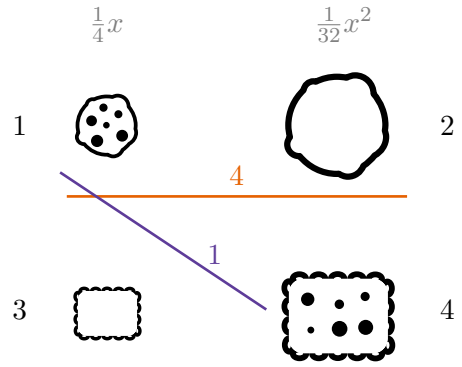


Figure 3.1: Four bakeries producing cookies of different size (small or large), shape (round or rectangular), and flavor (with chocolate chunks or plain shortbread). The production times depending on the number of packages x is given at the top in gray. A choice of the computer scientists is shown in orange. They need 4 packages of each size. A choice of the mathematicians is shown in purple. They only need 1 package of each size.

have different criteria for choosing the combination of their small and large cookies. The computer scientists prefer to have the same shape for both versions. So they either order the round cookies at bakeries 1 and 2 or the rectangular ones from bakeries 3 and 4. The mathematicians, on the other hand, prefer to have the same type of cookies. They either order the classic chocolate chip cookies from bakeries 1 and 4 or the shortbread from bakeries 2 and 3.

All of the bakeries follow the same procedure of handling their orders. They first collect the orders, that is, the number of cookies to be produced, then they bake the cookies, and finally (once *all* of the cookies are done) they are packaged and delivered to the customers. The time between placing an order and getting the cookies thus depends on the *total* number of cookies produced by the respective bakery. All bakeries sell packages of 100 cookies. The bakeries for the small cookies take $\frac{1}{4}$ unit of time per package to be produced. While the bakeries for the large cookies take $\frac{1}{32}$ unit of time for the square of the number of packages to be produced (see Figure 3.1).

As there are more computer scientists, they need 400 cookies (4 packages) of both sizes and the mathematicians only need 100 (1 package). Now the question arises, where should both groups place their orders to get the cookies as quickly as possible?

Figure 3.1 shows one choice of the two groups: the computer scientists (orange) chose the round shape and the mathematicians (purple) chose the chocolate chip cookies. In this scenario, the waiting time for the mathematicians is $\frac{1}{4}(4+1) + \frac{1}{32} = \frac{41}{32}$. If they would instead go for the shortbread, assuming the computer scientists still want the round cookies, the waiting time would be $\frac{1}{4} + \frac{1}{32}(4+1)^2 = \frac{33}{32}$. Hence, they would save some time by switching to ordering shortbread. Computing the waiting time of the computer scientists in the resulting scenario shows that they now would save some time by changing their order. These considerations can be continued, and one observes that in each of the 4 possible choices of the two groups, one group can save some time by changing their order. Thus, there is no stable scenario where none of the groups can decrease their waiting time.

However, both mathematicians and computer scientists are lazy people. They are only willing to change their order if the waiting time decreases by at least one-half of the time they are currently experiencing. In the scenario shown in Figure 3.1, the mathematicians are too lazy to change their order, and the computer scientists can not decrease their waiting time by changing their order. Hence, this is a lazy stable state.

We have seen that stable states do not necessarily exist, but stable states are obtained once the players are lazy enough. In this chapter, we study the following questions. How lazy do the players have to be to obtain stable states? Given a situation as above and a value of the laziness, can one decide whether there is a lazy stable state?

The above situation can be modeled as a *weighted congestion game* as introduced in Section 2.2, where the resources are the bakeries, the players are the two groups with the number of needed packages being their weight, and the strategies are the choices of the bakeries as explained above. The costs incurred to the players are the waiting times. The stable states in this game are the pure Nash equilibria (equilibria). We have seen in the example in Figure 3.1 that there are weighted congestion games that do not have equilibria. Hence, a natural approach is to relax the condition and study α-approximate pure Nash equilibria (α-equilibria). These are states where none of the players can decrease her cost by a factor of $\alpha \geq 1$ (the laziness). Notice that for $\alpha = 1$ the 1-equilibria are the same as the equilibria. To distinguish this case, we refer to 1-equilibria as *exact* equilibria. If the factor α is large enough, then α-equilibria do exist in any game. We study how large α has to be such that α-equilibria are guaranteed to

exist, and how hard it is to check for a specific game and a given factor α, whether the game has an α-equilibrium.

Note on Collaboration This chapter is based on joint work with George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, and Diogo Poças. The results were presented at *ICALP 2020* [Chr+20]. A full version of the conference paper is accepted for publication at *Mathematics of Operations Research*. The presentation of this chapter follows closely the one in the full version. Here we add a small discussion on transferring our results to network games.

Previous Work We give a brief overview of the literature related to the existence of exact and approximate equilibria in weighted congestion games. As in the rest of this chapter, we focus on polynomial congestion games.

Exact Equilibria. As mentioned already in the introduction of this thesis, Rosenthal [Ros73] showed that *unweighted* congestion games always have exact equilibria.

On the other hand, for *weighted* congestion games, there are examples that do not have any equilibria. Libman and Orda [LO97; LO01], Fotakis, Kontogiannis, and Spirakis [FKS04; FKS05], and Goemans, Mirrokni, and Vetta [GMV05] give instances of network games with two players of weights 1 and 2 that do not have equilibria. Goemans, Mirrokni, and Vetta use polynomial cost functions of degree 2.

On the positive side, there are special cases of weighted games where equilibria are guaranteed to exist. Fotakis, Kontogiannis, and Spirakis [FKS04; FKS05] show that network games with *linear* cost functions always have equilibria. Equilibria also exist for *exponential* cost functions, as shown by Panagopoulou and Spirakis [PS07], Harks, Klimm, and Möhring [HKM09; HKM11], and Harks and Klimm [HK10; HK12].

Besides restricting the cost functions of the game, one can also consider special structures of the strategies to obtain the existence of equilibria. Fabrikant, Papadimitriou, and Talwar [FPT04] and Fotakis, Kontogiannis, Koutsoupias, Mavronicolas, and Spirakis [Fot+02; Fot+09] show that *singleton* games (where every strategy consists of exactly one resource) possess equilibria. The same holds for games with the matroid property (where strategies consist of bases of a matroid over the resources), as shown by Ackermann, Röglin, and Vöcking [ARV06; ARV08].

Dunkel and Schulz [DS06; DS08] show that *deciding* whether an equilibrium exists is NP-complete, even for weighted symmetric network games with step cost functions. They extend the non-existence instance of Fotakis, Kontogiannis, and Spirakis [FKS04; FKS05] to a gadget that is used in a reduction from 3-Partition ([GJ90]).

Approximate Equilibria. The currently largest known lower bound on the non-existence of approximate equilibria in weighted *polynomial* congestion games is given by Hansknecht, Klimm, and Skopalik [HKS14]. They construct two-player polynomial congestion games of degree 4 that do not have α-equilibria where $\alpha \approx 1.153$.

On the positive side, Caragiannis, Fanelli, Gravin, and Skopalik [Car+11] show that polynomial congestion games of degree d always have $d!$-equilibria. This factor was later

improved to $d+1$ by Hansknecht, Klimm, and Skopalik [HKS14] and finally to d by Caragiannis and Fanelli [CF19; CF21].

To the best of our knowledge, there are no known lower bounds on the existence of approximate equilibria in weighted congestion games with *general* cost functions.

Our Results

Lower Bounds on the Non-Existence. We construct weighted congestion games with polynomial and general cost functions that do not have α-equilibria, where α is lower bounded by a super-constant.

For *polynomial* cost functions of degree d, we give games that do not have α-equilibria, for any $\alpha < \alpha(d)$, where $\alpha(d)$ grows as $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$ (Section 3.3, Theorem 1). In contrast to the previous constant lower bound of $\alpha \approx 1.153$, we had to use a construction where the number of players grows as a function of d.

For *general* cost functions, we study the non-existence of α-equilibria depending on the number of players n in Section 3.6. We build games with n players that do not have α-equilibria, for any $\alpha < \alpha(n)$, where $\alpha(n)$ grows as $\Omega(\frac{n}{\ln n})$ (Theorem 5). Our cost functions are step functions with a single breakpoint. We almost match this lower bound by showing that n player games always have n-equilibria (Theorem 4). Thus, one cannot derive super-constant lower bounds for the non-existence with instances with a fixed number of players (as for example studied by Hansknecht, Klimm, and Skopalik [HKS14]).

Hardness Construction. In Section 3.4, we create a polynomial unweighted congestion game from a Boolean circuit, such that in any α-equilibrium, the players emulate the computation of the circuit, for any $\alpha < 3^{d/2}$. This allows us to transfer hardness results from Circuit Satisfiability to the hardness of deciding the existence of α-equilibria. Inspired by similar results of Conitzer and Sandholm [CS08] for *mixed* equilibria, we show NP-hardness of deciding the existence of *pure* α-equilibria with additional properties (Theorem 2), such as: Is there an α-equilibrium where a specific resource is used by at least one player? Our hardness results hold for any $\alpha < 3^{d/2}$. An interesting property of our construction is that the set of α-equilibria and equilibria are the same. Hence, our gadget is *gap-introducing*.

NP-Hardness Results. We use the hardness gadget together with the non-existence gadgets to show that the decision versions of the existence of α-equilibria are NP-hard, where α is bounded by the bounds from the non-existence games.

In particular, for *polynomial* cost functions, both gadgets are combined in a *black-box* way. This immediately gives NP-hardness of the decision version for any polynomial congestion game without α-equilibria (Theorem 3). In particular, with the non-existence games from Section 3.3, we obtain that it is NP-hard to decide whether a congestion game with polynomials of degree d has an α-equilibrium for any $\alpha < \alpha(d)$. Further, our reduction is parsimonious, showing #P-hardness of counting different α-equilibria (Corollary 2).

For *general* cost functions, a similar combination of both gadgets is shown in Theorem 6. Thus, it is NP-hard to decide the existence of α-equilibria in n player games, where $\alpha < \alpha(n)$.

We want to point out that in the above decision problems, α is *not* part of the input. Further, for polynomial cost functions, we assume the degree d to be fixed and not part of the problem's input. In contrast to that, for general cost functions, the number of players n is part of the problem's input.

3.2 Model and Notation

We study congestion games as introduced in Section 2.2. For the main part of this chapter the resource cost functions are *polynomials* of degree $d \in \mathbb{N}$ with non-negative coefficients.

Approximate Equilibria Recall the definition of α-equilibria from Section 2.2. A profile σ is an α-equilibrium, if for all players $p \in P$ and all strategies $s \in S_p$ the inequality

$$C_p(\sigma) \leq \alpha \cdot C_p((s, \sigma_{-p}))$$

holds. Otherwise, there is a player p with a strategy $s' \in S_p \setminus \{\sigma_p\}$ for which

$$C_p(\sigma) > \alpha \cdot C_p((s', \sigma_{-p})).$$

Such a strategy s' is called an *α-improving move* for p in σ.

Observe that if a game has an α-equilibrium it also has a β-equilibrium for any $\beta > \alpha$. A game does *not* have an α-equilibrium, if for every strategy profile there is some player who has an α-improving move. Similarly to before, if a game does not have α-equilibria, it also does not have β-equilibria for any $\beta < \alpha$.

Example 11. Recall the example from the introduction (Figure 3.1). The given profile σ is a 2-equilibrium, but the game does not have any exact equilibria. In fact the game has α-equilibria for any $\alpha > \frac{57}{56}$ but for no smaller α. ⌟

Given a partial profile σ_{-p}, a strategy $s \in S_p$ is *α-dominating* for p, if it is an α-improving move for any other strategy of p. That is

$$\forall s' \in S_p \setminus \{s\} : C_p((s', \sigma_{-p})) > \alpha \cdot C_p((s, \sigma_{-p})).$$

Observe that a profile σ where the current strategy σ_p of every player is α-dominating, is an α-equilibrium and even an exact equilibrium.

Computational Complexity In this chapter, we study questions of the form: Given a game G, does G have an α-equilibrium? We are interested in the complexity of deciding this question for fixed values of α. We make the following assumptions on the games appearing as input:

- All players have *rational* weights.
- The set of available strategies for every player is given *explicitly*. (In contrast to congestion games with implicitly given strategies, like network games.)
- For polynomial resource cost functions all coefficients are *rationals*, and for step functions the values and breakpoints are *rationals*.

Our two main results show NP-hardness of deciding the aforementioned question for any real α below a given bound. This is independent on α being a rational or an irrational. To show that the problem lies in NP, we have to be able to check in polynomial time whether a strategy profile is an α-equilibrium. We say a real number α is *polynomial-time computable*, if the upper Dedekind cut $\{q \in \mathbb{Q} : q > \alpha\}$ is decidable in polynomial time. As in particular all rationals are polynomial-time computable, we have NP-completeness of the problem for any rational α below the given bound. We extend this notion of polynomial-time computability to sequences of reals $\alpha : \mathbb{N} \to \mathbb{R}$. Such a sequence is polynomial-time computable if $\{(n, q) \in \mathbb{N} \times \mathbb{Q} : q > \alpha(n)\}$ is decidable in polynomial time.

3.3 The Non-Existence Gadget

In this section we construct a family of weighted polynomial congestion games of degree d that do not have $\alpha(d)$-equilibria, where $\alpha(d)$ grows as $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$.

For a fixed degree $d \geq 2$, the game $\mathcal{G}^d_{(n,k,w,a)}$ is given by the parameters

$$n \in \mathbb{N}, \quad k \in \{1, \ldots, d\}, \quad w \in [0, 1], \quad \text{and} \quad a \in [0, 1].$$

There is a *heavy player* of weight 1 and n *light players* $1, \ldots, n$ all of weight w. There are $2(n + 1)$ resources $r_0, r_1, \ldots, r_n$ and $r'_0, r'_1, \ldots, r'_n$. The cost function for r_0 and r'_0 is

$$c_0(x) = x^k$$

and for all other resources the cost is

$$c_1(x) = ax^d.$$

Every player has exactly two strategies. The heavy player either plays all r resources or all r' resources:

$$S_{\text{heavy}} = \{\{r_0, \ldots, r_n\}, \{r'_0, \ldots, r'_n\}\}.$$

A light player i plays either r_0 or r'_0 together with her own resource of the other type:

$$S_i = \{\{r_0, r'_i\}, \{r'_0, r_i\}\}.$$

Figure 3.2 shows the structure of the strategies of $\mathcal{G}^d_{(n,k,w,a)}$.

This game is highly symmetric in the strategies and the cost functions of the resources. Hence, there are only two truly different situations: either the heavy player uses r_0 on her own, or she shares this resource with some of the light players. To show that the

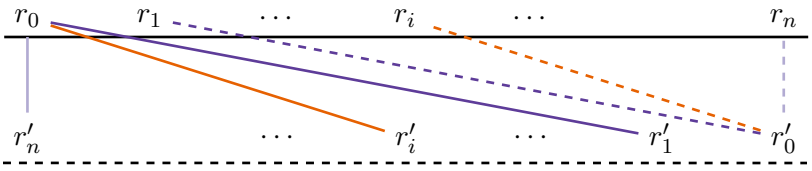


Figure 3.2: The strategies of $\mathcal{G}^d_{(n,k,w,a)}$. The heavy player plays either all r resources (solid black) or all r' resource (dashed black). The two strategies (solid and dashed) of light players $1, i$, and n are shown in purple, orange, and light purple, respectively.

game does not have an α-equilibrium, we show that in both situations some player has an α-improving move. Since we want to find a lower bound on α such that there is an improving move, we look at specific moves and the changes in the player costs. Choosing α to be less than those cost ratios, guarantees that the moves are α-improving. If we then maximize those values over all games, we get a bound $\alpha(d)$ for which there are games not having α-equilibria for any $\alpha < \alpha(d)$. Finally, we show the asymptotic lower bound of $\alpha(d)$ by choosing specific values for the parameters.

Now consider a game $\mathcal{G}^d_{(n,k,w,a)}$ and a strategy profile where the heavy player is alone on resource r_0. Then every light player i is playing $\{r'_0, r_i\}$. Hence the current cost of the heavy player is $c_0(1) + nc_1(1+w)$. Switching to her other strategy would incur a cost of $c_0(1+nw) + nc_1(1)$. Thus, the improvement factor is

$$\frac{c_0(1) + nc_1(1+w)}{c_0(1+nw) + nc_1(1)} = \frac{1 + na(1+w)^d}{(1+nw)^k + na}. \tag{3.1}$$

In the other situation, where the heavy player shares r_0 with some light player i, the current cost of i is at least $c_0(1+w) + c_1(w)$. Switching to her other strategy would incur a cost of at most $c_0(nw) + c_1(1+w)$. Thus, the improvement factor can be lower bounded by

$$\frac{c_0(1+w) + c_1(w)}{c_0(nw) + c_1(1+w)} = \frac{(1+w)^k + aw^d}{(nw)^k + a(1+w)^d}. \tag{3.2}$$

We define

$$\alpha_d(n,k,w,a) = \min\left\{ \frac{1 + na(1+w)^d}{(1+nw)^k + na}, \frac{(1+w)^k + aw^d}{(nw)^k + a(1+w)^d} \right\}.$$

Then we have from the above computations that the game $\mathcal{G}^d_{(n,k,w,a)}$ does not have an α-equilibrium for any $\alpha < \alpha_d(n,k,w,a)$. Let

$$\alpha(d) = \sup\{\alpha_d(n,k,w,a) : n \in \mathbb{N}, k \in \{1,\dots,d\}, w \in [0,1], a \in [0,1]\}. \tag{3.3}$$

We have shown the following theorem.

Theorem 1. *For any $d \in \mathbb{N}_{\geq 2}$ there exist weighted polynomial congestion games of degree d that do not have α-equilibria for any $\alpha < \alpha(d)$.*

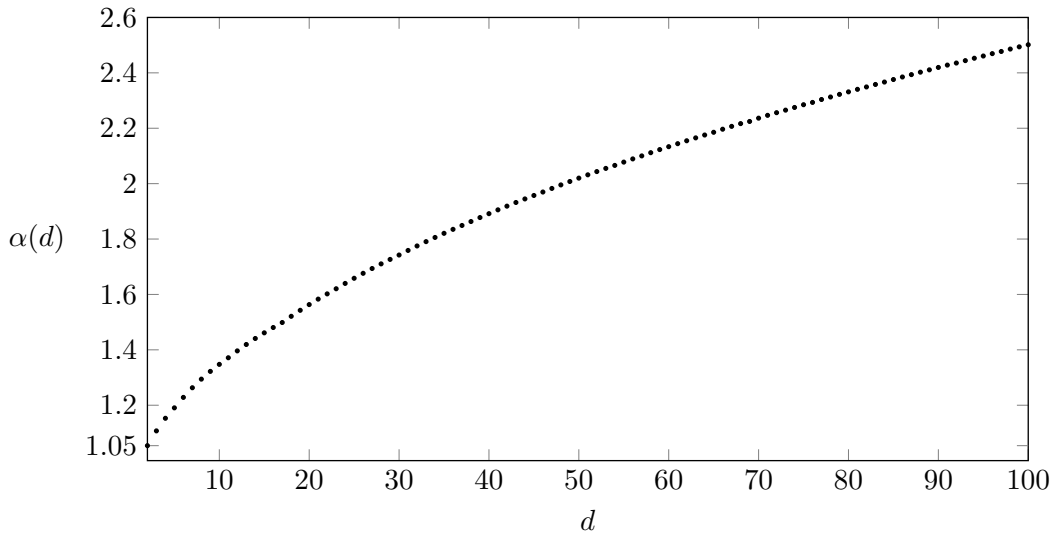


Figure 3.3: Values of $\alpha(d)$ for $d \in \{2, \dots, 100\}$, as given by (3.3). In particular, for small values of $d = 2, 3, 4$ we have $\alpha(2) \approx 1.054$, $\alpha(3) \approx 1.107$ and $\alpha(4) \approx 1.153$.

Figure 3.3 shows the values of $\alpha(d)$ for small values of d. We observe that for $d = 2, 3, 4$ the supremum of (3.3) is attained at $n = 1$, i.e., for 2 player games. Our bounds for $\alpha(d)$ are the same as the ones obtained by Hansknecht, Klimm, and Skopalik [HKS14], who study the non-existence of approximate equilibria in 2 player polynomial congestion games. Later we show that games with 2 players can not achieve a higher bound than 2. Thus for larger values of d we need more players to get a higher value in (3.3).

We conclude this section by giving the asymptotic growth of $\alpha(d)$.

Lemma 3. *For $d \in \mathbb{N}_{\geq 2}$ we have $\alpha(d) \in \Omega\left(\frac{\sqrt{d}}{\ln d}\right)$. More specifically, $\alpha(d) > \frac{\sqrt{d}}{2 \ln d}$ for large enough d.*

Proof. Take the following choice of the parameters:

$$k = \left\lceil \frac{\ln d}{2 \ln \ln d} \right\rceil, \quad w = \frac{\ln d}{2d}, \quad a = \frac{1}{d^{\frac{k}{2(k+1)}} (1+w)^d}, \quad n = \left\lfloor \frac{1}{d^{\frac{1}{2(k+1)}} w} \right\rfloor.$$

One can check that for $d \geq 4$ these choices satisfy $k \in \{1, \dots, d\}$ and $w, a \in [0, 1]$. We bound the terms appearing in (3.1) and (3.2) as follows.

$$\begin{aligned} 1 + na(1+w)^d &\geq 1 + \left(\frac{1}{d^{\frac{1}{2(k+1)}} w} - 1 \right) \frac{1}{d^{\frac{k}{2(k+1)}} (1+w)^d} (1+w)^d \\ &= 1 + \left(\frac{2d}{d^{\frac{1}{2(k+1)}} \ln d} - 1 \right) \frac{1}{d^{\frac{k}{2(k+1)}}} \\ &= \frac{2d}{d^{\frac{1}{2(k+1)} + \frac{k}{2(k+1)}} \ln d} + 1 - \frac{1}{d^{\frac{k}{2(k+1)}}} \end{aligned}$$

$$\geq \frac{2d}{d^{1/2}\ln d} \qquad \text{(since } d \geq 1\text{)}$$

$$= \frac{2\sqrt{d}}{\ln d}; \tag{3.4}$$

$$(1+nw)^k + na \leq \left(1 + \frac{1}{d^{\frac{1}{2(k+1)}}w}\right)^k + \frac{1}{d^{\frac{1}{2(k+1)}}wd^{\frac{k}{2(k+1)}}(1+w)^d}$$

$$= \left(1 + d^{-\frac{1}{2(k+1)}}\right)^k + \frac{1}{d^{1/2}\frac{\ln d}{2d}\left(1+\frac{\ln d}{2d}\right)^d}$$

$$= \left(1 + d^{-\frac{1}{2(k+1)}}\right)^k + \frac{2\sqrt{d}}{\ln d\left(1+\frac{\ln d}{2d}\right)^d}; \tag{3.5}$$

$$(1+w)^k + aw^d \geq 1; \tag{3.6}$$

$$(nw)^k + a(1+w)^d \leq \left(\frac{1}{d^{\frac{1}{2(k+1)}}w}\right)^k + \frac{1}{d^{\frac{k}{2(k+1)}}(1+w)^d}(1+w)^d$$

$$= 2 \cdot \frac{1}{d^{\frac{k}{2(k+1)}}} = \frac{2d^{\frac{1}{2(k+1)}}}{\sqrt{d}} \leq \frac{2d^{\frac{\ln\ln d}{\ln d}}}{\sqrt{d}} = \frac{2\ln d}{\sqrt{d}}. \tag{3.7}$$

In the Appendix, we prove (Lemma 43) that the final quantity in (3.5) converges to 1 as d goes to ∞. In particular, for large d it is upper bounded by 4. Numerically, we observe that $d \geq 8$ suffices. Thus, we can lower bound the ratios in (3.1) and (3.2) as

$$\frac{1+na(1+w)^d}{(1+nw)^k+na} \geq \frac{\frac{2\sqrt{d}}{\ln d}}{4} = \frac{\sqrt{d}}{2\ln d} \in \Omega\left(\frac{\sqrt{d}}{\ln d}\right), \qquad \text{(from (3.4), (3.5) and large } d\text{)}$$

$$\frac{(1+w)^k+aw^d}{(nw)^k+a(1+w)^d} \geq \frac{1}{\frac{2\ln d}{\sqrt{d}}} = \frac{\sqrt{d}}{2\ln d} \in \Omega\left(\frac{\sqrt{d}}{\ln d}\right). \qquad \text{(from (3.6) and (3.7))}$$

□

3.4 The Hardness Gadget

In this section we show NP-hardness of the existence of approximate equilibria with additional properties in unweighted congestion games. The main additional property is: Is a player playing a specific strategy? We show this hardness by a reduction from Circuit Satisfiability. From a Boolean circuit we construct an *unweighted* polynomial congestion game such that in any α-equilibrium the players emulate the computation of the circuit.

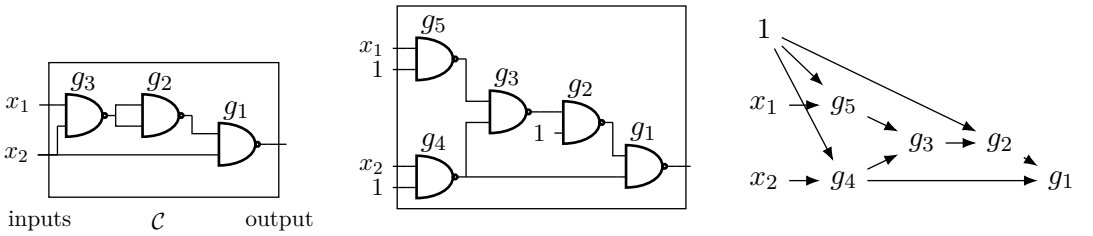
Circuit Satisfiability Recall the decision problem Circuit Satisfiability as introduced in Section 2.4. For a Boolean circuit $\mathcal{C}$ with n input bits and a vector $x \in \{0,1\}^n$, we denote the output of $\mathcal{C}$ for input x by $\mathcal{C}(x)$. So Circuit Satisfiability asks, given a circuit $\mathcal{C}$, is there a vector $x \in \{0,1\}^n$ such that $\mathcal{C}(x) = 1$?

This problem is NP-hard even for connected acyclic circuits with only 2-input NAND gates [Pap94]. A 2-input NAND gate computes the negation of the conjunction of its input bits. Figure 3.4 shows the truth table for a NAND gate. Hence, we only consider

x	y	out
0	0	1
0	1	1
1	0	1
1	1	0

Figure 3.4: Schematic drawing and semantics of a NAND gate.

connected acyclic circuits with 2-input NAND gates in this chapter. We make further assumptions on the structure of the circuits which do not change the hardness of the problem. They are merely syntactical transformations.



(a) A circuit $\mathcal{C}$.

(b) Canonical form of $\mathcal{C}$.

(c) Directed acyclic graph corresponding to $\mathcal{C}$'s canonical form.

Figure 3.5: Example of a circuit $\mathcal{C}$, its canonical form, and the corresponding directed graph.

Circuit Model Firstly, we assume that the two input bits to a NAND gate are different. If this is not the case, we set one of the input bits of this NAND gate to 1. See g_2 in Figure 3.5b. Observe from the truth table in Figure 3.4 that a NAND gate with one input fixed to 1 computes the negation of the free input bit just like a NAND gate with identical input bits.

Secondly, we negate every input bit of the circuit by adding a NAND gate with one input fixed to 1 in between. See Figure 3.5b for this transformation. Note that negating the input bits does not change the satisfiability of the circuit. Let $\mathcal{C}$ be a circuit and $\mathcal{C}'$ the circuit where we added NAND gates after the input bits as explained above. Then we have $\mathcal{C}(x) = \mathcal{C}'(\overline{x})$, where $\overline{x}$ is the vector resulting from x by flipping every bit. Hence, $\mathcal{C}$ is satisfiable exactly if $\mathcal{C}'$ is satisfiable.

A connected acyclic circuit containing only 2-input NAND gates where both input bits are different and every input of the circuit is connected to exactly one NAND gate where the other input is fixed to 1 is said to be in *canonical form*. See Figure 3.5b for an example of a circuit in canonical form.

We think of a circuit in canonical form as a *directed acyclic graph*. Every input bit, every gate and the fixed input 1 are the nodes of the graph. There is a directed arc between nodes, if the tail of the arc is an input to the head of the arc. This defines a directed acyclic graph. See Figure 3.5c for an example graph of a circuit. The gates are numbered in reverse topological ordering of this graph. Hence, the output bit of the circuit is the output of the first gate and the successors of a gate have all smaller numbers, i.e., $N^+(g_k) \subseteq \{g_j : j < k\}$. Since the circuit is in canonical form, we further have $|N^+(x_i)| = 1$ for every input bit x_i.

Translation to Congestion Games We will now construct a polynomial congestion game from the canonical form of a Boolean circuit in such a way that in any α-equilibrium the players emulate the computation of the gates in the circuit. Let $\mathcal{C}$ be a circuit in canonical form with input bits $x_1, \dots, x_n$, gates $g_1, \dots, g_K$ and the input fixed to 1. The game $\mathcal{G}^d_\mu(\mathcal{C})$ is given by the parameters

$$d \in \mathbb{N}_{\geq 1} \qquad \text{and} \qquad \mu \in \mathbb{R}_{\geq 0} \quad \text{where} \quad \mu > 1 + 2 \cdot 3^{d+d/2}. \tag{3.8}$$

It is an unweighted polynomial congestion game of degree d.

For every input bit there is an *input player* X_i, for every output of a gate there is a *gate player* G_k, and there is the *static player* for the input fixed to 1. We refer to G_1 as the *output player* as the output of g_1 is the output of the circuit.

For every gate g_k there are two *resources* 0_k and 1_k. Every player has two strategies representing the state of the corresponding bit. The *zero strategy* of a player contains her own zero resource (for gate players) together with all the zero resources of the direct successors. The *one strategy* is defined analogously with the one resources.

An input player X_i has the two strategies

$$s^0_{X_i} = \left\{0_k : g_k \in N^+(x_i)\right\} \qquad \text{and} \qquad s^1_{X_i} = \left\{1_k : g_k \in N^+(x_i)\right\}.$$

Recall that for a circuit in canonical form the input bits are connected to exactly one gate, hence the strategies of the input players contain exactly one resource.

The static player has only one strategy

$$s_{\text{static}} = \left\{1_k : N^+(1)\right\}$$

containing all one resources of gates where one input is fixed to 1.

For a gate player G_k the zero and one strategy are given by

$$s^0_{G_k} = \{0_k\} \cup \left\{0_l : g_l \in N^+(g_k)\right\} \qquad \text{and} \qquad s^1_{G_k} = \{1_k\} \cup \left\{1_l : g_l \in N^+(g_k)\right\}.$$

Since the gates are numbered by a reverse topological ordering, every strategy of a gate player G_k contains at most k resources. See Figure 3.6 for an illustration of the resources and strategies of $\mathcal{G}^d_\mu(\mathcal{C})$.

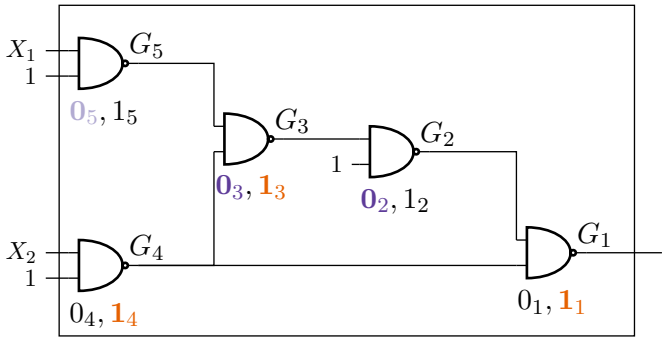


Figure 3.6: Structure of the game $\mathcal{G}_\mu^d(\mathcal{C})$ for the circuit $\mathcal{C}$ from Figure 3.5. Every gate has two resources $0_k, 1_k$, and an associated gate player G_k. Three strategies are highlighted: The one strategy of G_4 (orange), the zero strategy of G_3 (purple), and the zero strategy of X_1 (light purple).

Every gate g_k has 3 associated players: the gate player G_k and the two players corresponding to the input bits of the gate. Since the input bits of a NAND gate are assumed to be different by the definition of the canonical form, there are indeed three different players associated to every gate. From the above definitions of the strategies we observe that every resource 0_k and 1_k can only be played by the three players associated to gate g_k.

The resource cost functions are defined using parameter μ. We set

$$c_{1_k}(x) = \mu^k x^d \qquad \text{and} \qquad c_{0_k}(x) = \lambda \mu^k x^d \text{ where } \lambda = 3^{d/2}. \tag{3.9}$$

This finishes the description of the unweighted polynomial congestion game $\mathcal{G}_\mu^d(\mathcal{C})$.

Skopalik and Vöcking [SV08] also translate Boolean circuits to congestion games. Their construction differs in two main points. Firstly, they use 3 resources per gate and secondly, their resource cost functions are general (step) functions. Skopalik and Vöcking use these games to show PLS-hardness of computing approximate equilibria in congestion games with general increasing cost functions. Their result leaves open the following

Research Question 1. Is finding approximate equilibria in *polynomial* congestion games PLS-hard?

We were not able to specialize their construction to polynomial cost functions, or use our gadget for a PLS-reduction. However, we achieve several NP-hardness results of deciding the existence of approximate equilibria with additional properties by studying our gadget on its own.

α-equilibria in $\mathcal{G}_\mu^d(\mathcal{C})$ A strategy profile σ_X of the input players will be interpreted as a bit vector $x \in \{0,1\}^n$ by setting $x_i = 0$ if $s_{X_i} = s^0_{X_i}$ and $x_i = 1$ if $s_{X_i} = s^1_{X_i}$. To discuss the emulation of the computation of the circuit by the gate players in $\mathcal{G}_\mu^d(\mathcal{C})$, we introduce the term *following the NAND semantics*. In a strategy profile σ, the gate

players follow the NAND semantics, if every gate player plays her zero strategy if and only if both input players associated to the gate play their one strategy. As can be seen in the truth table in Figure 3.4 this corresponds to the semantics of a NAND gate. If in a profile σ all gate players follow the NAND semantics, and we take $x \in \{0,1\}^n$ to be the bit vector corresponding to the strategies of the input players in σ, then the output player G_1 plays according to $\mathcal{C}(x)$.

We show that for small enough α the α-equilibria in $\mathcal{G}_\mu^d(\mathcal{C})$ are exactly the profiles where the players emulate the computation of the circuit $\mathcal{C}$. To specify what "small enough" means, we define

$$\varepsilon(\mu) = \frac{3^{d+d/2}}{\mu - 1}. \tag{3.10}$$

Since $d \geq 1$ and $\mu > 1 + 2 \cdot 3^{d+d/2}$ (see (3.8)), we have

$$3^{d/2} - \varepsilon(\mu) \geq 3^{d/2} - \frac{1}{2} > 1$$

and hence, we can choose $\alpha \in [1, 3^{d/2} - \varepsilon(\mu))$.

Lemma 4. *For any circuit $\mathcal{C}$ in canonical form and valid choices of parameters d and μ (satisfying (3.8)) consider the game $\mathcal{G}_\mu^d(\mathcal{C})$. Let σ_X be a strategy profile for the input players $X_1, \ldots, X_n$ and let $x \in \{0,1\}^n$ be the corresponding bit vector.*

Then the partial profile σ_X can be extended to a unique *α-equilibrium σ of $\mathcal{G}_\mu^d(\mathcal{C})$ for any $\alpha \in [1, 3^{d/2} - \varepsilon(\mu))$. Further, in σ all gate players follow the NAND semantics, and in particular the output player G_1 plays according to $\mathcal{C}(x)$.*

Proof. Let $\mathcal{C}$ be a circuit in canonical form and take valid choices of d, μ and α.

We will first fix the input players to their strategies given by σ_X. We show that the gate players follow the NAND semantics in any α-equilibrium by showing that switching to the right strategy is an α-improving move. We will then see that once the gate players follow the NAND semantics, the input players do not have an incentive to change their strategies. This shows that there is a unique α-equilibrium where the input players play according to σ_X and the gate players emulate the computation of the circuit.

Input players fixed to σ_X. Let σ be an α-equilibrium of $\mathcal{G}_\mu^d(\mathcal{C})$, where the input players $X_1, \ldots, X_n$ play according to σ_X. Consider a gate g_k and its associated players G_k and P_a, P_b, where P_a and P_b correspond to the two input bits of the gate. We show that G_k plays her zero strategy exactly if both P_a and P_b play their one strategy.

Case 1: Both P_a and P_b play their one strategy in σ. The cost incurred to G_k on her one strategy is at least $c_{1_k}(3)$, since all three players G_k, P_a and P_b use 1_k. On the other hand, the cost for her zero strategy is at most $c_{0_k}(1) + \sum_{j=1}^{k-1} c_{0_j}(3)$. The improving factor is thus

$$\frac{C_{G_k}\left(\left(s^1_{G_k}, \sigma_{-G_k}\right)\right)}{C_{G_k}\left(\left(s^0_{G_k}, \sigma_{-G_k}\right)\right)} \geq \frac{c_{1_k}(3)}{c_{0_k}(1) + \sum_{j=1}^{k-1} c_{0_j}(3)} = \frac{\mu^k 3^d}{\lambda\mu^k + \sum_{j=1}^{k-1} \lambda\mu^j 3^d}.$$

Since $\frac{1}{\mu^k}\sum_{j=1}^{k-1}\mu^j = \frac{1}{\mu^k}\left(\frac{\mu^k-\mu}{\mu-1}\right) < \frac{1}{\mu-1}$, we can lower bound the above by

$$\frac{3^d}{\lambda}\left(\frac{1}{1+\frac{1}{\mu-1}3^d}\right) = 3^{d/2}\left(\frac{1}{1+\frac{1}{\mu-1}3^d}\right) > 3^{d/2}\left(1-\frac{1}{\mu-1}3^d\right) = 3^{d/2}-\varepsilon(\mu) > \alpha \quad (3.11)$$

where we used the definitions of λ (see (3.9)) and $\varepsilon(\mu)$ (see (3.10)). Thus, G_k plays her zero strategy in the α-equilibrium σ.
Case 2: At least one of P_a or P_b is playing her zero strategy in σ. The cost incurred to G_k on her zero strategy is at least $c_{0_k}(2)$, since at least one of P_a and P_b is using 0_k additionally to G_k. On the other hand, the cost for her one strategy is at most $c_{1_k}(2) + \sum_{j=1}^{k-1} c_{1_j}(3)$, as the one of P_a and P_b playing the zero strategy is not using 1_k. The improving factor is thus

$$\frac{C_{G_k}\left(\left(s^0_{G_k}, \sigma_{-G_k}\right)\right)}{C_{G_k}\left(\left(s^1_{G_k}, \sigma_{-G_k}\right)\right)} \geq \frac{c_{0_k}(2)}{c_{1_k}(2)+\sum_{j=1}^{k-1} c_{1_j}(3)} = \frac{\lambda\mu^k 2^d}{\mu^k 2^d + \sum_{j=1}^{k-1}\mu^j 3^d}.$$

Using the same bound on the sum of the μ^j as in the previous case, we lower bound the above by

$$\lambda\left(\frac{1}{1+\frac{1}{\mu-1}}3^d\right) > 3^{d/2}\left(\frac{1}{1+\frac{1}{\mu-1}3^d}\right) > 3^{d/2}\left(1-\frac{1}{\mu-1}3^d\right) = 3^{d/2}-\varepsilon(\mu) > \alpha \quad (3.12)$$

where we again used the definitions of λ and $\varepsilon(\mu)$. Thus, player G_k is playing her one strategy in the α-equilibrium σ.

This shows that in any α-equilibrium where the input players are fixed, the gate players follow the NAND semantics, and hence there is at most one such α-equilibrium. We now come to the second part. We show that the input players do not have an incentive to change their strategies once the gate players follow the NAND semantics.

Gate players follow NAND semantics. We will now show that in a profile where all gate players follow the NAND semantics, the input players do not have α-improving moves. Let σ be a strategy profile of $\mathcal{G}^d_\mu(\mathcal{C})$ where all gate players follow the NAND semantics and let X_i be one of the input players. Since $\mathcal{C}$ is in canonical form, there is exactly one gate g_k connected to x_i and the other input of this gate is fixed to 1.
Case 1: X_i plays her one strategy in σ. We know that G_k is playing her zero strategy, since she follows the NAND semantics and hence negates x_i. The cost incurred to X_i on her one strategy is $c_{1_k}(2) = \mu^k 2^d$. On the other hand, the cost of her zero strategy is $c_{0_k}(2) = \lambda\mu^k 2^d$. The improvement factor is thus

$$\frac{C_{X_i}\left(\left(s^0_{X_i}, \sigma_{-X_i}\right)\right)}{C_{X_i}\left(\left(s^1_{X_i}, \sigma_{-X_i}\right)\right)} = \frac{\lambda\mu^k 2^d}{\mu^k 2^d} = \lambda = 3^{d/2} > 3^{d/2}-\varepsilon(\mu) > \alpha.$$

This shows that her one strategy is α-dominating for X_i in σ.
Case 2: X_i plays her zero strategy in σ. Similarly to the previous case, we know that G_k plays her one strategy. The cost incurred to X_i on her zero strategy is $c_{0_k}(1) = \lambda\mu^k$. On

the other hand, the cost of her one strategy is $c_{1_k}(3) = \mu^k 3^d$. The improvement factor is thus

$$\frac{C_{X_i}\left(\left(s^1_{X_i}, \sigma_{-X_i}\right)\right)}{C_{X_i}\left(\left(s^0_{X_i}, \sigma_{-X_i}\right)\right)} = \frac{\mu^k 3^d}{\lambda \mu^k 3^d} = \frac{3^d}{\lambda} = 3^{d/2} > \alpha.$$

This shows that her zero strategy is α-dominating for X_i in σ.

This shows that the α-equilibrium is indeed unique. □

We observe from the above proof that the unique α-equilibrium is actually an *exact* equilibrium. Since every player has only two strategies, we showed that following the NAND semantics is α-dominating for every player. Hence, in the α-equilibrium all players play α-dominating strategies. Recall from the beginning that such a profile is an exact equilibrium.

With this key lemma we can now show NP-hardness of deciding the existence of approximate equilibria with additional properties. We emphasize that the parameters d, α and z are fixed and not part of the input of the problems in the following theorem.

Theorem 2. *For* unweighted *congestion games of degree $d \geq 1$, the problems*

- *"Is there an α-equilibrium where a given subset of players play specific strategies?"*
- *"Is there an α-equilibrium where a given resource is used by at least one player?"*
- *"Is there an α-equilibrium where a given player has cost at most z?"*

are NP-hard for any $\alpha \in [1, 3^{d/2})$ and any $z > 0$.

Proof.

α-equilibria with specific strategies. To show the NP-hardness of the first problem we reduce from Circuit Satisfiability. Let $\mathcal{C}$ be a Boolean circuit in canonical form. Take $\alpha \in [1, 3^{d/2})$. Then there is an $\varepsilon > 0$ such that $\alpha < 3^{d/2} - \varepsilon$. By taking μ to be a rational with

$$\mu > 1 + \frac{3^{d+d/2}}{\min\left\{\varepsilon, \frac{1}{2}\right\}}$$

we have $\mu > 1 + 2 \cdot 3^{d+d/2}$ and hence a valid choice of μ. Further, we obtain $\varepsilon(\mu) \leq \varepsilon$ and hence $3^{d/2} - \varepsilon(\mu) \geq 3^{d/2} - \varepsilon > \alpha$.

Now consider the game $\mathcal{G}^d_\mu(\mathcal{C})$. If $\lambda = 3^{d/2}$ is irrational we instead take a rational λ such that

$$\alpha\left(1 + \frac{1}{\mu - 1} 3^d\right) < \lambda < \frac{3^d}{\alpha\left(1 + \frac{1}{\mu-1} 3^d\right)}. \tag{3.13}$$

This preserves the key inequalities (3.11) and (3.12) and makes the game $\mathcal{G}^d_\mu(\mathcal{C})$ completely defined by rationals.

We show that there is an α-equilibrium where the output player G_1 plays her one strategy if and only if $\mathcal{C}$ is a Yes-instance to Circuit Satisfiability.

Case 1: $\mathcal{C}$ is Yes-instance. Let $x \in \{0,1\}^n$ be a bit vector such that $\mathcal{C}(x) = 1$ and let σ_X be the corresponding strategy profile for the input players of $\mathcal{G}^d_\mu(\mathcal{C})$. From our choices of

μ and α, we can apply Lemma 4 and extend the profile σ_X to an α-equilibrium where G_1 plays according to $\mathcal{C}(x)$, that is, $\sigma_{G_1} = s^1_{G_1}$.
Case 2: $\mathcal{C}$ is No-instance. Again using Lemma 4, we know that any choice of the strategies of the input players can be extended to a unique α-equilibrium. In this α-equilibrium the gate players emulate the computation of the circuit. Since for all vectors $x \in \{0,1\}^n$ we have $\mathcal{C}(x) = 0$, the output player is playing her zero strategy in any α-equilibrium.

α-equilibria with specific resource. For the NP-hardness of the second problem we use the same reduction as for the first problem. We add a new resource of cost 0 to the one strategy of the output player. Then there is an α-equilibrium where this new resource is used if and only if the circuit is a Yes-instance.

α-equilibria with upper bound on player cost. Again, we want to use a reduction similar to the one before. To have a better understanding of the costs incurred to the output player, we transform the circuit $\mathcal{C}$ in the following way. We negate the output of $\mathcal{C}$ by adding a NAND gate with fixed input 1 right after the first gate in $\mathcal{C}$. Let $\overline{\mathcal{C}}$ be the resulting circuit, then we have $\mathcal{C}(x) = \neg\overline{\mathcal{C}}(x)$.

Observe that $\overline{\mathcal{C}}$ is again in canonical form and the output player of $\overline{\mathcal{C}}$ corresponds to a NAND gate with one input fixed to 1. Consider the game $\mathcal{G}^d_\mu(\overline{\mathcal{C}})$, where we choose μ and λ as in the reduction for the first problem. We show that there is an α-equilibrium where the output player has cost at most $\lambda\mu$ if and only if $\mathcal{C}$ is a Yes-instance.
Case 1: $\mathcal{C}$ is Yes-instance. Let $x \in \{0,1\}^n$ be a bit vector such that $\mathcal{C}(x) = 1$ and let σ_X be the corresponding strategy profile for the input players of $\mathcal{G}^d_\mu(\overline{\mathcal{C}})$. Then we have $\overline{\mathcal{C}}(x) = 0$ and with Lemma 4 σ_X can be extended to an α-equilibrium σ where G_1 plays her zero strategy. Since the gate players follow the NAND semantics, both players corresponding to the two input bits of g_1 are playing their one strategy. Thus the cost incurred to G_1 by σ is $c_{0_1}(1) = \lambda\mu$.
Case 2: $\mathcal{C}$ is No-instance. As before we get from Lemma 4 that G_1 is playing her one strategy in any α-equilibrium, since for all bit vectors $x \in \{0,1\}^n$ we have $\mathcal{C}(x) = 0$ and thus $\overline{\mathcal{C}}(x) = 1$. Since the gate players follow the NAND semantics, the free input player to g_1 has to play her zero strategy. Thus the cost incurred to G_1 in σ is $c_{1_1}(2) = \mu 2^d$.

Since $\lambda = 3^{d/2} < 2^d$, we have shown that deciding the existence of an α-equilibrium where the output player has cost at most z is NP-hard, for all $\lambda\mu < z < 2^d\mu$. To show NP-hardness for any choice of $z > 0$, take a rational a with $a\lambda\mu < z < a2^d\mu$ and scale all resource cost functions in $\mathcal{G}^d_\mu(\overline{\mathcal{C}})$ by a. □

3.5 Hardness of Existence

We now show the main result of this chapter: that it is NP-hard to decide whether a polynomial congestion game has an α-equilibrium. Our reduction combines the circuit gadget of the previous section in a *black-box* way with any polynomial congestion game without α-equilibria (for example the games from Section 3.3). First, we make some assumptions on the structure of the game $\mathcal{G}$ without α-equilibria.

Structure of $\mathcal{G}$ We begin with two properties related to the players.

Every strategy contains at least one resource. If a player p had an empty strategy, then this would be α-dominating for p for any profile and any α. We can thus remove player p from the game without changing the (non-)existence of approximate equilibria.

Every player has positive weight. If a player p had weight zero, she would not affect the costs incurred to the other players. We can thus remove p from the game as above.

Further, we assume the resource cost functions to be monomials.

Every resource cost function is of the form $c_r(x) = a_r x^{k_r}$, where $a_r > 0$ and $k_r \in \{1, \dots, d\}$. Resources with cost zero can be removed from the game as they do not affect the choices of the players. Note that this may result in empty strategies for some players, but as explained above we can remove such players.

For a resource r with a general polynomial cost function $c_r(x) = a_{r,0} + a_{r,1}x + \ldots + a_{r,d}x^d$ we replace r by $d+1$ resources $(r,0), (r,1), \dots, (r,d)$ with monomial cost functions $a_{r,0}, a_{r,1}x, \dots, a_{r,d}x^d$ respectively. If a strategy contained r, it now contains all of $(r,0), (r,1), \dots, (r,d)$ and the cost of the strategy is unchanged. As before, if some of the coefficients are zero, we remove the resource.

Since we want all monomials to have degree at least 1, we replace resources r with constant cost $a_{r,0}$. For every player p that had r in some of her strategies, we introduce a new resource r_p with cost $\frac{a_{r,0}}{w_p}x$ and replace r by r_p in all her strategies. These new resources are only used by the respective player p. Since $c_{r_p}(w_p) = a_{r,0}$, the cost incurred to p does not change. In this way, the (non-)existence of approximate equilibria is maintained.

We say a game with the above properties is in *canonical form*. For a game $\mathcal{G}$ in canonical form with resources R and players P, we define

$$a_{\min}(\mathcal{G}) = \min_{r \in R} a_r, \quad W(\mathcal{G}) = \sum_{p \in P} w_p, \quad \text{and} \quad c_{\max}(\mathcal{G}) = \sum_{r \in R} c_r(W(\mathcal{G})). \tag{3.14}$$

Note that all of the above quantities are strictly positive and that the cost incurred to any player in any strategy profile is at most $c_{\max}$.

Later we need to scale $\mathcal{G}$ such that the weights of the players are small. For a polynomial congestion game $\mathcal{G}$ of degree d in canonical form and a scaling factor $\gamma \in (0, 1]$ let $\overline{\mathcal{G}}_\gamma$ be the game resulting from $\mathcal{G}$ when scaling the weights of the players and the coefficients of the resource cost functions by γ and γ^{d+1-k_r}. Formally, $\overline{\mathcal{G}}_\gamma$ has the same set of resources, players, and strategies as $\mathcal{G}$. The weights of the players and the resource cost functions change to

$$\overline{w}_p = \gamma w_p \qquad \text{and} \qquad \overline{c}_r(x) = \overline{a}_r x^{k_r}, \quad \text{where} \quad \overline{a}_r = \gamma^{d+1-k_r} a_r. \tag{3.15}$$

This transformation scales the player costs by γ^{d+1}. Thus, the existence of improving moves is maintained and hence also the (non-)existence of approximate equilibria.

We have the following relations between the parameters $a_{\min}, W$, and $c_{\max}$ of the original and of the scaled game.

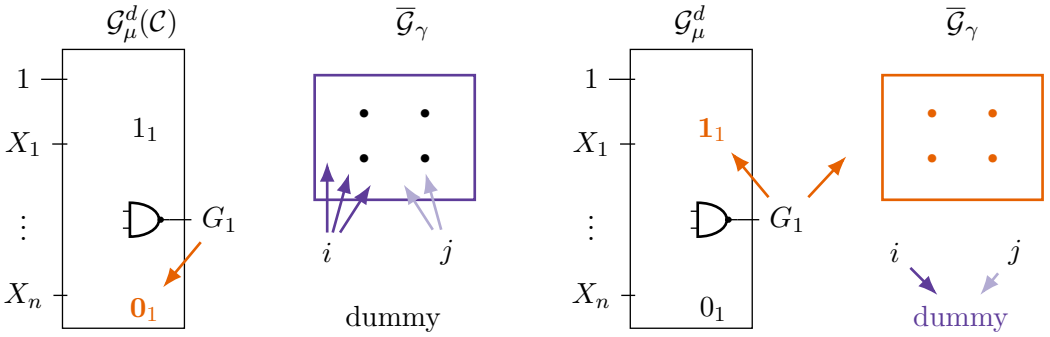
$$a_{\min}(\overline{\mathcal{G}}_\gamma) = \min_{r \in R} \gamma^{d+1-k_r} a_r \geq \gamma^d \min_{r \in R} a_r = \gamma^d a_{\min}(\mathcal{G}) \qquad (\text{since } k_r \geq 1) \tag{3.16}$$

$$W(\overline{\mathcal{G}}_\gamma) = \gamma W(\mathcal{G}) \tag{3.17}$$

$$c_{\max}(\overline{\mathcal{G}}_\gamma) = \sum_{r\in R} \overline{c}_r(W(\overline{\mathcal{G}}_\gamma)) = \sum_{r\in R} \gamma^{d+1} a_r W(\mathcal{G})^{k_r} = \gamma^{d+1} c_{\max}(\mathcal{G}). \tag{3.18}$$

Combining $\mathcal{G}$ with the Circuit Gadget We are now ready to describe the combination of $\mathcal{G}$ with the circuit gadget from Section 3.4.

The idea is to use the output player of the circuit gadget as mediator between the two games. If the output player plays her zero strategy then both games are *separated* and hence there is no α-equilibrium. If on the other hand, she plays her one strategy, she *stabilizes* $\mathcal{G}$ such that the game now has an α-equilibrium. See Figure 3.7 for an illustration of this idea.



(a) If G_1 plays her zero strategy, the games are separated. The players of $\overline{\mathcal{G}}_\gamma$ play in $\overline{\mathcal{G}}_\gamma$. Hence there is no α-equilibrium.

(b) If G_1 plays her one strategy, she additionally plays all resources of $\overline{\mathcal{G}}_\gamma$. The players of $\overline{\mathcal{G}}_\gamma$ play their new dummy strategy. This profile is an α-equilibrium.

Figure 3.7: Structure of the game $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. The output player G_1 acts as mediator between the circuit gadget $\mathcal{G}^d_\mu(\mathcal{C})$ and the non-existence gadget $\overline{\mathcal{G}}_\gamma$. The strategy of G_1 is shown in orange and the strategies of players in $\overline{\mathcal{G}}_\gamma$ are shown in purple and light purple.

We are now formalizing this idea. Recall that $\mathcal{G}$ is a polynomial congestion game of degree d in canonical form without any α-equilibria. Let $\mathcal{C}$ be a Boolean circuit in canonical form. To get the circuit game $\mathcal{G}^d_\mu(\mathcal{C})$ we need to specify parameter μ. We do this in a way similar to the first reduction in the proof of Theorem 2. Since polynomial congestion games of degree d have d-equilibria [CF21], we have $\alpha < d < 3^{d/2}$. Choose an $0 < \varepsilon < 3^{d/2} - \alpha$ and set μ to be a rational with

$$\mu > 1 + \frac{3^{d+d/2}}{\min\left\{\varepsilon, \frac{1}{2}\right\}}.$$

Then μ is a valid choice and $\alpha < 3^{d/2} - \varepsilon \leq 3^{d/2} - \varepsilon(\mu)$. Again, if $\lambda = 3^{d/2}$ is irrational, we take a rational λ satisfying (3.13).

To give the output player the power to stabilize $\mathcal{G}$, we use a scaled version of $\mathcal{G}$ as described above. We have to make sure that the weights of the players in $\overline{\mathcal{G}}_\gamma$ are small

compared to the weights in $\mathcal{G}_\mu^d(\mathcal{C})$ (which are all 1). We choose $\gamma \in (0,1]$ to be a sufficiently small rational such that

$$\gamma W(\mathcal{G}) < 1, \qquad \gamma \sum_{r \in R} a_r < \frac{\mu}{\mu - 1}\left(\frac{3}{2}\right)^d, \qquad \text{and} \qquad \gamma\alpha^2 < \frac{a_{\min}(\mathcal{G})}{c_{\max}(\mathcal{G})}. \tag{3.19}$$

The combined game $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ is given by:

- The players are the disjoint union of the players of $\mathcal{G}_\mu^d(\mathcal{C})$ of weight 1 and $\overline{\mathcal{G}}_\gamma$ of weight γw_p.
- The resources are the disjoint union of the 0_k and 1_k resources of $\mathcal{G}_\mu^d(\mathcal{C})$ and the resources R of $\overline{\mathcal{G}}_\gamma$ together with a new *dummy resource*.
- The resource cost function of the dummy resource is constantly $\frac{a_{\min}(\overline{\mathcal{G}}_\gamma)}{\alpha}$ and all other cost functions are unchanged.
- The *one strategy* of the output player is changed by adding *all* resources R of $\overline{\mathcal{G}}_\gamma$. For all players in $\overline{\mathcal{G}}_\gamma$ we add a new *dummy strategy* containing exactly the new dummy resource. All other strategies stay unchanged.

See Figure 3.7 for a graphical representation of this game.

We observe that the behavior of the circuit players is not affected by $\overline{\mathcal{G}}_\gamma$. That is, Lemma 4 still holds in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ for the circuit part.

Lemma 5. *Let σ_X be a strategy profile for the input players $X_1, \dots, X_n$ of the circuit part $\mathcal{G}_\mu^d(\mathcal{C})$ and $x \in \{0,1\}^n$ the corresponding bit vector. For all gate players it is α-dominating to follow the NAND semantics in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. In particular, it is α-dominating for the output player G_1 to play according to $\mathcal{C}(x)$ in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$.*

Proof. We use almost the same proof as for Lemma 4. Since the costs and strategies for all players but the output player are the same in $\mathcal{G}_\mu^d(\mathcal{C})$ and $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$, the same arguments show that it is α-dominating for those players to follow the NAND semantics. We only have to take care about the output player. Her zero strategy is unchanged and the cost of her one strategy increased, as she is now also playing all of the resources of $\overline{\mathcal{G}}_\gamma$. Hence, if $\mathcal{C}(x) = 0$ then it is α-dominating to play the zero strategy in $\mathcal{G}_\mu^d(\mathcal{C})$ and this is also the case in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. If on the other hand $\mathcal{C}(x) = 1$, then we have from Lemma 4 that G_1 plays according to $\mathcal{C}(x)$ in $\mathcal{G}_\mu^d(\mathcal{C})$, and hence not both of the input players of the output gate g_1 are playing their one strategy. As argued above, the behavior of these two players is the same in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. Thus, the cost of the zero strategy of G_1 in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ is at least $c_{0_1}(2) = \lambda\mu 2^d$ and the cost of her one strategy is at most

$$c_{1_1}(2) + \sum_{r \in R} \overline{c}_r(1 + W(\overline{\mathcal{G}}_\gamma)) = \mu 2^d + \sum_{r \in R} \gamma^{d+1-k_r} a_r (1 + \gamma W(\mathcal{G}))^{k_r}.$$

Using the first and second bound from (3.19) together with $\gamma \leq 1$, we can upper bound this by

$$\mu 2^d + \gamma \sum_{r \in R} a_r 2^d < \mu 2^d + \frac{\mu}{\mu - 1} 3^d.$$

The improvement factor for G_1 is thus at least

$$\frac{\lambda\mu 2^d}{\mu 2^d + \frac{\mu}{\mu-1}3^d} = \lambda\left(\frac{1}{1+\frac{1}{\mu-1}\left(\frac{3}{2}\right)^d}\right) > 3^{d/2}\left(\frac{1}{1+\frac{1}{\mu-1}3^d}\right) > 3^{d/2} - \varepsilon(\mu) > \alpha.$$

This shows that playing the one strategy is still α-dominating for G_1 in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. As in the proof of Lemma 4, the input players do not have an incentive to change their strategies once the gate players follow the NAND semantics. □

The next lemma shows the stabilization of $\overline{\mathcal{G}}_\gamma$ by the output player.

Lemma 6.

- *If the output player plays her one strategy in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$, it is α-dominating to play the dummy strategy for every player in $\overline{\mathcal{G}}_\gamma$.*
- *If the output player plays her zero strategy in $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$, the dummy strategy is α-dominated by any other strategy for every player in $\overline{\mathcal{G}}_\gamma$.*

Proof. First, let the output player play her one strategy and consider a player p in $\overline{\mathcal{G}}_\gamma$. If p is not playing her dummy strategy, she plays at least one resource $r \in R$ together with the output player. Hence her cost is at least

$$\overline{c}_r(1+\overline{w}_p) = \overline{a}_r(1+\overline{w}_p)^{k_r} > \overline{a}_r \geq a_{\min}(\overline{\mathcal{G}}_\gamma).$$

The strict inequality holds as $\mathcal{G}$ is in canonical form and hence $\overline{a}_r, \overline{w}_p$, and k_r are all strictly positive. On the other hand, the cost incurred to p on her dummy strategy is $\frac{a_{\min}(\overline{\mathcal{G}}_\gamma)}{\alpha}$. Thus the improvement factor is strictly greater than α, showing that the dummy strategy is α-dominating for every player in $\overline{\mathcal{G}}_\gamma$.

Now let the output player play her zero strategy. The dummy strategy of any player in $\overline{\mathcal{G}}_\gamma$ has cost of $\frac{a_{\min}(\overline{\mathcal{G}}_\gamma)}{\alpha} \geq \gamma^d \frac{a_{\min}(\mathcal{G})}{\alpha}$. On the other hand, any of the strategies in $\overline{\mathcal{G}}_\gamma$ incur a cost of at most $c_{\max}(\overline{\mathcal{G}}_\gamma) = \gamma^{d+1} c_{\max}(\mathcal{G})$. Thus the improvement factor is at least

$$\frac{\gamma^d a_{\min}(\mathcal{G})}{\alpha\gamma^{d+1}c_{\max}(\mathcal{G})} = \frac{a_{\min}(\mathcal{G})}{\alpha\gamma c_{\max}(\mathcal{G})}.$$

Using the third bound in (3.19) shows that the dummy strategy is α-dominated by any other strategy in $\overline{\mathcal{G}}_\gamma$. □

With these properties of both parts, we can show the key lemma for the combination of the games.

Lemma 7. *$\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ has an α-equilibrium if and only if $\mathcal{C}$ has a satisfying assignment.*

Proof. *"⇒":* Let σ be an α-equilibrium of $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. Let σ_X be the profile of the input players in σ and $x \in \{0,1\}^n$ be the corresponding bit vector. We show that the output player G_1 has to play her one strategy. From Lemma 6 we know that if G_1 plays her zero strategy then all players of $\overline{\mathcal{G}}_\gamma$ play one of their original strategies in

$\overline{\mathcal{G}}_\gamma$ (and not the dummy strategy). But since $\overline{\mathcal{G}}_\gamma$ does not have an α-equilibrium, this is a contradiction to σ being an α-equilibrium of $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ and thus, G_1 plays her one strategy. On the other hand, we get from Lemma 5 that in any α-equilibrium of $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ the gate players follow the NAND semantics, and in particular G_1 plays according to $\mathcal{C}(x)$. Thus, $\mathcal{C}(x) = 1$ and hence $\mathcal{C}$ has a satisfying assignment.
"$\Leftarrow$": Now let $x \in \{0,1\}^n$ be a satisfying assignment for $\mathcal{C}$ and let σ_X be the corresponding strategy profile for the input players of $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. Then by Lemma 5, the profile where all gate players follow the NAND semantics is a profile, where every strategy is α-dominating for every gate player. In particular, G_1 plays her one strategy. But then with Lemma 6 this profile can be extended to an α-equilibrium of $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ by letting all players of $\overline{\mathcal{G}}_\gamma$ play their dummy strategy. □

We observe that any α-equilibrium is indeed also an *exact* equilibrium, since we showed that all players play α-dominating strategies. Hence $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ has the following *gap-introducing* property:

- Either $\mathcal{C}$ has a satisfying assignment, then $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ has an exact equilibrium
- or $\mathcal{C}$ has no satisfying assignment and then $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ does not have any α-equilibria.

Hardness Results With the above lemmas, we can finally show the NP-hardness of deciding the existence of approximate equilibria in polynomial congestion games.

Theorem 3. *Fix a degree $d \in \mathbb{N}_{\geq 2}$ and a real $\alpha > 1$. If there is a weighted polynomial congestion game of degree d without an α-equilibrium, then it is NP-hard to decide whether a weighted polynomial congestion game of degree d has an α-equilibrium.*

Proof. For fixed $d \in \mathbb{N}_{\geq 2}$ and $\alpha > 1$ assume that there is a polynomial congestion game of degree d without α-equilibria. Then for any profile σ, there is a player p and a strategy $s' \in S_p \setminus \{\sigma_p\}$ such that $C_p(\sigma) > \alpha C_p((s', \sigma_{-p}))$. The player costs C_p are polynomials of degree d and hence continuous on the weights w_p and the coefficients a_r. Perturbing the weights and the coefficients will thus not change the above inequality. Hence there is a polynomial congestion game of degree d with rational weights and coefficients without an α-equilibrium. Similarly, we can choose a rational $\bar{\alpha} > \alpha$ such that the above inequality is still satisfied. Together with the transformations at the beginning of this section that do not change the (non-)existence of approximate equilibria, we get the existence of a polynomial congestion game $\mathcal{G}$ of degree d with rational weights and coefficients in canonical form without an $\bar{\alpha}$-equilibrium.

Consider the scaled game $\overline{\mathcal{G}}_\gamma$ as described above where we replace α by $\bar{\alpha}$. Since γ is chosen to be rational, $\overline{\mathcal{G}}_\gamma$ also has rational weights and coefficients.

We now show the statement by a reduction from Circuit Satisfiability. Let $\mathcal{C}$ be a Boolean circuit in canonical form. Construct the game $\mathcal{G}^d_\mu(\mathcal{C})$ as explained in the previous paragraph. Note that this can be done in polynomial time in the size of $\mathcal{C}$. Now consider the combined game $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. The part $\overline{\mathcal{G}}_\gamma$ involves only rational numbers and moreover its size is independent of the circuit $\mathcal{C}$. The dummy resource has

a constant cost which is a rational number. Changing the strategies as in the definition of $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ can be done in polynomial time.

We have thus constructed the game $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ from the circuit $\mathcal{C}$ in polynomial time. From Lemma 7 we have that this game has an $\bar{\alpha}$-equilibrium if and only if $\mathcal{C}$ is a YES-instance. We can choose $\bar{\alpha}$ to be close enough to α such that the set of $\bar{\alpha}$-equilibria and α-equilibria are the same. □

If in the above proof α is polynomial-time computable (see Section 3.2), then the problem is NP-complete, since we can check in polynomial time whether a profile is an α-equilibrium.

As we have shown in Section 3.3 that there are polynomial congestion games without α-equilibria for α growing as $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$, we immediately obtain NP-hardness of deciding the existence of such equilibria.

Corollary 1. *For $d \in \mathbb{N}_{\geq 2}$ let $\alpha(d)$ be the one from (3.3). Then it is NP-hard to decide whether a weighted polynomial congestion game of degree d has an α-equilibrium for any real $\alpha \in [1, \alpha(d))$.*

The proof of Lemma 7 shows that the reduction from CIRCUIT SATISFIABILITY used in the proof of Theorem 3 is *parsimonious*. There is a bijection between the satisfying assignments of $\mathcal{C}$ and the α-equilibria of $\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$. Thus we immediately get hardness of the respective counting versions.

Corollary 2. *For $d \in \mathbb{N}_{\geq 2}$ let $\alpha(d)$ be the one from (3.3) and let $\alpha \in [1, \alpha(d))$.*

- *It is #P-hard to count the number of α-equilibria in weighted polynomial congestion games of degree d.*
- *For a fixed $k \geq 1$, it is NP-hard to decide whether a weighted polynomial congestion game of degree d has at least k distinct α-equilibria.*

Proof. The first result is a consequence of the #P-hardness of the counting version of CIRCUIT SATISFIABILITY (see [Pap94]).

For the second result consider the following version of CIRCUIT SATISFIABILITY. For a fixed $k \geq 1$, decide whether a Boolean circuit has at least k distinct satisfying assignments. This problem is NP-complete, since we can add $\lceil \log k \rceil$ "dummy input bits" to the circuit that are not connected to the rest. Then the new circuit has at least k distinct satisfying assignments if and only if the original circuit has at least one satisfying assignment. We can even transform this circuit in polynomial time into canonical form by connecting the dummy input bits to the original circuit $\mathcal{C}$. Combine the dummy inputs by 2-input NAND gates in any way to a single output O and add a wiring of the form $NAND(NAND(\mathcal{C}, O), NAND(\mathcal{C}, \neg O))$. This will forward the output of $\mathcal{C}$ independently of the assignment of the dummy input bits. □

3.6 General Cost Functions

In this section, we consider general non-decreasing cost functions. In the previous sections, the parameter of interest was the degree d of the cost functions. Since we do

not have polynomial cost functions anymore, we instead take the number of players n as parameter. We show the existence of n-equilibria and the non-existence of $\Theta(\frac{n}{\ln n})$-equilibria together with the NP-hardness of deciding the existence of $\Theta(\frac{n}{\ln n})$-equilibria similarly to the previous sections.

Existence of n-Equilibria To show the existence of n-equilibria in n player games, we observe that the social cost strictly decreases at every n-improving move. This is closely related to the result of Caragiannis and Fanelli [CF21] that the weighted social cost decreases at every $(d+1)$-improving move for polynomial congestion games of degree d. This shows that (global or local) minimizers of the social cost are n-equilibria and additionally that those can be reached by a sequence of n-improving moves.

Theorem 4. *Every weighted congestion game with general non-decreasing cost functions and $n \geq 2$ players has an n-equilibrium.*

Proof. Let $\mathcal{G}$ be a weighted congestion game with n players. We show that a minimizer of the social cost is an n-equilibrium. Take σ to be a global minimizer of the social cost. Assume for a contradiction that player p has an n-improving move $s' \in S_p$ in σ and define $\sigma' = (s', \sigma_{-p})$.

First, we bound the change of the cost of any other player $q \neq p$ by the new cost of p. We have

$$\begin{aligned} C_q(\sigma') - C_q(\sigma) &= \sum_{r \in \sigma_q} c_r(n_{\sigma'}(r)) - \sum_{r \in \sigma_q} c_r(n_\sigma(r)) \\ &= \sum_{r \in \sigma_q \cap (s' \setminus \sigma_p)} (c_r(n_{\sigma'}(r)) - c_r(n_\sigma(r))) \\ &+ \sum_{r \in \sigma_q \cap (\sigma_p \setminus s')} (c_r(n_{\sigma'}(r)) - c_r(n_\sigma(r))). \end{aligned}$$

As the cost functions are non-decreasing, all of the terms in the second sum are non-positive. We can thus bound the change in the player cost of q from above by

$$C_q(\sigma') - C_q(\sigma) \leq \sum_{r \in \sigma_q \cap (s' \setminus \sigma_p)} (c_r(n_{\sigma'}(r)) - c_r(n_\sigma(r))) \leq \sum_{r \in s'} c_r(n_{\sigma'}(r)) = C_p(\sigma').$$

If we add the above inequalities for all of the $n-1$ players $q \neq p$, we obtain

$$\sum_{q \neq p} C_q(\sigma') - \sum_{q \neq p} C_q(\sigma) \leq (n-1) C_p(\sigma').$$

As the social cost is defined as the sum of all player costs, we equivalently have

$$(C(\sigma') - C_p(\sigma')) - (C(\sigma) - C_p(\sigma)) \leq (n-1) C_p(\sigma').$$

Rearranging the terms yields

$$C(\sigma') - C(\sigma) \leq n C_p(\sigma') - C_p(\sigma). \tag{3.20}$$

Since s' is an n-improving move for p in σ, we have that the right hand side is strictly negative. On the other hand, since σ is a global minimizer of the social cost, the left hand side is non-negative. This is a contradiction, and thus shows that any minimizer of the social cost is an n-equilibrium. □

In [CR06; CR09; CKS09; CKS11; Chr+18; Chr+19] the concept of *approximate potential* functions was studied. Here we showed in (3.20) that the social cost is an n-approximate potential for any congestion game with non-decreasing cost functions.

Non-Existence of $\Theta\left(\frac{n}{\ln n}\right)$-Equilibria We are giving a game with n players that does not have $\Theta(\frac{n}{\ln n})$-equilibria. The construction of the game is similar to the non-existence games for polynomial cost functions of degree d from Section 3.3. We introduce the real number φ_n[1] to be the unique positive solution of the equation

$$(x+1)^n = x^{n+1}.$$

This means that φ_n satisfies

$$\varphi_n = \left(1 + \frac{1}{\varphi_n}\right)^n.$$

This sequence is strictly increasing in n and asymptotically $\varphi_n \sim \frac{n}{\ln n}$ (see [Chr+19, Lemma A.3]). For $n = 1$, the number φ_1 is the golden ratio.

The game $\mathcal{G}_n$ has n players: a *heavy* player, and $(n-1)$ *light* players $1, \dots, n-1$. The weight of the heavy player is 1 and for a light player i it is $w_i = \frac{1}{2^i}$. There are $2n$ resources $r_0, r_1, \dots, r_{n-1}$ and $r'_0, r'_1, \dots, r'_{n-1}$. The cost function for r_0 and r'_0 is given by

$$c_0(x) = \begin{cases} 1, & \text{if } x \geq w_0, \\ 0, & \text{otherwise} \end{cases}$$

and for resource r_i and r'_i ($i \in \{1, \dots, n-1\}$) by

$$c_i(x) = \begin{cases} \frac{1}{\varphi_{n-1}}\left(1 + \frac{1}{\varphi_{n-1}}\right)^{i-1}, & \text{if } x \geq w_0 + w_i, \\ 0, & \text{otherwise} \end{cases}.$$

Every player has two strategies. The heavy player plays either all r resources or all r' resources:

$$S_{\text{heavy}} = \{\{r_0, r_1, \dots, r_{n-1}\}, \{r'_0, r'_1, \dots, r'_{n-1}\}\}.$$

A light player i plays her own r_i or r'_i resource together with all the previous resources of the other type:

$$S_i = \{\{r_0, r_1, \dots, r_{i-1}, r'_i\}, \{r'_0, r'_1, \dots, r'_{i-1}, r_i\}\}.$$

See Figure 3.8 for an illustration of the structure of the strategies.

[1] This number is usually denoted by Φ_n in the literature. To avoid confusion with the potential function Φ, we use φ_n here.

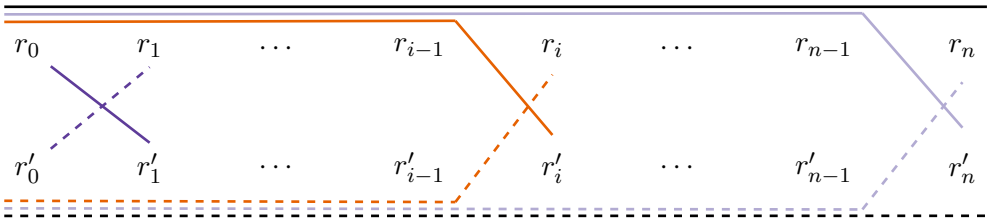


Figure 3.8: The strategies of $\mathcal{G}_n$. The heavy player plays either all r resources (solid black) or all r' resource (dashed black). The two strategies (solid and dashed) of light players $1, i$, and n are shown in purple, orange, and light purple, respectively.

We show that $\mathcal{G}_n$ does not have an α-equilibrium for any $\alpha < \varphi_{n-1}$. Similarly to Section 3.3, due to symmetries there are only two settings to consider. For both cases we show that some player has an φ_{n-1}-improving move.

Theorem 5. *For $n \in \mathbb{N}_{\geq 2}$ and $\alpha < \varphi_{n-1}$, there exists a weighted congestion game with n players and general non-decreasing cost functions that does not have an α-equilibrium.*

Proof. Consider the game $\mathcal{G}_n$ as described above and a strategy profile σ.
Case 1: The heavy player is alone on resource r_0. In this case the heavy player is playing $\{r_0, r_1, \ldots, r_{n-1}\}$ and all of the light players play $\{r'_0, r'_1, \ldots, r'_{i-1}, r_i\}$. Hence, the current cost of the heavy player is

$$c_0(w_0) + \sum_{i=1}^{n-1} c_i(w_0 + w_i) = 1 + \frac{1}{\varphi_{n-1}} \sum_{i=1}^{n-1} \left(1 + \frac{1}{\varphi_{n-1}}\right)^{i-1} = \left(1 + \frac{1}{\varphi_{n-1}}\right)^{n-1} = \varphi_{n-1}.$$

Deviating to $\{r'_0, r'_1, \ldots, r'_{n-1}\}$ incurs a cost of

$$c_0(w_0 + w_1 + \ldots + w_{n-1}) + \sum_{i=1}^{n-1} c_i \left(w_0 + \sum_{j=i+1}^{n-1} w_j\right).$$

From the structure of the weights, we have

$$\sum_{j=i+1}^{n-1} w_j < w_i \tag{3.21}$$

for all $i \in \{1, \ldots, n-1\}$. Hence the above cost evaluates to $1 + 0$, which shows that the heavy player has an φ_{n-1}-improving move in σ.
Case 2: The heavy player shares r_0 with at least one light player. Let i be the light player with smallest index playing r_0. Then i plays $\{r_0, r_1, \ldots, r_{i-1}, r'_i\}$ and all light players $j \in \{1, \ldots, i-1\}$ play $\left\{r'_0, r'_1, \ldots, r'_{j-1}, r_j\right\}$. The current cost of player i is thus at least

$$c_0(w_0 + w_i) + \sum_{j=1}^{i-1} c_j(w_0 + w_i + w_j) = 1 + \frac{1}{\varphi_{n-1}} \sum_{j=1}^{i-1} \left(1 + \frac{1}{\varphi_{n-1}}\right)^{j-1} = \left(1 + \frac{1}{\varphi_{n-1}}\right)^{i-1}.$$

Deviating to her other strategy $\{r'_0, r'_1, \dots, r'_{i-1}, r_i\}$ incurs a cost of at most

$$c_0\left(\sum_{j=1}^{n-1} w_j\right) + \sum_{j=1}^{i-1} c_j\left(\sum_{k=j+1}^{n-1} w_k\right) + c_i\left(w_0 + \sum_{k=i}^{n-1} w_k\right).$$

Using again (3.21) for $i = 0$ (where we set $w_0 = 1 = w_{\text{heavy}}$) and $i = j$, we evaluate the above to

$$0 + 0 + \frac{1}{\varphi_{n-1}}\left(1 + \frac{1}{\varphi_{n-1}}\right)^{i-1}.$$

This shows that player i has an φ_{n-1}-improving move. □

NP-Hardness Result Similarly to Section 3.5, we combine the above non-existence gadget with the circuit gadget from Section 3.4 to obtain NP-hardness of deciding the existence of α-equilibria for congestion games with general non-decreasing cost functions. Here, our approximation bound α is not fixed, but depends on the number of players n. However, we are not restricted to use polynomial cost functions anymore in our reduction. We show the following theorem.

Theorem 6. *For a fixed $1 > \varepsilon > 0$ and a sequence of reals $\alpha(n) : \mathbb{N}_{\geq 2} \to \mathbb{R}$ where $1 \leq \alpha(n) < \frac{\varphi_{n-1}}{1+\varepsilon}$ it is NP-hard to decide whether a weighted congestion game $\mathcal{G}$ with $n_{\mathcal{G}}$ players has an $\alpha(n_{\mathcal{G}})$-equilibrium.*

Proof. The proof is done by a reduction from Circuit Satisfiability and follows the structure of Section 3.5. We first describe how to build the circuit part and continue to give the combination of this gadget with the non-existence games from above. We will see that in the combined game, the gate players in the circuit part still follow the NAND semantics. As before the output player acts as mediator between the two games and stabilizes the non-existence gadget when playing her zero strategy such that the combined game has an $\alpha(n)$-equilibrium exactly if the circuit has a satisfying assignment.

Let $\mathcal{C}$ be a Boolean circuit in canonical form. We have to build a congestion game with n players, such that the game has an $\alpha(n)$-equilibrium if and only if $\mathcal{C}$ has a satisfying assignment.

For technical reasons we negate the output of $\mathcal{C}$ by adding an extra NAND gate with one input fixed to 1. Let $\overline{\mathcal{C}}$ be the resulting circuit. Then $\overline{\mathcal{C}}(x) = \neg\mathcal{C}(x)$ and hence $\mathcal{C}$ is a Yes-instance, if there is an assignment of the input bits x with $\overline{\mathcal{C}}(x) = 0$. Let m be the number of input bits and K be the number of NAND gates of $\overline{\mathcal{C}}$. Then there are $n_1 = m + 1 + K$ players in the circuit gadget.

Preliminary Choices of Parameters. In the non-existence games $\mathcal{G}_n$, the quantity φ_{n-1} appears. This number may in general be irrational. We thus consider an integer under-approximation

$$\bar{\varphi}_{n-1} = \left\lfloor \frac{n}{\ln n} \right\rfloor.$$

Recall that $\varphi_{n-1} \sim \frac{n}{\ln n}$ and hence we can choose a large enough $n_0 \in \mathbb{N}$ such that

$$\forall n \geq n_0 : \left(1 - \frac{\varepsilon}{3}\right)\frac{n}{\ln n} \leq \bar{\varphi}_{n-1} < \varphi_{n-1} \leq \left(1 + \frac{\varepsilon}{3}\right)\frac{n}{\ln n}. \tag{3.22}$$

We further choose a large enough $\ell \in \mathbb{N}$ such that

$$1 + \frac{1}{\ell} < \frac{(1+\varepsilon)(1-\frac{\varepsilon}{3})}{1+\frac{\varepsilon}{3}}. \tag{3.23}$$

Since $0 < \varepsilon < 1$, the right hand side is indeed strictly larger than 1.

The number of players considered for the non-existence game is set to

$$n_2 = \ell n_1. \tag{3.24}$$

We assume that $n_1 \geq n_0$. If n_1 was bounded by a constant, it could be decided in polynomial time whether $\mathcal{C}$ has a satisfying assignment.

Observe that the parameters n_1 and n_2 can be found in time polynomial in the description of $\mathcal{C}$.

The Circuit Gadget. For the circuit part $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$ we have to define the degree d and parameter μ. We have to choose those parameters to be rational and of size polynomial in n_1, the size of $\mathcal{C}$. We choose $d \in \mathbb{N}_{\geq 2}$ such that

$$3^{d/2} > 1 + \bar{\varphi}_{n_2-1} \qquad \text{and} \qquad 3^{d/2} > \left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1}. \tag{3.25}$$

Since ℓ is independent of $\mathcal{C}$, we have that n_2 is polynomial in n_1. Further, note that d is logarithmic in n_2 as $\bar{\varphi}_{n_2-1}$ grows asymptotically as $\frac{n_2}{\ln n_2}$ and $\left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1}$ grows as n_2. For μ we choose an integer with $\mu > 1 + 2 \cdot 3^{d+d/2}$. Then $\varepsilon(\mu) < 1$ (see (3.10)) and

$$3^{d/2} - \bar{\varphi}_{n_2-1} > \varepsilon(\mu).$$

Hence we can choose a rational λ such that

$$\bar{\varphi}_{n_2-1}\left(1 + \frac{1}{\mu - 1}3^d\right) < \lambda < \frac{3^d}{\bar{\varphi}_{n_2-1}\left(1 + \frac{1}{\mu-1}3^d\right)}. \tag{3.26}$$

The numerator and denominator of both bounds are of size polynomial in n_2, hence λ can be chosen to be of size polynomial in n_2. This shows that the parameters d, μ and λ for the circuit gadget $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$ can all be found in time polynomial in n_1 and for any $\tilde{\alpha} < \bar{\varphi}_{n_2-1} < 3^{d/2} - \varepsilon(\mu)$ the players emulate the computation of $\overline{\mathcal{C}}$ in any $\tilde{\alpha}$-equilibrium (Lemma 4).

The Non-Existence Gadget. In the non-existence games $\mathcal{G}_{n_2}$ we use the integer $\bar{\varphi}_{n_2-1}$ instead of φ_{n_2-1} and call the resulting game $\mathcal{G}'_{n_2}$. Then for any $\tilde{\alpha} < \bar{\varphi}_{n_2-1}$ the $\tilde{\alpha}$-improving moves are not changed, as $\bar{\varphi}_{n_2-1} < \left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1}$. In particular, for any $\tilde{\alpha} < \bar{\varphi}_{n_2-1}$ the game $\mathcal{G}'_{n_2}$ does not have $\tilde{\alpha}$-equilibria.

Combining $\mathcal{G}'_{n_2}$ with the Circuit Gadget. We combine both games via the output player of $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$. As in Section 3.5, we scale the game $\mathcal{G}'_{n_2}$. We divide all weights and breakpoints in the cost functions by 2 and denote by $\overline{\mathcal{G}}_{n_2}$ the resulting game. Then the sum of all player weights in $\overline{\mathcal{G}}_{n_2}$ is less than 1 and the behavior of the players is the same in both games.

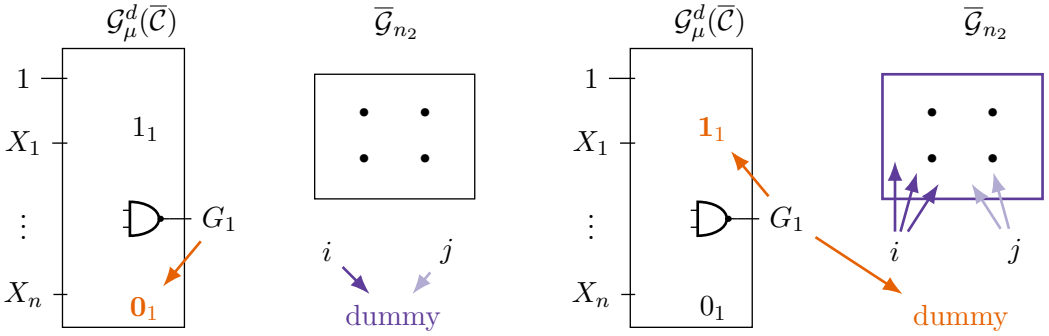
The combined game $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$ is given by:

- The players are the disjoint union of the players of $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$ of weight 1 and the players of $\overline{\mathcal{G}}_{n_2}$ with the respective weights $\frac{1}{2}w_i$.
- The resources are the disjoint union of the 0_k and 1_k resources of $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$ and the resources R of $\overline{\mathcal{G}}_{n_2}$ together with a new *dummy resource*.
- The resource cost function of the dummy resource is

$$c_{\text{dummy}}(x) = \begin{cases} \bar{\varphi}_{n_2-1}\left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1}, & \text{if } x \geq 1, \\ 0, & \text{otherwise} \end{cases}$$

  and all other cost functions are unchanged.
- The *one strategy* of the output player is changed by adding the dummy resource. For all players in $\overline{\mathcal{G}}_{n_2}$ we add a new *dummy strategy* containing exactly the new dummy resource. All other strategies stay unchanged.

See Figure 3.9 for a graphical representation of $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$.



(a) If G_1 plays her zero strategy, the dummy resource incurs cost zero to every player of $\overline{\mathcal{G}}_{n_2}$. Hence there is an $\tilde{\alpha}$-equilibrium.

(b) If G_1 plays her one strategy, she additionally plays the dummy resource. It is now $\tilde{\alpha}$-dominating for every player of $\overline{\mathcal{G}}_{n_2}$ to play in the original game. Hence there is no $\tilde{\alpha}$-equilibrium.

Figure 3.9: Structure of the game $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$. The output player G_1 acts as mediator between the circuit gadget $\mathcal{G}_\mu^d(\mathcal{C})$ and the non-existence gadget $\overline{\mathcal{G}}_{n_2}$. The strategy of G_1 is shown in orange and the strategies of players in $\overline{\mathcal{G}}_\gamma$ are shown in purple.

The combined game has $n = n_1 + n_2$ players. It remains to show that $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$ has an $\alpha(n)$-equilibrium if and only if $\mathcal{C}$ has a satisfying assignment. First, we show this statement for any $\tilde{\alpha} < \bar{\varphi}_{n_2-1}$ and then we show that $\alpha(n) < \bar{\varphi}_{n_2-1}$.

NAND Semantics and Stabilization of $\overline{\mathcal{G}}_{n_2}$. Let σ be an $\tilde{\alpha}$-equilibrium in $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$. Let σ_X be the profile for the input players and let $x \in \{0,1\}^m$ be the corresponding bit vector.

Since $\tilde{\alpha} < \bar{\varphi}_{n_2-1} < 3^{d/2} - \varepsilon(\mu)$ we can reuse the proof of Lemma 4. Since the costs and strategies of all players in $\mathcal{G}_\mu^d(\overline{\mathcal{C}})$ but the output player are the same in $\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$, the same arguments show that it is $\tilde{\alpha}$-dominating for those players to follow the NAND semantics. We only have to consider the output player.

Her zero strategy is unchanged and the cost of her one strategy increased as she is now also playing the dummy strategy. Hence, if $\overline{\mathcal{C}}(x) = 0$, then it is still $\tilde{\alpha}$-dominating to play the zero strategy.

If on the other hand $\overline{\mathcal{C}}(x) = 1$, then we have from Lemma 4 that G_1 plays according to $\overline{\mathcal{C}}(x)$ in $\mathcal{G}^d_\mu(\overline{\mathcal{C}})$. Thus not both of the input players of the output gate g_1 play their one strategy. As before the behavior of those players is the same in $\mathcal{G}^d_\mu(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$. Hence, the cost of the zero strategy of G_1 in $\mathcal{G}^d_\mu(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$ is at least $c_{0_1}(2) = \lambda\mu 2^d$ and the cost of her one strategy is at most

$$c_{1_1}(2) + \bar{\varphi}_{n_2-1}\left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1} < \mu 2^d + \frac{\mu}{\mu - 1} 3^d.$$

Thus, the improvement factor for G_1 is at least $\tilde{\alpha}$ as in Lemma 5. This shows that following the NAND semantics remains $\tilde{\alpha}$-dominating for all gate players, and hence as in the proof of Lemma 4, the input players do not have an incentive to change their strategies.

Claim 1.

- If the output player plays her zero strategy, it is $\tilde{\alpha}$-dominating to play the dummy strategy for every player in $\overline{\mathcal{G}}_{n_2}$.
- If the output player plays her one strategy, the dummy strategy is $\tilde{\alpha}$-dominated by any other strategy for every player in $\overline{\mathcal{G}}_{n_2}$.

Proof. If the output player plays her zero strategy, she is not using the dummy resource. Hence, the cost of the dummy strategy for any of the players in $\overline{\mathcal{G}}_{n_2}$ is zero, and hence $\tilde{\alpha}$-dominating.

Now, observe that the cost of any of the original strategies of the players in $\overline{\mathcal{G}}_{n_2}$ is upper bounded by $\left(1 + \frac{1}{\bar{\varphi}_{n_2-1}}\right)^{n_2-1}$. If the output player plays her one strategy, then the dummy resource is $\bar{\varphi}_{n_2-1}$-dominated by any other strategy of the players in $\overline{\mathcal{G}}_{n_2}$. □

Claim 2. $\mathcal{G}^d_\mu(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$ has an $\tilde{\alpha}$-equilibrium if and only if $\mathcal{C}$ has a satisfying assignment.

Proof. The proof is the same as for Lemma 7 using Claim 1 and swapping zero and one strategy of the output player. In any $\tilde{\alpha}$-equilibrium, the output player plays her zero strategy. As otherwise by Claim 1, the players of $\overline{\mathcal{G}}_{n_2}$ do not have an $\tilde{\alpha}$-equilibrium. If the output player, plays her zero strategy, then by the discussion at the beginning of the proof, the gate players follow the NAND semantics, and hence $\overline{\mathcal{C}}(x) = 0$. Showing that $\mathcal{C}$ has a satisfying assignment.

On the other hand for a satisfying assignment, the gate players follow the NAND semantics and hence the output player plays her zero strategy. With Claim 1, we obtain an $\tilde{\alpha}$-equilibrium by letting all players of $\overline{\mathcal{G}}_{n_2}$ play their dummy strategies. □

As for polynomial cost functions (see comment after Lemma 7), we observe that any $\tilde{\alpha}$-equilibrium is also an exact equilibrium. Hence, we again have the gap-introducing property of the combined game. Either $\mathcal{C}$ has a satisfying assignment, in which case the

game has an exact equilibrium, or $\mathcal{C}$ does not have a satisfying assignment, in which case the game does not have an $\tilde{\alpha}$-equilibrium.

Feasibility of $\alpha(n)$. We are left to show that $\alpha(n) < \bar{\varphi}_{n_2}$. By the definition of $\alpha(n)$ we have

$$\alpha(n) = \alpha(n_1 + n_2) < \frac{\bar{\varphi}_{n_1+n_2}}{1+\varepsilon}.$$

Using the upper and lower bounds on $\bar{\varphi}$ from (3.22) we continue with

$$\begin{aligned}\alpha(n) &\leq \frac{1+\frac{\varepsilon}{3}}{1+\varepsilon} \cdot \frac{n_1+n_2}{\ln(n_1+n_2)} \\ &\leq \frac{1+\frac{\varepsilon}{3}}{(1+\varepsilon)(1-\frac{\varepsilon}{3})} \cdot \frac{\ln n_2}{n_2} \cdot \frac{n_1+n_2}{\ln(n_1+n_2)} \bar{\varphi}_{n_2-1},\end{aligned}$$

where we used for the second inequality that $n_1 + n_2 \geq n_2 \geq n_1 \geq n_0$. As $\ln n_2 < \ln(n_1 + n_2)$ and $n_2 = \ell n_1$ we continue with

$$\begin{aligned}\alpha(n) &< \frac{1+\frac{\varepsilon}{3}}{(1+\varepsilon)(1-\frac{\varepsilon}{3})} \cdot \left(1 + \frac{n_1}{n_2}\right) \bar{\varphi}_{n_2-1} \\ &= \frac{1+\frac{\varepsilon}{3}}{(1+\varepsilon)(1-\frac{\varepsilon}{3})} \cdot \left(1 + \frac{1}{\ell}\right) \bar{\varphi}_{n_2-1}.\end{aligned}$$

Finally, by the choice of ℓ (3.23) we have $\alpha(n) < \bar{\varphi}_{n_2-1}$. □

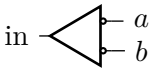
As before in Theorem 3, if the sequence $\alpha(n)$ is polynomial-time computable (as described in Section 3.2), the problem is NP-complete.

3.7 Network Games

Up to now we assumed the strategies of the games to be given explicitly. In some cases the strategies of the players can be represented in a succinct way. One such example are network games. There may be exponentially many paths for a player connecting her source and sink node. However, they are given implicitly by the underlying graph. In this section we discuss how our hardness results can be carried over to *network* games. We begin with the hardness gadget from Section 3.4, where we are able to give a construction of a network game with the same properties as $\mathcal{G}_\mu^d(\mathcal{C})$. For the non-existence gadget $\mathcal{G}_n$ from Section 3.3, we give an idea of a translation to a network game. Unfortunately, we were not able to find a complete construction. Finally, we briefly discuss the combination of both gadgets.

The Hardness Gadget We show that the game $\mathcal{G}_\mu^d(\mathcal{C})$ from Section 3.4 can be translated to a network congestion game. For this we refine our circuit model so that no connection splits, i.e., every output of a gate is input to at most one other gate.

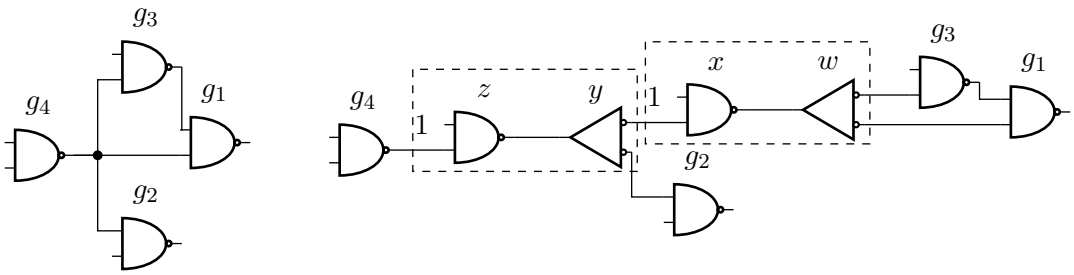
Circuit Model. The idea is to introduce a *doubling gadget* which has one input and two outputs. The semantics of these gadgets are that both output bits are the same as the



in	a	b
0	1	1
1	0	0

Figure 3.10: Schematic drawing and semantics of a doubling negation gate.

input bit. Using these gadgets we can replace a fork in a wire by a series of doubling gadgets, so that every output is input to at most one other gate. The doubling gadget consists of a NAND gate with one input fixed to 1 followed by a new *doubling negation* gate. The semantics of this gate is that both output bits are the negation of the input bit (see Fig. 3.10).



(a) part of a circuit with a 3-way fork (•) in a wire

(b) canonical form of the circuit part using two doubling gadgets (dashed boxes)

Figure 3.11: Replacement of a forking in a wire with two doubling gadgets consisting of NOT gates z and x and doubling negation gates y and w

Our circuits contain only 2-input NAND gates and doubling negation gates, every output of a gate is input to at most one other gate, and there are no doubling negation gates which are direct successors. We look at circuits in canonical form where we replace any fork with a series of doubling gadgets as described above. See Fig. 3.11 for an example of replacing a fork in a circuit. In this section we overload notation and call a circuit of this form to be in *canonical form*. For a circuit in canonical form, we again look at the corresponding directed acyclic graph $\overrightarrow{G}(\mathcal{C})$, where every input bit and gate is a node and the wires are the arcs. See Figure 3.12. We observe that every node

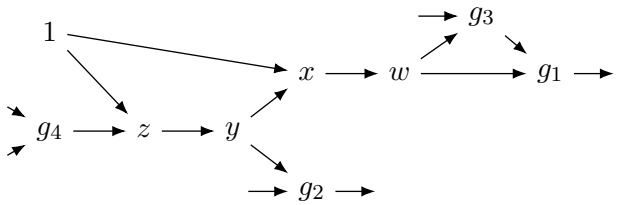


Figure 3.12: Part of the directed acyclic graph $\overrightarrow{G}(\mathcal{C})$ for the canonical form of the circuit part of Figure 3.11

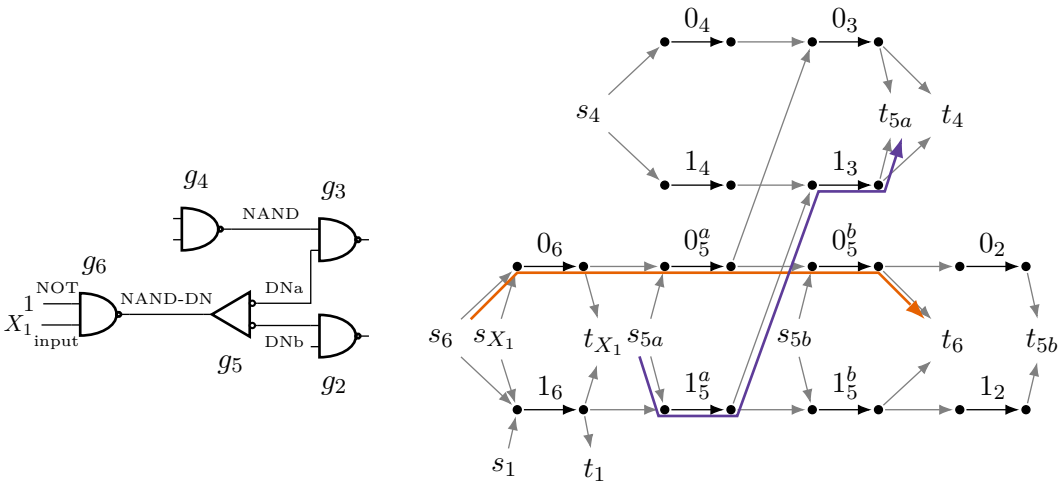
corresponding to a gate has either two ingoing and one outgoing arc (NAND gates), or one ingoing arc and two outgoing arcs (doubling negation gates).

Translation to Network Games. The translation of a circuit in canonical form to a network congestion game $\mathcal{N}_\mu^d(\mathcal{C})$ is similar to the construction of $\mathcal{G}_\mu^d(\mathcal{C})$. However, now we have a player for every *wire* in the circuit, i.e., every arc in the directed graph associated to the circuit.

For the formal definition of the network game associated to a circuit $\mathcal{C}$ in canonical form let $\mathcal{A}$ be the set of all NAND gates and $\mathcal{D}$ be the set of all doubling negation gates in $\mathcal{C}$. Consider the directed graph $\overrightarrow{G}(\mathcal{C})$ associated to $\mathcal{C}$. There are 7 types of arcs (u, v) in $\overrightarrow{G}(\mathcal{C})$:

- An *input arc* connects an input bit to some gate, i.e., u is an input bit of $\mathcal{C}$.
- A *NOT arc* connects the fixed input 1 to one of the NOT gates, i.e., $a \in \delta^+(1)$.
- The *output arc* corresponds to the output of the circuit, i.e., $u = g_1$.
- A *NAND arc* connects two NAND gates, i.e., $u, v \in \mathcal{A}$.
- A *NAND-DN arc* connects a NAND gate to a doubling negation gate, i.e., $u \in \mathcal{A}$ and $v \in \mathcal{D}$.
- A *DNa arc* connects the a-output of a doubling negation gate to some NAND gate.
- A *DNb arc* connects the b-output of a doubling negation gate to some NAND gate.

See Figure 3.13a for a part of a circuit containing 6 of the above types of arcs (no output arc).



(a) Part of a circuit in canonical form. Labels on the wires give the type of the corresponding arc.

(b) The part of the network related to the circuit part. The zero strategy of the NAND-DN arc is shown in orange and the one strategy of the DNa arc in purple. For simplicity the players p_a are identified by the head of a.

Figure 3.13: Translation of a part of a circuit to a network. (a) shows 6 of the arc types and (b) the construction of the network. Resource arcs are drawn black and zero arcs gray.

For every arc $a = (u, v)$ in $\overrightarrow{G}(\mathcal{C})$ we introduce a player p_a. The players related to input arcs are called *input players*. Additionally, there is an *output player* that is not

related to any arc but corresponds to the output of $\mathcal{C}$. One can think of adding a single outgoing arc to the last gate in $\overrightarrow{G}(\mathcal{C})$. Then the output player is the player associated to this arc.

In the directed graph for the network game, there are unique source nodes s_p and sink nodes t_p for every player p. The remaining nodes of the graph are the endpoints of *resource arcs* related to the gates of $\mathcal{C}$. For every NAND gate $g_k \in \mathcal{A}$ we introduce the two resource arcs 0_k and 1_k, and for every doubling negation gate $g_k \in \mathcal{D}$ we introduce four resource arcs $0_k^a, 0_k^b$ and $1_k^a, 1_k^b$. These will be the only arcs having some cost. All other arcs have cost zero and are used to connect the resource arcs and the sources and sinks of the players in the right way.

Every arc a in $\overrightarrow{G}(\mathcal{C})$ induces some arcs in our network connecting the source and sink of player p_a to the resource arcs. Table 3.1 shows which arcs are added depending on the type of a. Every $\to$ corresponds to a new arc. By $u \to v$ we mean the arc going from (the head of) u to (the tail of) v. For example for an input arc $a = (x_i, g_k)$ the two arcs added in the first path are $(s_{p_a}, \text{tail}(0_k))$ and $(\text{head}(0_k), t_{p_a})$. The arcs added in

type of a	paths	type of a	paths
input arc $a = (x_i, g_k)$	$s_{p_a} \to 0_k \to t_{p_a}$ $s_{p_a} \to 1_k \to t_{p_a}$	NAND arc $a = (g_k, g_{k'})$	$s_{p_a} \to 0_k \to 0_{k'} \to t_{p_a}$ $s_{p_a} \to 1_k \to 1_{k'} \to t_{p_a}$
NOT arc $a = (1, g_k)$	$s_{p_a} \to 1_k \to t_{p_a}$	NAND-DN arc $a = (g_k, d_{k'})$	$s_{p_a} \to 0_k \to 0^a_{k'} \to 0^b_{k'} \to t_{p_a}$ $s_{p_a} \to 1_k \to 1^a_{k'} \to 1^b_{k'} \to t_{p_a}$
output arc	$s_{p_a} \to 0_1 \to t_{p_a}$ $s_{p_a} \to 1_1 \to t_{p_a}$	DNa arc $a = (d_k, g_{k'})$	$s_{p_a} \to 0^a_k \to 0_{k'} \to t_{p_a}$ $s_{p_a} \to 1^a_k \to 1_{k'} \to t_{p_a}$
		DNb arc $a = (d_k, g_{k'})$	$s_{p_a} \to 0^b_k \to 0_{k'} \to t_{p_a}$ $s_{p_a} \to 1^b_k \to 1_{k'} \to t_{p_a}$

Table 3.1: The paths defining the directed graph for $\mathcal{N}_\mu^d(\mathcal{C})$.

this process are called *zero arcs*. They all have cost zero. See Fig. 3.13 for an example of the network game construction for a small part of a circuit.

To define the costs on the resource arcs, we again index the gates in reverse topological ordering, so that the output gate is g_1 and the gates connected to the input bits have high index.

The parameters of the network game $\mathcal{N}_\mu^d(\mathcal{C})$ are

$$d \in \mathbb{N}_{\geq 1} \qquad \text{and} \qquad \mu \in \mathbb{R}_{\geq 0} \quad \text{where} \quad \mu > 3^{2d+2}. \tag{3.27}$$

For the resource arcs 0_k and 1_k, we define the cost as before for $\mathcal{G}_\mu^d(\mathcal{C})$ (see (3.9)) by

$$c_{1_k}(x) = \mu^k x^d \qquad \text{and} \qquad c_{0_k}(x) = \lambda \mu^k x^d, \text{ where } \lambda = 3^{d/2}.$$

For the new resource arcs corresponding to the doubling negation gates we set

$$c_{0_k^a}(x) = c_{0_k^b}(x) = c_{1_k^a}(x) = c_{1_k^b}(x) = \frac{1}{2}\mu^k x^d.$$

All other arcs have cost zero. This finishes the description of the directed graph and the network game $\mathcal{N}_\mu^d(\mathcal{C})$. The strategies of a player p are all $s_p \to t_p$ paths in the graph.

Properties of the Network Game. We show that the behavior of $\mathcal{N}_\mu^d(\mathcal{C})$ is the same as for $\mathcal{G}_\mu^d(\mathcal{C})$. In particular, in any α-equilibrium of $\mathcal{N}_\mu^d(\mathcal{C})$ the players emulate the computation of the gates in the circuit.

We begin with two observations on the structure of the paths for the players that follow directly from the definition of the graph.

Observation 2. For every player p_a the set of $s_{p_a} \to t_{p_a}$ paths contains exactly the paths given in Table 3.1.

We will thus refer to the two $s_p \to t_p$ paths as *zero* and *one strategy* of player p. From the structure of these $s_p \to t_p$ paths we immediately get

Observation 3.

- Every player uses resource arcs of at most 2 gates (the gates she connects in $\overrightarrow{G}(\mathcal{C})$).
- Every resource arc of a NAND gate can be used by at most 3 players (the players associated to the ingoing and the two outgoing arcs in $\overrightarrow{G}(\mathcal{C})$).
- Every resource arc of a doubling negation gate can be used by at most 2 players (the players associated to the ingoing and the corresponding outgoing arc in $\overrightarrow{G}(\mathcal{C})$).

With these properties we are able to show that Lemma 4 also holds in $\mathcal{N}_\mu^d(\mathcal{C})$. That is, for small enough α, the α-equilibria of $\mathcal{N}_\mu^d(\mathcal{C})$ are exactly the profiles where the players emulate the computation of $\mathcal{C}$. As before we define a small cut-off

$$\varepsilon'(\mu) = \frac{3^{d+d/2}}{\mu}.$$

Then by the choices of $d \geq 1$ and $\mu > 3^{2d+2}$ (see (3.27)), we have $3^{d/2} - \varepsilon'(\mu) > 1$ as before.

Lemma 8. *(cf. Lemma 4) For any circuit $\mathcal{C}$ in canonical form and valid choices of parameters d and μ (satisfying (3.27)) consider the network game $\mathcal{N}_\mu^d(\mathcal{C})$. Let σ_X be a strategy profile for the input players and let $x \in \{0,1\}^n$ be the corresponding bit vector.*

Then the profile σ_X can be extended to a unique α-equilibrium σ of $\mathcal{N}_\mu^d(\mathcal{C})$ for any $\alpha \in [1, 3^{d/2} - \varepsilon'(\mu))$. Further, in σ the three players associated to a gate follow the semantics of the gate. In particular, the output player plays according to $\mathcal{C}(x)$.

Proof. Let $\mathcal{C}$ be a circuit in canonical form and take valid choices of d, μ, and α. We proceed as in the proof of Lemma 4. We first fix the input players to their strategies in σ_X and show that the players related to a gate follow the corresponding semantics. As the strategies and incurred costs for the input players are the same as in $\mathcal{G}_\mu^d(\mathcal{C})$, the same proof as in Lemma 4 shows that the input players do not have an incentive to change their strategies in a profile where the players emulate the computation of the circuit.

Let σ be an α-equilibrium of $\mathcal{N}_\mu^d(\mathcal{C})$ where the input players play according to σ_X. Consider a *doubling negation gate* $g_k \in \mathcal{D}$ and the three associated players: P_{in} corresponding to the ingoing NAND-DN arc, P_a corresponding to the outgoing DNa arc, and

P_b corresponding to the outgoing DNb arc. We want to show that if P_{in} plays her one strategy, then it is α-improving for both P_a and P_b to play their zero strategies and analogously for the zero strategy. Due to symmetries consider only P_a. This player is related to an arc connecting g_k to some NAND gate $g_l \in \mathcal{A}$, where $k > l$.

Case 1: P_{in} plays her one strategy. The cost of the one strategy of P_a is at least $c_{1_k^a}(2)$ since P_{in} and P_a use the resource arc 1_k^a. On the other hand, the cost of her zero strategy is at most $c_{0_k^a}(1) + c_{0_l}(3)$ as she is alone on resource arc 0_a^k but potentially all of the three players related to the NAND gate g_l are playing their zero strategy. Hence, the ratio of both costs is at least

$$\frac{c_{1_k^a}(2)}{c_{0_k^a}(1) + c_{0_l}(3)} = \frac{\frac{1}{2}\mu^k 2^d}{\frac{1}{2}\mu^k + \lambda\mu^l 3^d} = \frac{2^{d-1}}{\frac{1}{2} + \mu^{l-k}\lambda 3^d} > \frac{2^{d-1}}{\frac{1}{2} + \frac{1}{\mu}\lambda 3^d} > \frac{2^{d-1}}{\frac{1}{2} + \frac{1}{3^{d/2+2}}},$$

where the last inequality follows from the choice of $\mu > 3^{2d+2}$. From $d \geq 1$, we further have $2^d > 3^{d/2} + \frac{2}{3^2}$. Thus, we get that the ratio is strictly larger than $3^{d/2}$. This shows that P_a plays her zero strategy in the α-equilibrium σ.

Case 2: P_{in} plays her zero strategy. By similar arguments to the previous case, we have that the cost of the zero strategy of P_a is at least $c_{0_k^a}(2)$ and the cost of her one strategy is at most $c_{1_k^a}(1) + c_{1_l}(3)$ As $c_{0_l}(3) > c_{1_l}(3)$, the ratio of the costs is at least

$$\frac{c_{0_k^a}(2)}{c_{1_k^a}(1) + c_{0_l}(3)} = \frac{\frac{1}{2}\mu^k 2^d}{\frac{1}{2}\mu^k + \lambda\mu^l 3^d} > 3^{d/2}$$

exactly as in the previous case. Thus P_a plays her one strategy in the α-equilibrium σ.

Now consider a *NAND gate* $g_k \in \mathcal{A}$ and the three associated players: P_a and P_b corresponding to the two ingoing arcs, and P_{out} corresponding to the outgoing arc. We want to show that it is an α-improving move for P_{out} to play her zero strategy if and only if both P_a and P_b play their one strategy.

Let g_l be the successor of g_k in $\overrightarrow{G}(\mathcal{C})$. Recall that $k > l$. We consider three cases: either $g_l \in \mathcal{A}$, or $g_l \in \mathcal{N}$, or $k = 1$, that is, P_{out} is the output player of the circuit. We begin with $g_l \in \mathcal{A}$.

Case 1: Both P_a and P_b play their one strategy. The cost incurred to P_{out} on her one strategy is at least $c_{1_k}(3)$, since all three players P_a, P_b, and P_{out} use arc 1_k. On the other hand, the cost for her zero strategy is at most $c_{0_k}(1) + c_{0_l}(3)$. The ratio of the costs is thus at least

$$\frac{c_{1_k}(3)}{c_{0_k}(1) + c_{0_l}(3)} = \frac{3^d}{\lambda}\left(\frac{1}{1 + \mu^{l-k}3^d}\right) > \frac{3^d}{\lambda}\left(1 - \frac{1}{\mu}3^d\right).$$

By the definitions of λ and $\varepsilon'(\mu)$ the last term is equal to $3^{d/2} - \varepsilon'(\mu)$ and hence the ratio is strictly larger then α. Thus P_{out} plays her zero strategy in the α-equilibrium σ.

Case 2: At least one of P_a or P_b is playing her zero strategy. The cost incurred to P_{out} on her zero strategy is at least $c_{0_k}(2)$ and the cost of her one strategy is at most $c_{1_k}(2) + c_{1_l}(3)$. The ratio of these costs is thus at least

$$\frac{c_{0_k}(2)}{c_{1_k}(2) + c_{1_l}(3)} = \lambda\left(\frac{1}{1 + \mu^{l-k}(\frac{3}{2})^d}\right) > \lambda\left(1 - \frac{1}{\mu}3^d\right).$$

As in the previous case the last term is equal to $3^{d/2} - \varepsilon'(\mu)$ and hence strictly larger than α. Thus, P_{out} plays her one strategy in the α-equilibrium σ.

Now assume that $g_l \in \mathcal{D}$. Note that in the above computations only the second term in the upper bounds on the strategies changes. Instead of using arc 0_l (possibly together with at most two other players), P_{out} is now using arcs 0_l^a and 0_l^b (possibly together with at most one other player each). Observe that for any index l we have the relations

$$c_{0_l^a}(2) + c_{0_l^b}(2) = c_{1_l^a}(2) + c_{1_l^b}(2) = 2 \cdot \frac{1}{2}\mu^l 2^d = \mu^l 2^d < c_{1_l}(3) = \mu^l 3^d < c_{0_l}(3).$$

Thus, the ratio of the costs for P_{out} can be bounded as before.

Finally, if P_{out} is the output player of the circuit, she only plays the resource arc 0_1 or 1_1. The ratios of her costs are then at least

$$\frac{c_{1_1}(3)}{c_{0_1}(1)} = \frac{3^d}{\lambda} \qquad \text{and} \qquad \frac{c_{0_1}(2)}{c_{1_1}(2)} = \lambda.$$

By the definition of λ those are both strictly larger than α. Thus, also the output player follows the NAND semantics.

This shows that in any α-equilibrium where the input players are fixed, the other players emulate the computation of the circuit. The same proof as in Lemma 4 shows that the input players do not have an incentive to deviate. Thus there is a unique α-equilibrium as in the statement of the theorem. □

With this lemma, Theorem 2 also holds for network games. As the costs incurred to the input players and the output player is the same in $\mathcal{G}_\mu^d(\mathcal{C})$ and $\mathcal{N}_\mu^d(\mathcal{C})$, we can use almost the same proof for the network version. We choose $\mu > \frac{3^{2d+2}}{\varepsilon}$ such that $\varepsilon'(\mu) \leq \varepsilon$. As before we can choose a rational λ such that

$$\alpha\left(1 + \frac{1}{\mu}3^d\right) < \lambda < \frac{3^d}{\alpha\left(1 + \frac{1}{\mu}3^d\right)}$$

without changing the key inequalities above.

For the first and third problem, the same proof can be used, replacing the application of Lemma 4 by Lemma 8. For the second problem, the addition of the new resource of cost zero to the one strategy of the output player can be modeled in the network game. Instead of adding the path $s_{p_{\text{out}}} \to 1_1 \to t_{p_{\text{out}}}$ for the output player (see Table 3.1), we introduce a new dummy arc corresponding to the new resource and add the path $s_{p_{\text{out}}} \to 1_1 \to \text{dummy} \to t_{p_{\text{out}}}$.

Non-Existence Gadgets First, consider the games $\mathcal{G}^d_{(n,k,w,a)}$ from Section 3.3 with polynomial cost functions and no α-equilibrium for $\alpha < \alpha(d)$. We propose a network structure as shown in Figure 3.14.

For this construction to work, we need a gadget for the orange and purple parts. The purple gadget will be a flipped version of the orange gadget. This gadget has to have the following properties.

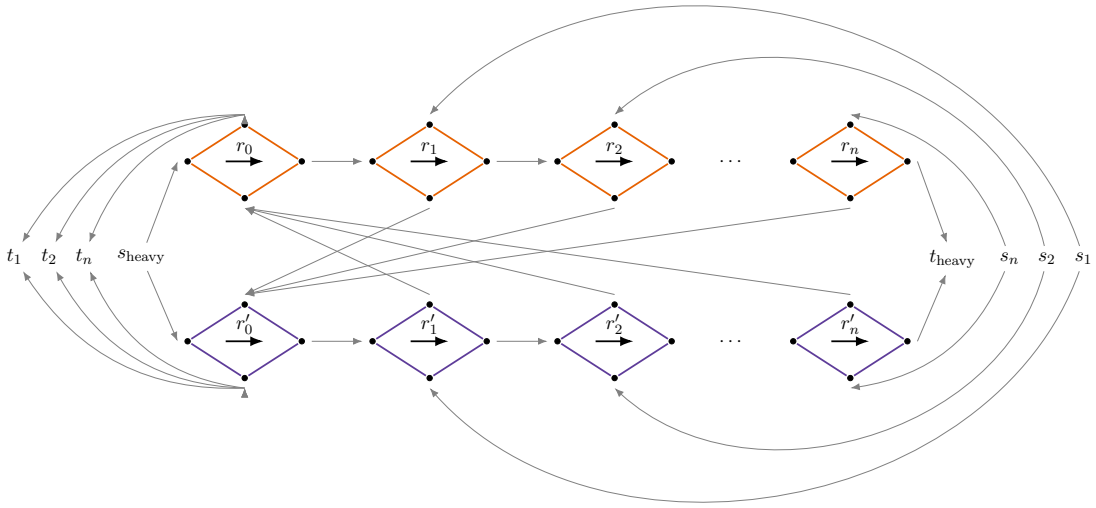


Figure 3.14: The structure of the network version of the game $\mathcal{G}^d_{(n,k,w,a)}$. The network uses gadgets (orange, purple) containing the resource arcs with the costs as in $\mathcal{G}^d_{(n,k,w,a)}$. All other arcs (gray) have cost zero.

- There are exactly 4 entry points: left, top, right, and bottom.
- A path that enters the gadget at the top has to leave the gadget at the bottom.
- A path that enters the gadget at the left has to leave the gadget at the right.
- The top-bottom and left-right paths share the resource arc in the gadget.

With such a gadget, there would be exactly two $s_i \to t_i$ paths in the network, corresponding to the strategies in $\mathcal{G}_n$. We were not able to find such a gadget. The key difficulty is to reproduce the congestion on the resource arcs without introducing more available $s_i \to t_i$ paths in the network.

Another approach would be to replace the gadgets by just the resource arcs (identify the left and top entry point as single node and similarly for right and bottom) and to put some cost on the arcs connecting the gadgets. To make sure that the two desired $s_i \to t_i$ paths for every player are α-dominating in any profile. Here the key difficulties are that we do not have the nice structure of the strategies as before in $\mathcal{G}^d_{(n,k,w,a)}$. For example, previously we had that if the heavy player is playing r'_0 she is not using r_0. However, in the network there is an $s_{\text{heavy}} \to t_{\text{heavy}}$ path using both of the resources. Further, previously resource r_i could only be used by light player i. However, in the network there are $s_j \to t_j$ paths using the resource arc r_i for some $j \neq i$. We were not able to find polynomial cost functions for the connecting arcs satisfying all of the above properties.

Similar problems arise for the games $\mathcal{G}_n$ with *general* cost functions from Section 3.6. It is thus left open as

Research Question 2. Can the games $\mathcal{G}^d_{(n,k,w,a)}$ and $\mathcal{G}_n$ be modeled as network games?

Combination of the Hardness Gadget and the Non-Existence Gadget As for the combination of both parts in our hardness reductions, we do not see a way to model the combination for polynomial congestion games. Recall that the combination is done by letting the output player of $\mathcal{G}^d_\mu(\mathcal{C})$ play all of the resources from the non-existence game in her one strategy (see the definition of $\mathcal{G}^d_\mu(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$ in Section 3.5). This would mean that we have to introduce a path in the network non-existence gadget containing all of the resources. This will introduce new paths for the players of the non-existence part and thereby affect the behavior of this gadget, even if the output player is not playing this strategy. Further, our transformations from Section 3.5 to the canonical structure are not applicable to network games. For example, when splitting a resource with constant cost into player specific resources with monomial cost, we can not guarantee anymore that each player uses only his own resource.

On the other hand, the combination for general costs can be done. Assume that there is a network version of $\mathcal{G}_n$ from Section 3.6. We adapt the network version of the circuit part $\mathcal{N}^d_\mu(\mathcal{C})$. Add a new arc related to the dummy resource. Add this arc to the path corresponding to the one strategy of the output player, i.e., add $s_{p_{\text{out}}} \to 1_1 \to \text{dummy} \to t_{p_{\text{out}}}$ instead of $s_{p_{\text{out}}} \to 1_1 \to t_{p_{\text{out}}}$. Further, for every player p of the non-existence game add an additional $s_p \to \text{dummy} \to t_p$ path. This does not change the non-existence part. Also the circuit part is unchanged, except for the additional dummy arc for the output player. Hence, the same arguments as in Section 3.6 can be used for this combination.

3.8 Conclusion

In this chapter, we construct weighted congestion games with polynomial cost functions that do not have α-equilibria for $\alpha < \alpha(d) \in \Omega\left(\frac{\sqrt{d}}{\ln d}\right)$. For general cost functions, we show that n player games always have n-equilibria, but there are games that do not have α-equilibria for any $\alpha < \varphi_{n-1} \in \Theta(\frac{n}{\ln n})$. Further, we develop a translation of a Boolean circuit to an unweighted polynomial congestion game. With this gadget, we are able to show NP-hardness of deciding the existence of α-equilibria for α as bounded above, and the existence of α-equilibria with additional properties.

Our results leave open the question of where exactly the non-existence begins. For polynomials of degree d we know from Caragiannis and Fanelli [CF21] that d-equilibria always exist. In Theorem 3, we show that $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$-equilibria do not necessarily exist. Similarly, for general cost functions, we show that n-equilibria exist, but $\Theta(\frac{n}{\ln n})$-equilibria do not necessarily exist. Closing these gaps is one of the obvious future research directions.

Research Question 3. What is the smallest α such that all weighted (polynomial) congestion games have an α-equilibrium?

In Section 3.7, we show that the hardness gadget can also be transformed into a network game. However, we do not have such translations for our non-existence gadgets (Research Question 2). It would be interesting to see whether the same bounds hold for network games.

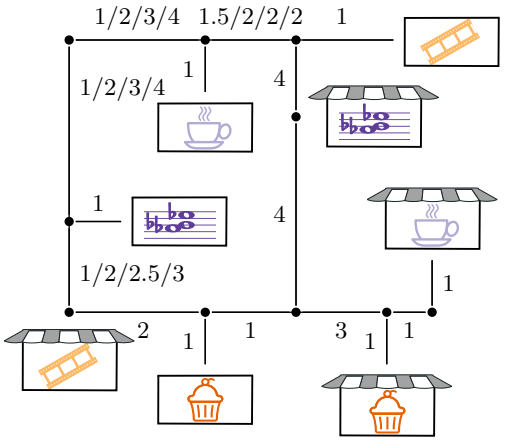
Another related area, which we did not touch on in this chapter, is the complexity of *computing* α-equilibria in the cases where they are guaranteed to exist. For *polynomial* games of degree d, Caragiannis and Fanelli [CF21] show that d-improving moves converge to a d-equilibrium. Hence, computing a d-equilibrium is a problem in PLS. The only known hardness result is the PLS-completeness of finding *exact* equilibria in *unweighted* games by Fabrikant, Papadimitriou, and Talwar [FPT04] and Ackermann, Röglin, and Vöcking [ARV08]. On the other hand, $d^{\mathcal{O}(d)}$-equilibria can be computed in polynomial time (see for example [Car+15; Fel+17; GNS21]). For *general* cost functions, Skopalik and Vöcking [SV08] show that it is PLS-complete to compute an α-equilibrium for any fixed α.

4 Network Design Games with Economies of Scale

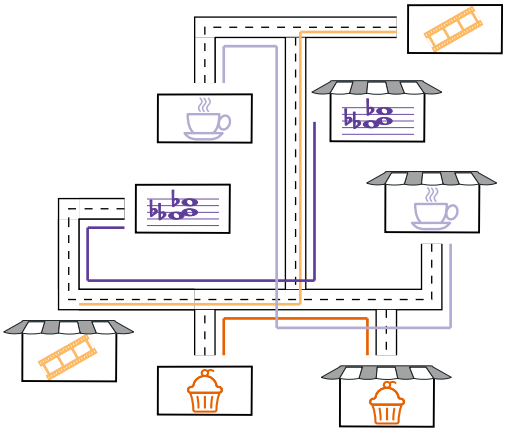
4.1 Introduction

In contrast to (negative) congestion effects on shared resources as in the previous chapter, there are situations where using the same resource is beneficial for the participants. Consider a small town which is about to be extended by a new district. Some companies were assigned a local storehouse and a shop each to make the new neighborhood attractive. The missing piece is the construction of the new road network.

A map of all possible roads has been published together with the respective construction costs. Each company wants to connect its storehouse to its shop in the district and is paying for the roads needed to be built. The construction cost of each road depends on the number of companies using this road. For a single company, a country road is sufficient, but for 10 companies a proper motorway has to be built. Hence, the construction cost is increasing. Further, extending a motorway by an additional lane is



(a) Map of possible roads. The edge labels give the construction cost for the street. The label $a/b/c/d$ means that for 1 company the cost is a, for 2 companies the cost is b and so on. A label a means that the construction cost is a independent of the number of companies.



(b) A realization of a road network connecting all stores to their corresponding shop. The streets built by the companies are highlighted in the respective color.

Figure 4.1: Example of the planning of a new district. There are 4 companies: a cinema, the coffeeORtea club, a cupcake shop, and the chord company with a shop and a storehouse each.

cheaper than upgrading a country road to a motorway. Hence, there are economies of scale effects in place, and the construction cost of each road is concave. Since anyone can use a road, once it is built, the construction cost of a road is shared by all companies wanting to build this road. To incentivize the companies to share roads to protect the environment, the share for each company is decreasing in the number of companies using a road. Figure 4.1 shows an example of the situation.

In the process of deciding which roads to build for which company, there are two competing interests to consider. On the one hand, there is the overall social cost of the roads to be built; on the other hand, there are the individual costs incurred by each company. While the local authorities want to minimize the social cost of the new road network, each company wants to minimize its own cost. The companies will thus propose a stable road network, where no company is better off by building a different path of roads given the choices of the other companies.

Example 12. Figure 4.1b shows a road network of minimal social cost 23.5. This network is not stable as the cinema can decrease its cost from $1 + \frac{4}{2} + \frac{4}{3} + \frac{1}{3} + \frac{2}{2} = 5 + \frac{2}{3}$ to $1 + \frac{2}{2} + 1 + 1 + \frac{2}{2} = 5$ by building the other path using the left upper corner. ⌟

The local authorities' only interest is the social cost, ignoring the stability of the proposed network. The companies' only interest is the stability of the proposed network, ignoring the social cost. To satisfy both sides, the goal is to find a stable network of minimal social cost and show that the social cost of this network is not too far away from the minimal social cost achievable. The quality of such stable networks and how to find them is the topic of this chapter.

Example 13. The network resulting from the one shown in Figure 4.1b where both the cinema and the coffeeORtea club build the other path (using the left upper corner) is the only stable state with social cost of 25. In this example, the gap between a stable state and a state with minimal social cost is thus $\frac{25}{23.5} \approx 1.06$ ⌟

The task of the local authorities in the above setting is the *network design problem* with economies of scale as introduced by Salman et al. [Sal+97]. Taking the viewpoint of the companies puts the problem in the scope of game theory. The game described above, where the companies are the players, can be formulated as a congestion game. In particular, a network congestion game with *decreasing* per-player cost functions. The stable states are the pure Nash equilibria (equilibria) in these games. To measure the quality of equilibria, the two notions of *Price of Anarchy (PoA)* ([Pap01]) and *Price of Stability (PoS)* ([Ans+08a]) have been introduced as the worst and best ratio of the social cost of an equilibrium and the overall minimal social cost.

Anshelevich et al. [Ans+08a] were the first to study the quality of equilibria in network congestion games with *decreasing* cost functions. Most of the subsequent literature focuses mainly on the case where the total construction cost of each road (the cost shared by all players) is *constant* (and hence independent of the number of players). Note that in this case, the task of the local authorities is to find a minimum Steiner forest w.r.t. the construction costs.

In this chapter, we extend the study of equilibria as in [Ans+08a] to the case of general non-decreasing concave total construction costs with decreasing per-player costs as in the network design problem. We thus call these games *network design games with economies of scale*.

An instance of such a game is given by an undirected graph and a set of players. Every player wants to connect her source and sink node by a path in the graph. Once all players chose a path, the cost incurred to each player is computed by the sum of the costs of the edges used on their respective paths. Each edge in the graph has a cost function, which gives the *per-player cost* incurred to every player using the edge depending on the total number of players using the edge. If, for example, 3 players use edge e with per-player cost function f on their path, then each of the three players has to pay $f(3)$ for this edge. We thus get a *total cost* of $3 \cdot f(3)$ paid for edge e. We assume the per-player costs to be decreasing and at the same time the total costs to be non-decreasing and concave (with non-increasing marginal costs). Such functions will be called *sharing functions with economies of scale*. Two extreme cases of sharing functions with economies of scale are $f(k) = \gamma$ and $f(k) = \frac{\gamma}{k}$, where $\gamma > 0$ is some constant. Functions of the second form fall into the model of *fair cost allocation* studied in [Ans+08a] and most of the subsequent works. If the per-player cost function is the same up to some constant scaling factor on all edges, then we speak of a *uniform* game; otherwise, the game is called *non-uniform*.

For a choice of player paths, the sum of all player costs is called the *social cost*. Two kinds of collections of player paths are of interest: *equilibria* (collections where no player can reduce her cost by switching to another path) and *social optima* (collections of minimal social cost). We are especially interested in *broadcast* network design games with economies of scale where all players have a common sink node, and there is a player associated to each node of the graph, which is the source node of the player.

We study the Price of Stability (PoS) of network design games with economies of scale from two perspectives. Firstly, we are interested in the value of the PoS, that is, how good can equilibria be in terms of their social cost. Secondly, we investigate how those good equilibria can be found efficiently.

Note on Collaboration This chapter is based on joint work with Yiannis Giannakopoulos and Marcus Kaiser. The results presented here also appear in the PhD thesis of Marcus Kaiser.

Previous Work

Network Design Games. The model of network design games we use here is the one from Anshelevich, Dasgupta, Kleinberg, Tardos, Wexler, and Roughgarden [Ans+04; Ans+08a]. Every player wants to connect their source node to their destination node and pays for all edges on the path. The total cost of an edge is shared equally by players using the edge. Anshelevich et al. [Ans+08a] focus on the case where the total edge cost is constant, which they call *fair cost allocation*. They also mention that some of their results extend to non-decreasing and concave total edge costs with decreasing per-player

costs. Such cost functions are the main focus of our work. The same model has also been studied by Syrgkanis [Syr10].

There are many variants of connection games in networks that differ in the set of edges players pay for and how they share the cost of used edges. We mention only two well-studied variants.

Anshelevich, Dasgupta, Tardos, and Wexler [Ans+03; Ans+08b] introduce a quite general model where players want to connect their terminal nodes (possibly more than two) by buying edges. For every edge used by a player, she offers a cost-share she is willing to pay. Once an edge is bought (if all cost shares cover the total edge cost), all players can use the edge, even if they did not pay for it. This models the unregulated and unrestricted formation of a new communication network.

In contrast to this global perspective of the players, Bala and Goyal [BG03] and Fabrikant, Luthra, Maneva, Papadimitriou, and Shenker [Fab+03] define a local model, where players pay only for some of the incident edges of the source node. The cost incurred to a player is the sum of the edge cost paid and the distance in the resulting graph to every other node (in terms of number of edges). This models the formation of a social network, where users connect to some friends and get thereby connected to friends of friends.

Extensions of the model of Anshelevich et al. [Ans+08a] include giving weights to players (see for example Anshelevich et al. [Ans+08a] and Chen and Roughgarden [CR06; CR09]) or adding capacity constraints to the edges (see for example Feldman and Ron [FR12; FR15]).

Network Design Games with Fair Cost Allocation. The study of the quality of equilibria in network design games with fair cost allocation was initiated by Anshelevich et al. [Ans+08a]. They give an example of a directed multicast game, with a *Price of Anarchy* of n, the number of players. Their example (two parallel arcs) can be transformed into a broadcast instance even for undirected graphs. Hence in all settings (directed, undirected; general, multi-, broadcast), the PoA is n.

For the *Price of Stability*, they show an upper bound of $H(n)$ (the nth harmonic number) for both directed and undirected games with n players, which is asymptotically $\Theta(\log n)$. They give a tight example of a general network game that can easily be transformed into a broadcast game, which shows that the PoS for *directed* (broadcast) games is $H(n)$. The value of the PoS for *undirected* games is posed as an open problem. A series of papers have attacked this question from different angles without finding an answer until today. There are huge gaps between the best-known upper and lower bounds for all three cases of multi-source-sink, multicast, and broadcast games.

For *general* network games, the $H(n)$ upper bound from Anshelevich et al. [Ans+08a] is only slightly improved up to now. Disser, Feldmann, Klimm, and Mihalák [Dis+15] show an improved bound of $\left(1 - \Theta\left(\frac{1}{n^4}\right)\right)H(n)$ and Mamageishvili, Mihalák, and Montemezzani [MMM18] further reduce this to $H(\frac{n}{2})$. Asymptotically, the current bound is still $\mathcal{O}(\log n)$. The lower bound was increased from 1.714 by Fiat, Kaplan, Levy, Olonetsky, and Shabo [Fia+06] (a broadcast instance) to 1.826 by Christodoulou, Chung, Ligett, Pyrga, and Stee [Chr+09]. Bilò, Caragiannis, Fanelli, and Monaco [Bil+10; Bil+13]

construct a general game with the currently highest lower bound on the PoS of 2.245. Only for small numbers of players, there are better bounds on the PoS for general games. Christodoulou, Chung, Ligett, Pyrga, and Stee [Chr+09] show that the PoS for 2 player games is exactly $\frac{4}{3}$, and for 3 players, the PoS is in the interval $[1.542, 1.65]$. Disser, Feldmann, Klimm, and Mihalák [Dis+15] improve the lower bound to 1.571 and Bilò and Bove [BB11] improve the upper bound to 1.634.

The only improvement of the $H(n)$ upper bound of Anshelevich et al. [Ans+08a] for *multicast* games is given in Li [Li09] as $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$. Anshelevich et al. [Ans+08a] give an example of a multicast game with 2 players achieving a PoS of $\frac{4}{3}$, which they show to be the exact PoS for 2 player multicast games. For 3 players, Bilò and Bove [BB11] show that the PoS is in the interval $[1.524, 1.532]$. The currently best lower bound of 1.862 for any number of players is constructed in Bilò, Caragiannis, Fanelli, and Monaco [Bil+10; Bil+13]. They give a family of multicast games parameterized by the number of players. They show that the PoS of each instance is the solution of a linear program where the total edge costs are the variables.

Freeman, Haney, and Panigrahi [FHP16] consider *quasi-bipartite* games as intermediate setting between multi- and broadcast games. In a quasi-bipartite game, there are no edges between nodes that are not source nodes of players (i.e., every edge is incident to at least one source node). Extending the method of Bilò, Flammini, and Moscardelli [BFM13] from broadcast games to quasi-bipartite games, they show that the PoS in such games is constant (without specifying the exact value).

All mentioned upper bounds also hold for *broadcast* games. However, this special case allowed for a series of improvements culminating in a constant upper bound. The first improvement over the $H(n)$ bound from Anshelevich et al. [Ans+08a] is a bound of $\mathcal{O}(\log \log n)$ shown by Fiat, Kaplan, Levy, Olonetsky, and Shabo [Fia+06]. Lee and Ligett [LL13] further improve this bound to $\mathcal{O}(\log \log \log n)$. Finally, Bilò, Flammini, and Moscardelli [BFM13; BFM20] show that the PoS for broadcast games is bounded by a constant. Unfortunately, they do not give the value of their constant, but it is assumed to be very large. In contrast to that, the best lower bound so far is 1.81 from Bilò, Caragiannis, Fanelli, and Monaco [Bil+10; Bil+13]. Similar to the multicast case, this bound is shown by solving a linear program. Their instances are optimized versions of the instance for the first lower bound of 1.714 by Fiat, Kaplan, Levy, Olonetsky, and Shabo [Fia+06]. For 3 players Bilò and Bove [BB11] show that the PoS is 1.485 in broadcast games.

Cost of a Social Optimum. To specify the PoS of a network game exactly, we have to compute the cost of a social optimum of the instance. For bounds on the PoS, bounds on the social optimum are sufficient.

In *undirected* networks computing the cost of a social optimum is a special case of the (unsplittable) *buy-at-bulk network design* problem (see for example [Sal+97; AA97]), where the demand of every source-sink pair is 1. For network design games with fair cost allocation, the contribution of an edge to the social cost is independent of the number of players using the edge. Hence, a social optimum corresponds to a minimum Steiner forest connecting all source-sink pairs. In the broadcast case, this problem reduces to finding

a minimum spanning tree. Thus, buy-at-bulk network design is NP-hard ([Sal+97]) for all cases except broadcast instances.

Hence, many papers study algorithms to approximate the cost of an optimal network for different settings. The first approximation by Salman, Cheriyan, Ravi, and Subramanian [Sal+97] considers the *uniform* case. They study *single-sink* instances in the Euclidean plane and show a $\mathcal{O}(\log D)$ approximation where D is the total demand. Later, constant factor approximations have been shown for this setting ([GMM01; Tal02; GKR03; Gup+07; JR04; JR09; GI06]). The currently best constant achieved is 40.82 by Grandoni and Rothvoß [GR10].

For uniform *multi-sink* instances the first approximation by Awerbuch and Azar [AA97] gives a factor of $\mathcal{O}\left(\log^2 n\right)$, where n is the number of nodes in the network. Using the tree embedding of Fakcharoenphol, Rao, and Talwar [FRT03; FRT04] gives an improved factor of $\mathcal{O}(\log n)$.

In the *non-uniform* setting, there are approximations for the *single-sink* case. Meyerson, Munagala, and Plotkin [MMP00] give a randomized $\mathcal{O}(\log k)$-approximation algorithm, where k is the number of sources. Chekuri, Khanna, and Naor [CKN01] derandomized their algorithm, maintaining the approximation ratio. The best known approximation ratio for *multi-sink* instances is $\mathcal{O}\left(\log^4 k\right)$ by Chekuri, Hajiaghayi, Kortsarz, and Salavatipour [Che+10], where k is the number of source-sink pairs.

On the other hand, Andrews [And04] gives inapproximability results for uniform and non-uniform buy-at-bulk network design.

Finding Equilibria. Already in the first paper studying network design games, Anshelevich et al. [Ans+04] show that computing one of *the best* equilibria, an equilibrium of minimal social cost, is NP-complete for directed multicast games by a reduction from 3D-Matching. For undirected multicast games, Syrgkanis [Syr10] remarks that the NP-hardness can be shown similarly to an NP-hardness proof in Chekuri, Chuzhoy, Lewin-Eytan, Naor, and Orda [Che+06; Che+07]. Instead of finding the best equilibrium, Chekuri et al. consider the global potential minimizer.

Finding the global potential minimizer is an uncapacitated *minimum concave cost multicommodity flow* computation, where the edge costs are the contributions of the edge to the potential function. For multi- and broadcast instances, the problem simplifies to *single-source single-commodity minimum concave cost flow*. Both problems are NP-hard in general as they contain minimum Steiner forest for constant edge costs. For network design games, the potential is not constant, but Guisewite and Pardalos [GP91a] show that the problem remains NP-hard even for single-source minimum concave cost flow with strictly concave cost functions. Exact algorithms have been developed for restricted instances, see for example [EMV87; GP93; Tuy+95; War99; Tuy00]. We refer to Guisewite [Gui95] for a survey of the early results and algorithms. Recent works on the minimum concave cost flow problem have focused on better heuristics [FG07; BS07; MFF13; WKD20].

Since network design games are potential games, the equilibria are exactly the local minimizers of the potential function, w.r.t. single player deviations. Even though the global minimizer of the potential is an equilibrium, Kawase and Makino [KM12; KM13]

show that there are instances of undirected broadcast network design games, where the ratio of the social cost of the global minimizer to the social optimum is $\Omega(\sqrt{\log \log n})$. Since the PoS for undirected broadcast games is constant, this tells us that using the global potential minimizer as an approximation to a best equilibrium is not a good idea.

From these hardness results on finding the best equilibrium, the question arises whether it is easier to find just *some* equilibrium. The task of computing an equilibrium of an instance of a network design game can be formulated as a local search problem, where the neighbors are profiles reachable by single-player deviations. The standard algorithm is the improving-dynamics.

There are local search algorithms for minimum concave cost flow problems (for example [GP91a; GP91b]), but we emphasize here that their concepts of neighborhoods are often geometric and do not compare to single-player deviations. In particular, the notion of adjacent flow defined by rerouting a subpath of flow is not a single-player deviation but rather all players using the subpath changing their strategies. Further, to the best of our knowledge, there are no theoretical results on the quality of reached local minima.

Johnson, Papadimitriou, and Yannakakis [JPY88] introduced the complexity class of polynomial-time local search problems (PLS) for local search problems. For general directed network design games with fair cost allocation, Anshelevich et al. [Ans+08a] give an example where the improving-dynamics takes exponential time in n, the number of players, by simulating a binary counter with n bits. Later, Syrgkanis [Syr10] shows that finding an equilibrium in general directed network design games is PLS-hard for classes of cost functions satisfying some properties. In particular, fair cost allocation satisfies those properties. He also gives a reduction for undirected network design games, where the cost functions have to satisfy some other set of properties. Here, fair cost allocation is not included. Only recently, Bilò, Flammini, Monaco, and Moscardelli [Bil+21] filled the gap showing PLS-hardness for undirected general network design games with fair cost allocation. The complexity for finding an equilibrium in multi- and broadcast games is still open for directed and undirected games and any set of cost functions.

Despite these hardness results, there might be efficient centralized algorithms to find an equilibrium. To the best of our knowledge, no such algorithms have been developed for network design games.

Our Results In this chapter, we study the Price of Stability (PoS) of uniform undirected broadcast network design games with concave total edge costs and decreasing per-player costs. We consider three classes of per-player cost functions characterized by their total cost. Functions of the first type have *constant* total cost, and thus, they represent fair cost allocation as in [Ans+08a]. The two remaining classes are interpolations between the two extreme cases of sharing functions with economies of scale with constant and linear (slope greater than 1) total costs. On the one hand, we have *linear* total cost functions where the slope is between 0 and 1. On the other hand, we consider *polynomial* total cost functions where the exponent is between 0 and 1. Note that fair cost allocation is an extreme case of both classes: linear functions with slope 0 and polynomial functions with exponent 0.

After stating our model formally in Section 4.2, we begin with some structural properties of equilibria in these games (Section 4.3).

Price of Anarchy. The first new result is an extension of the lower and upper bound on the Price of Anarchy (PoA) from [Ans+08a] from fair-cost allocation to sharing functions with economies of scale. In Section 4.4, we show that the PoA for uniform broadcast games with underlying cost function f is $\frac{f(1)}{f(n)}$.

Price of Stability. The main part (Sections 4.5 and 4.6) focuses on bounding the Price of Stability. In Section 4.5, we give *upper bounds* on the PoS for general games. We extend the method from [Ans+08a] using the potential function from fair cost allocation to sharing functions with economies of scale (Section 4.5.1). In particular, we show a *constant* upper bound on the PoS for all linear and polynomial functions except for fair cost allocation. In both cases, the constant gets large as the function approaches fair cost allocation.

In Section 4.5.2, we extend the homogenization-absorption framework from [BFM20] for broadcast games from fair cost allocation to sharing functions with economies of scale. We show a *constant* upper bound for quickly decreasing functions where two quantities (a supremum and a series) are bounded (Theorem 9). The bound on the PoS is given explicitly in dependence on the function values at 1 and 2, the two aforementioned quantities, and the absorption- and charging-radii. Theorem 9 applies to fair cost allocation, for which we give an upper bound of 264.25, improving upon the currently best known bound of [BFM20].

For uniform broadcast games, we then turn to *lower bounds* for the PoS in Section 4.6. We generalize the lower bound construction by Bilò et al. [Bil+13] for fair cost allocation to sharing functions with economies of scale. As in [Bil+13], we get lower bounds by considering a feasible solution to an LP. The variables are the scaling factors of the underlying cost function on each edge. For functions satisfying two properties, the constraints of the LP characterize instances where a special tree is the unique (and hence the best) equilibrium (Corollary 4). We show that all three classes of cost functions satisfy those properties. By computing the objective value for different values of the parameters (slope and exponent), we obtain *the first lower bounds* for linear and polynomial total cost functions (Sections 4.6.2 and 4.6.3). For fair cost allocation, we show (Theorem 10) that the weights computed in [Bil+13] determine the PoS in the subclass of instances considered in that paper.

These theoretical results are complemented with *computational experiments* (Section 4.6.5). Using methods of constraint programming, we enumerate small instances of up to 5 nodes (4 players). The experiments show that for fair cost allocation, instances with highest PoS are of the form as considered in [Bil+13]. To reduce the search space, we restrict to a subclass of instances. The experiments suggest that the instances studied in Sections 4.6.2 and 4.6.3 are optimal.

Computing Equilibria. Finally, we consider the question of computing (good) equilibria in Section 4.7. First (Section 4.7.1), we give an example of an improving move in a multicast game with fair cost allocation, where the decrease in the potential is very

small. It remains open, whether such moves can often appear in the improving-dynamics and hence, whether improving-dynamics may take exponential steps.

For general games, we extend the *PLS-hardness* proof of Bilò et al. [Bil+21] from fair cost allocation to sharing functions with economies of scale in Section 4.7.2. Syrgkanis [Syr10] already considers sharing functions with economies of scale and shows PLS-hardness for *non-uniform* games.

Additionally to this local search problem of the players, trying to find some equilibrium, we consider the task of computing the *best* equilibrium for sharing functions with economies of scale. We give a reduction (Section 4.7.3), showing that this is *NP-hard* for uniform multicast and broadcast games. The same reduction also shows that computing the global potential minimizer is NP-hard for broadcast games.

4.2 Model and Notation

In this chapter we are interested in network games on undirected graphs with decreasing edge costs. We use the game theoretic notation as introduced in Section 2.2. First, we state our model of network games with a special class of decreasing cost functions formally, before recalling some of the notation related to network games and introducing new notation.

Model We consider an (unweighted, undirected) *network game* as defined in Section 2.2. A game is given by an undirected graph $G = (V, E)$ and a set of players P with a corresponding source $s_p \in V$ and sink $t_p \in V$ for each player $p \in P$. The set of available *strategies* $S_p \subseteq 2^E$ for player p is the set of all $s_p - t_p$ paths in G.

Every edge has a *per-player cost* $c_e : \mathbb{N}_{\geq 1} \to \mathbb{R}_{\geq 0}$, that gives the cost incurred to each player using edge e on her path. This cost depends on the total number of players using e. If k players use edge e on their paths then each of them pays $c_e(k)$ for edge e and the *total cost* of e is $kc_e(k)$. Note that c_e is only defined on $\mathbb{N}_{\geq 1}$. We sometimes extend this to $\mathbb{N}$ by setting $c_e(0) = 0$. Another way of thinking of these costs is as follows. Every edge has a total cost depending on the number of players using the edge. This cost is shared equally by all players using the edge.

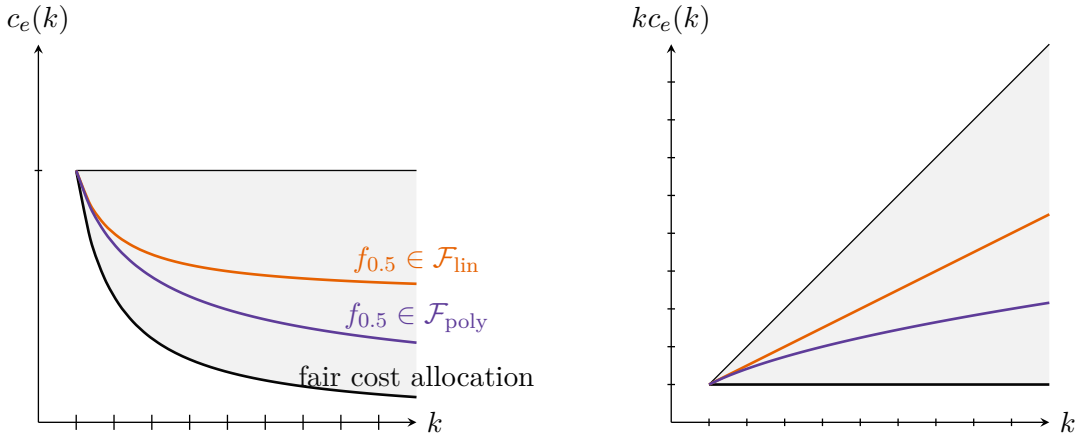
In this chapter we are looking at a special class of cost functions where sharing an edge is beneficial to the players, in contrast to the previous chapter, where players tried to avoid using the same edge. We have the following assumptions on the cost functions $c_e : \mathbb{N}_{\geq 1} \to \mathbb{R}_{\geq 0}$

- the per-player cost $c_e(k)$ is strictly decreasing in k or $c_e \equiv 0$ and
- the total cost $kc_e(k)$ is non-decreasing and $(k+1)c_e(k+1) - kc_e(k)$ is non-increasing in k.

The first condition models the benefit of sharing edges. If more players use the same edge, the cost for each player decreases. The second condition models economies of scale effects. The total cost for an edge is getting larger for more players, but the marginal increase gets smaller. By forcing c_e to be *strictly decreasing*, we exclude some decreasing functions. Together with the non-increasing marginal costs assumption, these functions

have the following structure: they are constant up to some point, and then strictly decreasing. In particular, constant per-player cost functions are *not* part of our model. For technical reasons we allow constant per-player costs of zero. Note that from the total cost being non-decreasing we have $c_e(k) \geq \frac{c_e(1)}{k} > 0$ if and only if $c_e(1) > 0$. Hence if some edge has per-player cost of 0 for some k, then it has to be 0 always. Thus, c_e is either always strictly positive or always zero.

We call a function satisfying the above two properties a *sharing function with economies of scale*. See Figure 4.2 for an illustration of such functions. Cost functions of this form have already been considered by Anshelevich et al. [Ans+08a] and Syrgkanis [Syr10]. Note that the main model of *fair cost allocation* studied in [Ans+08a] is a special case of our model, where the total cost of each edge is constant.



(a) Per-player cost functions are strictly decreasing in the shaded region.

(b) Total cost functions are non-decreasing with non-increasing marginal cost in the shaded region.

Figure 4.2: Feasible regions for sharing functions with economies of scale on the left together with a representative of every class of interest. On the right, the corresponding total cost functions.

We are looking at three classes of sharing functions with economies of scale described by their related total cost. First, we consider the case of *constant* total cost, where the per-player cost is $c_e(k) = \frac{\gamma_e}{k}$ for some constant $\gamma_e > 0$. As mentioned before this is the model of *fair cost allocation* from [Ans+08a]. Since the total cost is assumed to be non-decreasing this is one extreme case where c_e is decreasing as fast as possible. Another boundary case would be that c_e is constant, meaning that there is no sharing effect. Recall that constant c_e are not part of our model.

The two remaining classes are interpolations between these extremes. The class $\mathcal{F}_{\text{lin}}$ contains functions with *linear* total cost $kf_s(k) = 1 + s(k-1)$ for $s \in [0, 1)$. The per-player costs are of the form $f_s(k) = s + \frac{(1-s)}{k}$ and the marginal cost of the total cost is always s. These functions f_s are linear interpolations between $f(k) = \frac{1}{k}$ for $s = 0$ and $f \equiv 1$ for $s = 1$. The class $\mathcal{F}_{\text{poly}}$ contains functions, where the total cost is a *polynomial*

$kf_\alpha(k) = k^\alpha$ for $\alpha \in [0, 1)$. The per-player costs are polynomially decreasing functions of the form $f_\alpha(k) = k^{\alpha-1}$. For $\alpha = 0$ we obtain fair cost allocation and for $\alpha = 1$ we have the constant case. Since $0 \leq \alpha < 1$, f_α and kf_α are indeed decreasing and non-decreasing. Further, since k^α is concave, the marginal cost is non-increasing in k. Figure 4.2 shows the two functions $f_{0.5}$ in $\mathcal{F}_{\text{lin}}$ (orange) and $\mathcal{F}_{\text{poly}}$ (purple).

We consider two cases of network games with decreasing cost functions. A *non-uniform network game* is given by a set of available functions $\mathcal{F}$ satisfying our assumptions and a network game as described above, where every cost function c_e is one of the given functions, i.e., $c_e \in \mathcal{F}$ for every edge e. Such game is denoted by (N, c), N is a network game and c is the vector of cost functions $c_e \in \mathcal{F}$ on the edges. In the *uniform* case we specify a single sharing function with economies of scale $f : \mathbb{N}_{\geq 1} \to \mathbb{R}_{>0}$ and every cost function of the network game has to be a scaled version of f, i.e., for every edge e there is some *scaling factor* $\gamma_e \geq 0$ such that $c_e(k) = \gamma_e f(k)$. We refer to f as the *underlying function* of the game. Note that for uniform games we force f to be strictly positive everywhere (except at 0). Edge costs of zero can be modeled by setting $\gamma_e = 0$. A *uniform network game* will be denoted by (N, f, γ), where N is a network game, f is the underlying function, and γ is the vector of the scaling factors γ_e for all edges. By setting $\mathcal{F} = \{k \mapsto \gamma f(k) : \gamma \in \mathbb{R}_{\geq 0}\}$ we see that uniform games are special cases of non-uniform games. We will mostly look at the uniform case here.

Network Games We introduce additional notation to the one of Section 2.2. In this chapter, we denote by $(V, E, \{(s_p, t_p) : p \in P\}, c)$ an unweighted undirected network game, where (V, E) is the undirected graph, the third entry is the set of the source-sink pairs for every player, and c is a vector of the cost functions c_e for every edge. For a strategy profile σ we say player p *uses* edge e in σ if $e \in \sigma_p$ and p *goes through* v if v is one of the nodes lying on the path σ_p. We denote by $\text{supp}(\sigma)$ the *support* of σ which is the set that consists of all the edges being used by at least one player. Sometimes the support of a profile will be a tree in which case there is a correspondence between the tree as a graph and the profile. If we are given a profile we can construct the tree by taking the edges being used. On the other hand, if we are given the tree there is exactly one path for every pair of nodes in the tree, which defines the strategies of the players. In some cases we use both terms interchangeably.

Note that we are looking at unweighted network games, and thus the congestion on edge $e \in E$ is $n_\sigma(e) = |\{p \in P : e \in \sigma_p\}|$, the number of players using e in σ. The *cost of edge* e in a profile σ is $c_\sigma(e) = c_e(n_\sigma(e))$, and the *cost of player* p is $C_p(\sigma) = \sum_{e \in \sigma_p} c_\sigma(e)$. We extend the cost of an edge to the cost of a set of edges $F \subseteq E$ naturally to $c_\sigma(F) = \sum_{e \in F} c_\sigma(e)$. Since we are often interested in single player deviations, it is useful to introduce the short notation $c_\sigma^{+1}(e)$ for the cost on an edge e if there is one extra player using edge e additionally to the players using e in σ, that is, $c_\sigma^{+1}(e) = c_e(n_\sigma(e) + 1)$.

We consider two special cases of network games. A network game where all players have the same sink node is called a *multicast game*. We refer to this common sink by the *root* of the game, and we say player p *is in node* v if the source of p is v. A *broadcast game* is a special case of a multicast game with $|V| - 1$ players and each node except the

root is the source of exactly one player. Hence, a broadcast game (V, E, r) is given by a graph (V, E) and a root node r, and every node wants to connect to the root, while in a multicast game (V, E, r, P) only a subset P of the nodes has to connect to the root. To distinguish these special cases from the multi source-sink case $(V, E, \{(s_p, t_p) : p \in P\})$, we refer to the latter as *general game*.

We give a small example to illustrate our notation.

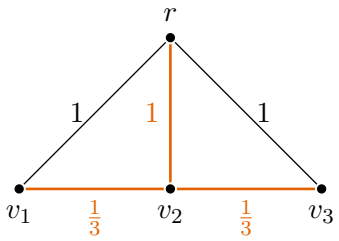


Figure 4.3: A uniform broadcast game with root r and three players located in v_1, v_2, and v_3. The underlying function is $f_{0.5} \in \mathcal{F}_{\text{lin}}$, that is, $f_{0.5}(k) = 0.5 + \frac{0.5}{k}$. The labels on the edges are the scaling factors γ_e. The orange edges represent a strategy profile σ which is an equilibrium.

Example 14. Consider the graph shown in Figure 4.3. This is an instance of a uniform broadcast game. The root is node r and there are three players located in nodes v_1, v_2 and v_3. As underlying function for the costs on the edges we take $f_{0.5} \in \mathcal{F}_{\text{lin}}$, that is, $f_{0.5}(k) = 0.5 + \frac{0.5}{k}$. The scaling factors γ are given by the labels on the edges. For example, we have $\gamma_{\{v_1,r\}} = 1$. The orange edges represent a strategy profile σ, where $\sigma_{v_1} = \{\{v_1, v_2\}, \{v_2, r\}\}, \sigma_{v_2} = \{\{v_2, r\}\}$, and $\sigma_{v_3} = \{\{v_3, v_2\}, \{v_2, r\}\}$. In this profile the congestion on $\{v_2, r\}$ is 3. We observe that σ is an equilibrium, since for player v_1 the current cost is $C_{v_1}(\sigma) = \frac{1}{3} \cdot f(1) + 1 \cdot f(3) = \frac{1}{3} + \frac{2}{3} = 1$, while deviating to $s' = \{\{v_1, r\}\}$ would incur a cost of $C_{v_1}(s', \sigma_{-v_1}) = 1 \cdot f(1) = 1$. Hence deviating is not an improving move for v_1. Checking the remaining strategy for v_1 and applying symmetry shows that no player wants to deviate and hence σ is an equilibrium. The social cost of σ is given by $C(\sigma) = \frac{1}{3} \cdot f(1) + 3 \cdot f(3) + \frac{1}{3} \cdot f(1) = \frac{8}{3}$. Note that σ is no longer an equilibrium for $f_{0.9} \in \mathcal{F}_{\text{lin}}$ as underlying cost function. ⌟

Quality of Equilibria To recall the definitions of PoA and PoS from Section 2.2, we give another example.

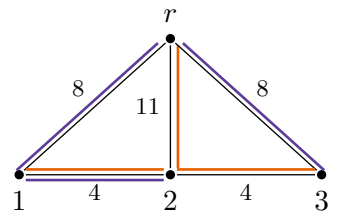


Figure 4.4: Instance of a broadcast game with fair cost allocation. The best equilibrium is shown in orange and one of the worst equilibria is shown in purple.

Example 15. Consider the broadcast game with fair cost allocation shown in Figure 4.4. The path $r - 1 - 2 - 3$ is a social optimum with social cost $8 + 4 + 4 = 16$. There are three equilibria in the game. The orange tree with social cost 19, the purple tree and the symmetric version of the purple tree both with social cost 20.

We observe that the Price of Anarchy of this instance is $\frac{20}{16}$ and the Price of Stability is $\frac{19}{16}$. Thus $\text{PoA}(3) \geq \frac{20}{16}$ and $\text{PoS}(3) \geq \frac{19}{16}$. ⌟

Trees Finally, let us introduce notation related to trees. In a spanning tree T of a graph (V, E) there is exactly one path for every pair of nodes. We denote by $T[v, w]$ the unique $v - w$ path in T. We are mostly dealing with rooted trees which have a distinguished root node. In a rooted tree (T, r) we will think of the edges being directed towards the root. That is the path $T[v, r]$ starts at v and goes to r. The first edge on the path $T[v, r]$ (the one incident to v) is called the *parent edge* of v in T and denoted by $e_T(v)$. The other node of this edge is called the *parent* of v in T denoted by $p_T(v)$, i.e., if $e_T(v) = \{u, v\}$ then $u = p_T(v)$.

Looking from v at the other direction we have the *descendants* $D_T(v)$ of v, which are all nodes that have v on their path: $D_T(v) = \{u \in V : v \in T[u, r]\}$. Note that this also includes v itself. For two nodes u and v we denote by $\text{lca}_T(u, v)$ the *lowest common ancestor* which is the first node where the paths $T[u, r]$ and $T[v, r]$ meet. We refer to the part of the path $T[u, r]$ up to $\text{lca}_T(u, v)$ by $T[u \nearrow v] = T[u, \text{lca}_T(u, v)]$ and analogously for the part of $T[v, r]$ by $T[v \nearrow u] = T[v, \text{lca}_T(u, v)]$. Note that $T[u \nearrow v]$ can be empty, which is the case if $v \in D_T(u)$. From the definition of $\text{lca}_T(u, v)$ we have the following equalities for the edge sets of the paths $T[u, r] \setminus T[u \nearrow v] = T[\text{lca}_T(u, v), r] = T[v, r] \setminus T[v \nearrow u]$. See Figure 4.5 for an illustration of the notation. In figures we draw single edges as straight lines and paths containing more edges as curves.

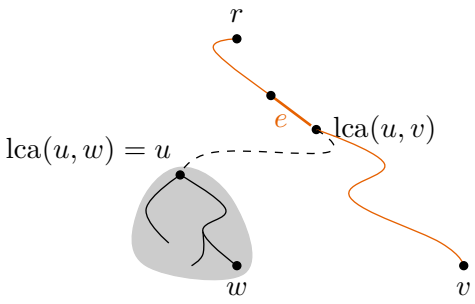


Figure 4.5: Schematic drawing of parts of a rooted tree (T, r). The path $T[v, r]$ is shown in orange. The thick edge labeled e is $e_T(\text{lca}(u, v))$. It is the first edge which v and u use together. The dashed path shows $T[u \nearrow v]$. The nodes in the gray area are the descendants $D_T(u)$ of u (including u). The path $T[u \nearrow w]$ is empty as w is a descendant of u.

4.3 Preliminaries

We begin with some structural properties of equilibria in network games, which will be used throughout this chapter. We consider equilibria in multicast games and show that we can focus on trees spanning all sinks of the players. The first lemma deals with the case where all edge costs are strictly positive.

Lemma 9. *Let (V, E, r, P, c) be a (non-uniform) multicast game, where every edge cost is strictly positive. For any equilibrium σ the support* $\mathrm{supp}(\sigma)$ *is a tree.*

Proof. Let σ be an equilibrium in (V, E, r, P, c). Consider the shortest paths tree T rooted at r w.r.t. the following weights. For an edge e which is used in σ the weight is set to $c_\sigma(e)$ and for $e \notin \mathrm{supp}(\sigma)$ to $+\infty$. We show that for every player p the current strategy σ_p is exactly the path from s_p to r in T. Assume for a contradiction that there is a player p with $\sigma_p \not\subseteq T$. Define $s' = T[s_p, r]$ and consider the new profile $\sigma' = (s', \sigma_{-p})$ where p switched to the path in the shortest paths tree. Now either $\sigma'_p \subset \sigma_p$ and since all edge costs are positive we have the contradiction $C_p(\sigma') < C_p(\sigma)$ to σ being an equilibrium. In the other case we have $\sigma'_p \setminus \sigma_p \neq \emptyset$. From the definitions of σ' and s' we have $C_p(\sigma') = C_{\sigma'}(s')$. We write

$$C_{\sigma'}(s') = \sum_{e \in s' \setminus \sigma_p} c_{\sigma'}(e) + \sum_{e \in s' \cap \sigma_p} c_{\sigma'}(e).$$

For every edge $e \in s' \setminus \sigma_p$ we have $c_{\sigma'}(e) < c_\sigma(e)$ and for all edges $e' \in s' \cap \sigma_p$ it holds $c_{\sigma'}(e') = c_\sigma(e')$. Since $s' \setminus \sigma_p \neq \emptyset$, we obtain $C_{\sigma'}(s') < C_\sigma(s')$. Now, since s' is an $s_p - r$ path in $T \subseteq \mathrm{supp}(\sigma)$ of minimal length w.r.t. c_σ, and σ_p is another $s_p - r$ path in $\mathrm{supp}(\sigma)$, we get $C_\sigma(s') \leq C_\sigma(\sigma_p) = C_p(\sigma)$. In total, we thus have $C_p(\sigma') < C_p(\sigma)$. Again, this is a contradiction to σ being an equilibrium, which concludes the proof. □

If there are edges with cost zero in the graph, an equilibrium need not necessarily be a tree. However, we show that one can always choose a tree which is an equilibrium and differs only on edges of cost zero.

Lemma 10. *Let σ be an equilibrium in the multicast game (V, E, r, P, c). Then there exists a profile σ' such that*

- $\mathrm{supp}(\sigma')$ *is a tree and*
- *for every player p and any edge $e \in \sigma_p \triangle \sigma'_p$ holds $c_e \equiv 0$.*

Proof. If $\mathrm{supp}(\sigma)$ is already a tree, choose $\sigma' = \sigma$. Otherwise, consider the graph $\bar{G}$ constructed from $G = (V, E)$ by contracting all edges where $c_e \equiv 0$, and the profile $\bar{\sigma}$ build from σ in the same way by contracting edges of cost zero in every player path. Note that there will not appear any loops in $\bar{\sigma}$ since this would correspond to paths leaving and reentering a component of zero cost edges. Since σ is an equilibrium and connecting both points within the component is a feasible deviation of less cost (cost zero) this can not happen. Since every path in $\bar{G}$ can be transformed into a path in the original graph G by adding some edges of cost zero, $\bar{\sigma}$ is an equilibrium in $\bar{G}$. Since all

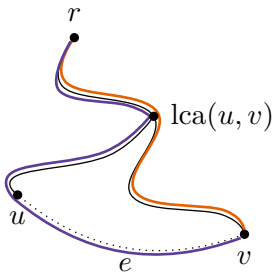
edge costs in $\bar{G}$ are positive we know from Lemma 9 that $\text{supp}(\bar{\sigma})$ is a tree. Hence the only cycles in $\text{supp}(\sigma)$ are within components of zero cost edges. We will now add some of those edges to $\bar{\sigma}$ to obtain our profile σ' in the original instance. By this construction we immediately have the second property that the player paths in σ and σ' differ only on zero cost edges. Take a component Z of zero cost edges containing a cycle of $\text{supp}(\sigma)$, and let $P' \subseteq P$ be the players whose paths form this cycle. From the discussion that there are no loops in $\bar{\sigma}$ we get that every player enters and leaves Z exactly once on their way to the root. Denote by v_1^p and v_2^p the first and last node in Z on the path σ_p starting at s_p for player $p \in P'$. Since $\bar{\sigma}$ is a tree, the nodes where the players leave Z have to be the same, i.e., $v_2^p = v$ for all $p \in P'$. Now we choose any tree T_Z in Z connecting every v_1^p to v. We build σ' by replacing the part of the path σ_p in Z by $T_Z[v_1^p, v]$ for every $p \in P'$. Applying this construction in every component of zero cost edges where $\text{supp}(\sigma)$ contains a cycle, yields a feasible profile σ' whose support is a tree. □

This will help us for example when looking at the PoS of an instance, since we only have to consider trees for equilibria. The lemma shows that for every non-tree equilibrium in a multicast game, there is another equilibrium which is a tree and has the same social cost. The result that an equilibrium in a broadcast game is a tree if all edge costs are positive is already implicitly shown in [Fia+06].

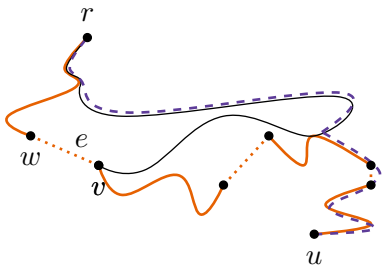
For broadcast games we give a characterization of when a spanning tree is an equilibrium.

Lemma 11. *Let (V, E, r, c) be a broadcast game, T a spanning tree rooted at r and τ the corresponding strategy profile. τ is an equilibrium, iff for every edge $e = \{u, v\}$ not in T the following holds*

$$c_\tau(T[v \nearrow u]) \leq c_e(1) + c_\tau^{+1}(T[u \nearrow v]) \quad \textit{and} \quad c_\tau(T[u \nearrow v]) \leq c_e(1) + c_\tau^{+1}(T[v \nearrow u]). \tag{NECond}$$



(a) The current strategy of v in T is shown in orange. A different strategy t' for v using e and then following u is shown in purple. If $c_\tau(T[v \nearrow u]) > c_e(1) + c_\tau^{+1}(T[u \nearrow v])$, then deviating to purple is an improving move for v.



(b) An improving move t' for v is shown in orange. It uses three edges outside of T. The last edge outside of T is $e = (v, w)$. The other strategy of interest t'' is shown in dashed purple. It is built by using the orange path from u to v and then following $T[v, r]$ (shown in black).

Figure 4.6: Illustrations for Lemma 11. Parts of tree T are shown together with some strategies. Edges in T are drawn solid, while edges outside of T are dotted.

Proof. Let $e = \{u, v\}$ be an edge outside of T and consider player v who is currently connecting to the root r via T, that is, $\tau_v = T[v, r]$. Another feasible strategy for v is to use e and then follow the current strategy of u. Set $t' = \{e\} \cup T[u, r]$ (see Figure 4.6a). Switching to t' is an improving move for v, if and only if

$$\begin{aligned} 0 < C_v(\tau) - C_v(t', \tau_{-v}) &= \sum_{e \in \tau_v \setminus t'} c_\tau(e) - \sum_{e \in t' \setminus \tau_v} c_{(t', \tau_{-v})}(e) \\ &= c_\tau(T[v \nearrow u]) - c_{(t', \tau_{-v})}(e) - c_{(t', \tau_{-v})}(T[u \nearrow v]) \\ &= c_\tau(T[v \nearrow u]) - c(e)(1) - c_\tau^{+1}(T[u \nearrow v]) \end{aligned}$$

Swapping u and v everywhere in the above argument yields that u has an improving move, iff

$$0 < c_\tau(T[u \nearrow v]) - c_e(1) - c_\tau^{+1}(T[v \nearrow u]).$$

Thus, if τ is an equilibrium, no player can have an improving move. In particularly, the players incident to edges outside T. Thus (NECond) has to hold for any $e \in E \setminus T$.

For the other direction, assume that τ is not an equilibrium and let u be a player who has an improving move. Let t' be one of the improving moves for u which minimizes $|t' \setminus T|$, that is, every other improving move for u has at least as many edges outside of T as t'. Further define $e = (v, w)$ to be the last edge outside of T in t', that is, the path t' can be partitioned into three parts: a first part $t'_{u \to v}$ taking u to v, the edge e, and the part from w to r which is $T[w, r]$. Consider an alternative strategy t'' for u, where u follows v instead of taking edge e, i.e., $t'' = t'_{u \to v} \cup T[v, r]$. Note that this is not necessarily a path. If not, we shortcut cycles to obtain a path (see Figure 4.6b). If $C_u(t'', \tau_{-u}) \leq C_u(t', \tau_{-u})$, then t'' would be an improving move for u with fewer edges outside of T (namely e) which contradicts our choice of t'. Thus, we have

$$0 > C_u(t', \tau_{-u}) - C_u(t'', \tau_{-u}) = c_e(1) + c_{(t', \tau_{-u})}(T[w \nearrow v]) - c_{(t'', \tau_{-u})}(T[v \nearrow w])$$

as the first parts and the parts from $\mathrm{lca}_T(v, w)$ are the same in t' and t''. Since player u may not have used any edge of $T[w \nearrow v]$ in τ, we get the lower bound $c_\tau^{+1}(T[w \nearrow v]) \leq c_{(t', \tau_{-u})}(T[w \nearrow v])$. For $T[v \nearrow w]$ it could have been the case that u already uses every edge in τ, thus we get the lower bound $-c_\tau(T[v \nearrow w]) \leq -c_{(t'', \tau_{-u})}(T[v \nearrow w])$. In total we obtain

$$\begin{aligned} 0 &> c_e(1) + c_{(t', \tau_{-u})}(T[w \nearrow v]) - c_{(t'', \tau_{-u})}(T[v \nearrow w]) \\ &\geq c_e(1) + c_\tau^{+1}(T[w \nearrow v]) - c_\tau(T[v \nearrow w]) \end{aligned}$$

which shows that $e = \{v, w\}$ violates (NECond). □

We use this characterization to introduce a special type of improving-dynamics.

Lemma 12. *Consider a spanning tree T whose corresponding profile τ is not an equilibrium in a broadcast game (V, E, r, c). Let $e = \{u, v\} \in E \setminus T$ be an edge violating (NECond), in particular $c_\tau(T[v \nearrow u]) > c_e(1) + c_\tau^{+1}(T[u \nearrow v])$. Define a new spanning tree $T' = T \setminus \{e_T(v)\} \cup \{e\}$ from T by replacing the current parent edge of v be e. Then the profile τ' corresponding to T' can be reached from τ by a sequence of improving moves and for every $w \in D_T(v)$ we have $C_w(\tau') < C_w(\tau)$.*

Proof. First observe that T' is indeed a spanning tree as $u \notin D_T(v)$. If u is a descendant of v then $c_\tau(T[v \nearrow u]) = c_\tau(T[v,v]) = 0$ and since both terms on the right hand side of the condition on edge e are non-negative, this is not possible. From $c_\tau(T[v \nearrow u]) > c_e(1) + c_\tau^{+1}(T[u \nearrow v])$ we immediately see that v has an improving move taking edge e and then following u. Note that the same holds for every descendant of v when replacing the path from v by e followed by the path from u. These paths are exactly the strategies of the descendants in τ'. We show that switching to τ'_w stays an improving move for w even if some of the other descendants already switched to their strategy in τ'. For $W \subseteq D_T(v)$ define τ'_W to be the profile where every player $w \in W$ switched from τ to τ'_w. For example, we have $\tau'_\emptyset = \tau$, $\tau'_{\{w\}} = (\tau'_w, \tau_{-w})$, and $\tau'_{D_T(v)} = \tau'$. Now consider a descendant x of v who did not yet change her path. For the costs of τ_x and τ'_x in τ'_W we have $C_x(\tau'_x, \tau'_W) - C_x(\tau_x, \tau'_W) = c_{(\tau'_x, \tau'_W)}(e) + c_{(\tau'_x, \tau'_W)}(T[u \nearrow v]) - c_{(\tau_x, \tau'_W)}(T[u \nearrow v])$. Since the edge costs are decreasing and x uses e as well as $T[u \nearrow v]$ in (τ'_x, τ'_W) but not in (τ_x, τ'_W), we have $c_{(\tau'_x, \tau'_W)}(e) \leq c_e(1)$ and $c_{(\tau'_x, \tau'_W)}(T[u \nearrow v]) \leq c_\tau^{+1}(T[u \nearrow v])$. On the other hand there are fewer players on $T[v \nearrow u]$ in (τ'_x, τ'_W) than in τ since all players in W left. Hence, we have $c_{(\tau_x, \tau'_W)}(T[u \nearrow v]) \geq c_\tau(T[u \nearrow v])$. In total we obtain

$$C_x(\tau'_x, \tau'_W) - C_x(\tau_x, \tau'_W) \leq c_e(1) + c_\tau^{+1}(T[u \nearrow v]) - c_\tau(T[u \nearrow v]) < 0$$

from the condition on e. Hence τ' can be reached by a sequence of improving moves from τ, where the descendants of v switch subsequently.

Using the same arguments as before we get for the player costs

$$\begin{aligned} C_w(\tau') - C_w(\tau) &= c_{\tau'}(e) + c_{\tau'}(T[u \nearrow v]) - c_\tau(T[v \nearrow u]) \\ &\leq c_e(1) + c_\tau^{+1}(T[u \nearrow v]) - c_\tau(T[v \nearrow u]) < 0. \end{aligned}$$

□

We call the step going from T to T' as in Lemma 12 an (improving) *tree-move*. The process of making several tree-moves is called *tree-dynamics*. The previous lemma shows that tree-dynamics is a refinement of the standard improving-dynamics. Together with Lemma 11 we see that tree-dynamics terminate in equilibria. The advantage of this new dynamics is that we always have a tree after every step, compared to the standard dynamics where cycles can appear in the support.

The above characterization is for equilibria in broadcast games. Recall from Section 2.2, the characterization of Rosenthal [Ros73] for equilibria in general games as the local minimizers of the potential function

$$\Phi(\sigma) = \sum_{r \in R} \sum_{i=1}^{n_\sigma(r)} c_r(i).$$

We introduce the short notation $F_e(k)$ for $\sum_{i=1}^{k} c_e(i)$ and for uniform games we define $F(k) = \sum_{i=1}^{k} f(i)$. Hence the potential can be written as $\Phi(\sigma) = \sum_{e \in \mathrm{supp}(\sigma)} F_e(n_\sigma(e))$

and for uniform games as $\Phi(\sigma) = \sum_{e \in \text{supp}(\sigma)} \gamma_e F(n_\sigma(e))$. For uniform fair cost allocation we obtain $\Phi(\sigma) = \sum_{e \in \text{supp}(\sigma)} H(n_\sigma(e))$ where $H(n)$ is the n-th harmonic number $\sum_{i=1}^{n} \frac{1}{i}$.

Note that since the per-player costs are decreasing, F_e and F are strictly concave. We further denote the average of the first values of a function f by $\bar{f}(k) = \frac{F(k)}{k}$. Observe that for a sharing function f with economies of scale also $\bar{f}$ is a sharing function with economies of scale. As f is decreasing, so is the average of the first values. The total cost w.r.t. $\bar{f}$ is F and hence non-decreasing and concave. There is a relation between socially optimal profiles and equilibria on the same graph but with different underlying cost functions.

Observation 4. A social optimum in a uniform game $(N, \bar{f})$ is an equilibrium in the game (N, f).

In the next sections we look at the quality of equilibria in uniform network games. First, we give the Price of Anarchy for broadcast games. Then we look at the Price of Stability and give upper bounds for general games and broadcast games. For broadcast games we also give lower bounds on the PoS for special classes of cost functions.

4.4 Price of Anarchy

In this section we determine the Price of Anarchy (PoA) of uniform broadcast games. Recall that the PoA is the worst ratio of the social costs of an equilibrium and a social optimum:

$$\text{PoA}(n) = \text{PoA}(\mathcal{N}_n) = \max_{\substack{\text{broadcast game } \mathcal{G} \\ \text{with at most } n \text{ players}}} \max_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\text{OPT}(\mathcal{G}))}$$

Lower bound First we give a lower bound example which is a generalization of the example for fair cost allocation given in [Ans+08a].

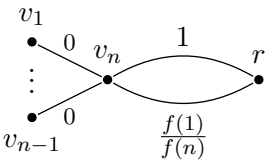


Figure 4.7: A uniform broadcast game with n players at $v_1, \ldots, v_n$ and underlying function f with PoA of $\frac{f(1)}{f(n)}$

Consider the instance of a uniform broadcast game depicted in Figure 4.7. There are n players located in node v_n and we use the standard construction to transform an instance with more than one player per node to a proper broadcast instance with exactly one player at every node. Essentially there are two edges connecting v_n to the root r. We call these the top edge e_{top} and the bottom edge e_{bot}. The underlying function to this uniform game is f and the scaling factors are $\gamma_{e_{\text{top}}} = 1$ and $\gamma_{e_{\text{bot}}} = \frac{f(1)}{f(n)}$.

First we show that the social optimum is given by the profile σ_{OPT} where all players use the top edge. The corresponding social cost is $nf(n)$. For any strategy profile σ we denote by n_{bot} the number of players using the bottom edge. We can then bound the social cost of σ as:

$$\begin{aligned} C(\sigma) &= (n - n_{\mathrm{bot}})f(n - n_{\mathrm{bot}}) + \frac{f(1)}{f(n)} n_{\mathrm{bot}} f(n_{\mathrm{bot}}) \\ &\geq (n - n_{\mathrm{bot}})f(n) + \frac{f(1)}{f(n)} n_{\mathrm{bot}} f(n) \\ &= \left(n + \left(\frac{f(1)}{f(n)} - 1\right) n_{\mathrm{bot}}\right) f(n) \end{aligned}$$

where the inequality holds as f is decreasing and $n - n_{\mathrm{bot}}, n_{\mathrm{bot}} \leq n$. From $\frac{f(1)}{f(n)} - 1 \geq 0$ we see that the social cost is minimal if $n_{\mathrm{bot}} = 0$, that is, if all players use the top edge.

Now consider the profile σ^* where all players play the bottom edge. This is an equilibrium since the cost of each player is $\frac{f(1)}{f(n)} f(n) = f(1)$ and switching to the top edge is not an improving move as the incurred cost would be $f(1)$.

In total we get a lower bound on the PoA of:

$$\frac{C(\sigma^*)}{C(\sigma_{\mathrm{OPT}})} = \frac{\frac{f(1)}{f(n)} nf(n)}{nf(n)} = \frac{f(1)}{f(n)}.$$

Upper bound For the upper bound on the PoA we bound the cost for each player in an equilibrium by her cost in a social optimum. For an instance (V, E, r, f, γ) of a uniform broadcast game let σ_{OPT} be a profile corresponding to a social optimum and let σ^* be an equilibrium. Consider the profile σ' constructed from σ^* by letting one player, say v, switch to her strategy in σ_{OPT}, i.e., $\sigma' = (\sigma_{\mathrm{OPT},v}, \sigma^*_{-v})$. For any edge $e \in \sigma'_v = \sigma_{\mathrm{OPT},v}$ we have $n_{\sigma'}(e) \geq 1$ since at least v uses e, and $n_{\sigma_{\mathrm{OPT}}}(e) \leq n$. As f is decreasing we get the following bound on the cost of v

$$\begin{aligned} C_v(\sigma') = \sum_{e \in \sigma'_v} \gamma_e f(n_{\sigma'}(e)) &= \sum_{e \in \sigma'_v} \gamma_e \frac{f(n_{\sigma'}(e))}{f(n_{\sigma_{\mathrm{OPT}}}(e))} f(n_{\sigma_{\mathrm{OPT}}}(e)) \\ &\leq \sum_{e \in \sigma'_v} \gamma_e \frac{f(1)}{f(n)} f(n_{\sigma_{\mathrm{OPT}}}(e)) = \frac{f(1)}{f(n)} C_v(\sigma_{\mathrm{OPT}}) \end{aligned}$$

As σ^* is an equilibrium we know that the cost of player v can not decrease by deviating to her strategy in σ_{OPT}. Thus we get $C_v(\sigma^*) \leq C_v(\sigma') \leq \frac{f(1)}{f(n)} C_v(\sigma_{\mathrm{OPT}})$.

This bound holds for every player so that in total the social cost of σ^* is bounded by

$$C(\sigma^*) \leq \frac{f(1)}{f(n)} C(\sigma_{\mathrm{OPT}}).$$

With this matching upper and lower bound we have shown the following theorem.

Theorem 7. *The PoA for uniform broadcast games with n players and underlying function f is*

$$\frac{f(1)}{f(n)}.$$

Observe that for fair cost allocation, where $f(k) = \frac{1}{k}$, we obtain the known bound $\frac{f(1)}{f(n)} = n$ from [Ans+08a].

4.5 Price of Stability – Upper Bounds

After looking at the worst equilibria in the previous section, we will now look at the best equilibria. In this section, we give upper bounds on the Price of Stability (PoS) for uniform network games. Recall the definition of the PoS as the best ratio of the social costs of an equilibrium and a social optimum:

$$\mathrm{PoS}(n) = \mathrm{PoS}(\mathcal{N}_n) = \max_{\substack{\text{network game } \mathcal{G} \\ \text{with at most } n \text{ players}}} \min_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\mathrm{OPT}(\mathcal{G}))}.$$

To get upper bounds on the PoS, we need to show that in every instance, there is an equilibrium with social cost close to a social optimum. For general games, we relate the social cost of an equilibrium and a social optimum by the potential as done in [Ans+08a] for fair cost allocation. With this method we give specific upper bounds for functions in $\mathcal{F}_{\mathrm{lin}}$ and $\mathcal{F}_{\mathrm{poly}}$. For broadcast games, we simplify the homogenization-absorption framework of Bilò, Flammini, and Moscardelli [BFM20], allowing us to generalize it to other cost functions than fair cost allocation. Further, we explicitly compute a constant upper bound on the PoS for fair cost allocation.

4.5.1 Potential Method for General Games

Anshelevich et al. observe in [Ans+08a] that the potential $\Phi(\sigma) = \sum_{e \in \mathrm{supp}(\sigma)} \sum_{i=1}^{n_\sigma(e)} c_e(i)$ not only guarantees the existence of equilibria (as noted by [Ros73]) but also gives a bound on the social cost of a particular equilibrium. We show how this method can be used for sharing functions with economies of scale. For any strategy profile σ we give an upper and a lower bound on the social cost in terms of the potential. First, observe that since the edge cost functions are decreasing we have for any $i \in \{1, \dots, n_\sigma(e)\}$ the bound $c_e(n_\sigma(e)) \leq c_e(i)$. From this we immediately get

$$C(\sigma) = \sum_{e \in \mathrm{supp}(\sigma)} n_\sigma(e) c_e(n_\sigma(e)) \leq \Phi(\sigma).$$

On the other hand, we have for any edge $e \in \mathrm{supp}(\sigma)$

$$\sum_{i=1}^{n_\sigma(e)} c_e(n_\sigma(e)) = \frac{\sum_{i=1}^{n_\sigma(e)} c_e(n_\sigma(e))}{n_\sigma(e) c_e(n_\sigma(e))} n_\sigma(e) c_e(n_\sigma(e)) \leq \max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} c_e(n)}{n c_e(n)} n_\sigma(e) c_e(n_\sigma(e))$$

and hence

$$\Phi(\sigma) \leq \max_{e \in \text{supp}(\sigma)} \max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} c_e(n)}{n c_e(n)} C(\sigma).$$

For uniform games with underlying function f the factor on the right hand side evaluates to $\max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} f(i)}{nf(n)}$, since for every edge the scaling factor γ_e cancels.

To bound the PoS consider a profile σ_{OPT} corresponding to a social optimum. This need not be an equilibrium, but if we start improving-dynamics from here we will reach an equilibrium σ^* at some point. By this construction we have $\Phi(\sigma^*) \leq \Phi(\sigma_{\text{OPT}})$. Combining all bounds we established so far, we get

$$C(\sigma^*) \leq \Phi(\sigma^*) \leq \Phi(\sigma_{\text{OPT}}) \leq \max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} f(i)}{nf(n)} C(\sigma_{\text{OPT}}).$$

Hence, the value $\max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} f(i)}{nf(n)}$ is an upper bound on the PoS for uniform network games. We define the short hand notation $M(f)$

$$M(f) = \max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} f(i)}{nf(n)}.$$

Using this method, we can show a kind of monotonicity of the PoS. If the slope of a function is point-wise higher than the slope of another function, then the PoS w.r.t. the first function is smaller.

Lemma 13. *For two sharing functions with economies of scale f and g satisfying*

$$\forall k \in \mathbb{N}_{\geq 1} : \frac{f(k+1) - f(k)}{f(1)} \leq \frac{g(k+1) - g(k)}{g(1)}$$

we have $M(g) \leq M(f)$.

Proof. Chaining the inequalities from the assumption of the statement we obtain

$$\begin{aligned} \frac{f(n) - f(k)}{f(1)} &= \frac{f(n) - f(n-1) + f(n-1) - \ldots + f(k+1) - f(k)}{f(1)} \\ &\leq \frac{g(n) - g(n-1) + g(n-1) - \ldots + g(k+1) - g(k)}{g(1)} = \frac{g(n) - g(k)}{g(1)} \end{aligned}$$

for any $k \leq n$. Plugging in $k = 1$, we especially obtain $\frac{f(n)}{f(1)} \leq \frac{g(n)}{g(1)}$. Combining all inequalities we get

$$\begin{aligned} \frac{\sum_{k=1}^{n} g(k)}{ng(n)} &= \frac{ng(n) + \sum_{k=1}^{n} (g(k) - g(n))}{ng(n)} = 1 + \frac{\sum_{k=1}^{n} \frac{g(k) - g(n)}{g(1)}}{n \frac{g(n)}{g(1)}} \\ &\leq 1 + \frac{\sum_{k=1}^{n} \frac{f(k) - f(n)}{f(1)}}{n \frac{f(n)}{f(1)}} = \frac{\sum_{k=1}^{n} f(k)}{nf(n)} \end{aligned}$$

and thus $M(g) \leq M(f)$. □

In the remainder of this section we give bounds on $M(f)$ for some functions f to obtain specific bounds on the PoS.

We begin with a first bound for any sharing function f with economies of scale which is a generalization of the $H(n)$ bound of [Ans+08a] for fair cost allocation.

Lemma 14. *For any sharing function f with economies of scale we have*

$$M(f) \leq 1 + \ln\left(\frac{f(1)}{f(n)}\right).$$

Proof. Since f is decreasing we have $f(k) \leq f(1)$ for any $k \in \mathbb{N}$. From the non-decreasing total cost $kf(k)$ we get another upper bound $f(k) \leq \frac{n}{k} f(n)$ for all $1 \leq k \leq n$. Combining both bounds yields

$$M(f) \leq \frac{\sum_{k=1}^{n} \min\{f(1), \frac{n}{k} f(n)\}}{nf(n)} \leq \frac{\int_{k=0}^{n} \min\{f(1), \frac{n}{k} f(n)\} \mathrm{d}k}{nf(n)}$$

since $\min\{f(1), \frac{n}{k} f(n)\}$ is non-increasing in k. The point where the minimum switches values is at $k^* = n\frac{f(n)}{f(1)}$, where for smaller k the minimum is $f(1)$. We can thus compute the integral and obtain the desired bound

$$\begin{aligned}
\frac{\int_{k=0}^{n} \min\{f(1), \frac{n}{k} f(n)\}}{nf(n)} &= \frac{\int_{k=0}^{n\frac{f(n)}{f(1)}} f(1)\mathrm{d}k + \int_{k=n\frac{f(n)}{f(1)}}^{n} \frac{n}{k} f(n)\mathrm{d}k}{nf(n)} \\
&= \frac{f(1)n\frac{f(n)}{f(1)} + nf(n)\left(\ln(n) - \ln\left(n\frac{f(n)}{f(1)}\right)\right)}{nf(n)} \\
&= 1 + \ln(n) - \ln\left(n\frac{f(n)}{f(1)}\right) \\
&= 1 + \ln\left(\frac{f(1)}{f(n)}\right).
\end{aligned}$$

□

For fair cost allocation, where $f(k) = \frac{1}{k}$ this yields a bound of $1 + \ln(n)$ which is an upper bound on the bound $H(n)$ given in [Ans+08a].

We will now consider functions in $\mathcal{F}_{\text{lin}}$. Recall that for $f_s \in \mathcal{F}_{\text{lin}}$ we have $f_s(k) = s + \frac{(1-s)}{k}$.

Lemma 15. *For $f_s \in \mathcal{F}_{lin}$ with $s > 0$ we have*

$$M(f_s) \leq 1 + W\left(\frac{1-s}{se}\right).$$

Where W denotes the principal branch of the Lambert W function with the property $W(x)\mathrm{e}^{W(x)} = x$ for $x > 0$.

Proof. We compute

$$M(f_s) = \max_{n\in\mathbb{N}_{>0}} \frac{sn + (1-s)H(n)}{sn+1-s} \leq \max_{n\in\mathbb{N}_{>0}} \frac{sn + (1-s)(\ln(n)+1)}{sn+1-s} = 1 + \max_{n\in\mathbb{N}_{>0}} \frac{(1-s)\ln(n)}{sn+1-s}$$

where we used the bound $H(n) \leq \ln(n) + 1$. To find the maximum on the right hand side we set the derivative w.r.t. n to 0:

$$\frac{\mathrm{d}}{\mathrm{d}n}\frac{(1-s)\ln(n)}{sn+1-s} = 0 \iff \frac{1-s}{n}(sn+1-s) - s(1-s)\ln(n) = 0 \iff \ln(n) = 1 + \frac{1-s}{sn}$$

This can be equivalently written as $x = y\mathrm{e}^y$ where $x = \frac{1-s}{s\mathrm{e}}$ and $y = \frac{1-s}{sn}$. From the definition of the Lambert W function we know that the unique solution to this equation is $y = W(x)$. Solving this for n yields $n = \frac{1-s}{sW(x)}$. Rearranging the equality to $\frac{x}{W(x)} = \mathrm{e}^{W(x)}$ we can write $n = \mathrm{e}^{W(x)+1}$. We check that this point actually gives a maximum. We have $\frac{(1-s)\ln(n)}{sn+1-s} \geq 0$ for all $n \in \mathbb{N}_{>0}$. Further, since for $n = 1$ the value is 0 and there is a unique extreme point it has to be a maximum. Evaluating $\frac{(1-s)\ln(n)}{sn+1-s}$ at this value for n we obtain for the maximum:

$$\frac{(1-s)\ln(n)}{sn+1-s} = \frac{(1-s)\left(W\left(\frac{1-s}{s\mathrm{e}}\right)+1\right)}{s\frac{(1-s)}{sW\left(\frac{1-s}{s\mathrm{e}}\right)}+1-s} = W\left(\frac{1-s}{s\mathrm{e}}\right).$$

□

This shows that for any function in $\mathcal{F}_{\text{lin}}$ other than fair cost allocation (where $s = 0$), the PoS in general games is constant. However, for s getting smaller this constant is growing big. For $s \leq \frac{1}{1+e^2}$ we have the bound $1 + W\left(\frac{1-s}{s\mathrm{e}}\right) \leq \ln\left(\frac{1}{s}\right)$ (using the inequality $W(x) \leq \ln(x)$ for $x \geq \mathrm{e}$ from [Has05]).

For functions $f_\alpha \in \mathcal{F}_{\text{poly}}$ we get a similar result saying that the PoS is constant for any function except fair cost allocation (where $\alpha = 0$). Recall that for $f_\alpha \in \mathcal{F}_{\text{poly}}$ we have $f_\alpha(k) = k^{\alpha-1}$.

Lemma 16. *For $f_\alpha \in \mathcal{F}_{poly}$ with $\alpha > 0$ we have*

$$M(f_\alpha) \leq \frac{1}{\alpha}.$$

Proof. We compute

$$M(f_\alpha) = \max_{n\in\mathbb{N}_{>0}} \frac{\sum_{k=1}^n k^{\alpha-1}}{nn^{\alpha-1}} \leq \max_{n\in\mathbb{N}_{>0}} \frac{1 + \int_{k=1}^n k^{\alpha-1}\mathrm{d}k}{n^\alpha} = \max_{n\in\mathbb{N}_{>0}} \frac{1}{\alpha} - \frac{1-\alpha}{\alpha n^\alpha} \leq \frac{1}{\alpha}$$

□

The above bounds are good for large values of s and α, where the per-player cost decreases slowly.

4.5.2 Homogenization-Absorption Framework for Broadcast Games

For broadcast games we show a constant upper bound for functions decreasing quickly. Upper bounds on the PoS in broadcast games have been studied for fair cost allocation. Fiat et al. [Fia+06] give a first bound of $\mathcal{O}(\log \log n)$ for this class, which was later improved to $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$ by Li [Li09] and $\mathcal{O}(\log \log \log n)$ by Lee and Ligett [LL13], and finally Bilò, Flammini, and Moscardelli [BFM20] show that the PoS is bounded by a constant independent on the number of players. The results by [Fia+06], [LL13] and [BFM20] have been achieved by a *homogenization-absorption framework*.

Recall that to prove an upper bound on the PoS, we need to show that there is a good equilibrium in every broadcast game, whose social cost is close to the cost of a social optimum. The idea in the homogenization-absorption framework is closely related to the upper bound from the potential method. We start with a profile of a social optimum and show that by improving moves an equilibrium of good social cost can be reached. Since we did not restrict the improving moves in the potential method, we had to look at the worst ratio of the contributions of every edge to the social cost. In the homogenization-absorption framework, we are now considering specifically chosen improving moves which allows us to have a better understanding of the relation of the social cost of the start and end state. The guiding idea is to stay as close as possible to the social optimum in terms of edges being used.

We consider improving moves introducing only one new edge, i.e., tree-moves. This process reaches an equilibrium by Lemma 12. If a player introduces a new edge outside of the social optimum, we *absorb* players who are close by in the social optimum. For every player we consider the path in the social optimum to the deviating player, who introduced the edge. If the length of this path (w.r.t. some notion of distance) is less than the *absorption-radius*, the player switches to the social optimum path and then follows the deviating player. The radius is chosen relative to the cost of the new edge. This process possibly reduces the number of edges in the support outside of the social optimum and decreases the potential. Further, two edges of similar cost in the support of the final equilibrium which are not in the social optimum have to be far apart, as otherwise one of the players absorbs the other.

The idea of using edges of the social optimum is also used in the *homogenization* of states, where the goal is to assimilate player costs. As absorption, also homogenization decreases the number of edges outside of the social optimum and the potential.

The iterative algorithm to find a good equilibrium from a social optimum is as follows. As long as the profile is not an equilibrium, let a player deviate by a tree-move, absorb players via the social optimum, homogenize the resulting profile and repeat. At the beginning of each iteration we have a homogeneous profile corresponding to a tree, and in every iteration the potential decreases. Hence, this algorithm terminates. The bound on the PoS obtained by this method is related to the absorption-radius. If this radius is large, edges of similar cost have to be far apart w.r.t. the social optimum and hence there can not be that many. On the other hand, it could be the case that there are many edges outside of the social optimum with smaller and smaller costs such that absorption would not remove some of them. By carefully charging the cost of the edge to a part of

the absorption-ball given by the *charging-radius*, it is possible to bound the total cost of these edges. On the one hand, we want to make the charging-radius large, that the cost of the edge is covered by the charged parts, on the other hand, we have to make the charging-radius small enough, so that not too many of the charging-balls can overlap.

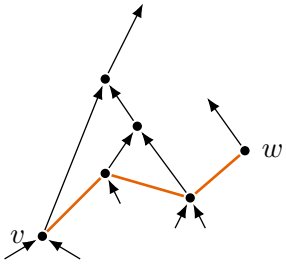
Fiat et al. [Fia+06] are the first to use this homogenization-absorption framework to get an upper bound of $\mathcal{O}(\log \log n)$. While they consider single player improvement moves, Lee and Ligett [LL13] introduce group moves reducing the potential. Further, they reduce the overlap in the charging-balls to obtain the improved bound of $\mathcal{O}(\log \log \log n)$. Using a better homogenization and charging-scheme, Bilò, Flammini, and Moscardelli [BFM20] show that the PoS is constant. The constant is not explicitly given but estimated to be very large.

In this section, we show how to generalize the techniques of Bilò, Flammini, and Moscardelli [BFM20] to sharing functions with economies of scale. While the techniques are the same, we use a different representation, which we think is more clear.

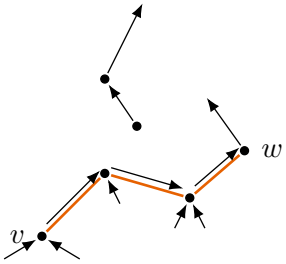
We consider the algorithm as explained above. Starting with a social optimum, let a player deviate using a single new edge, absorb other nearby players and homogenize the resulting profile w.r.t. the social optimum. These steps are repeated until we reach an equilibrium. To show that this algorithm terminates after finitely many iterations, we show that the potential decreases in every iteration. To get an upper bound on the PoS, we design a charging-scheme distributing the contribution of an edge outside of the social optimum to the social cost to edges in the social optimum.

First, we describe the three parts at a high level before going into details.

Homogenization. The goal of homogenization is to assimilate the costs of the players, such that the difference of two player costs can be bounded by the length of the path connecting them in a social optimum. Here the length is w.r.t. some distance introduced later. See Figure 4.8 for a drawing of a homogenization step. In a homogeneous profile,



(a) Parts of the initial profile, where v connects to r not using w.



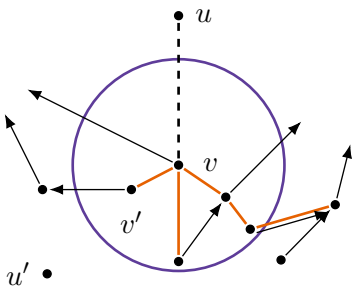
(b) A move of v: going to w via edges in a social optimum. In contrast to only v deviating, we consider also all players met along the path to join.

Figure 4.8: The moves considered for homogenization of profiles. The current support is shown in black, where the directions point to the root. Edges in a social optimum are shown in orange.

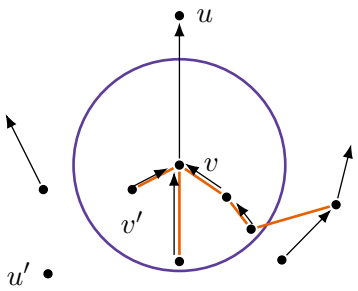
no player has an "improving" move of the form: connecting via a social optimum to some other node and following the strategy of this node. Here, the cost along the path in the social optimum is not the actual cost incurred to the player, but the share of

the potential on these edges. We give the formal definition later. The idea is that in a homogeneous profile, the potential can not be decreased by moves as described above and shown in Figure 4.8.

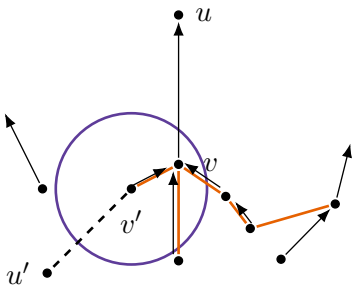
Absorption. After a player deviated and introduced a new edge outside of the social optimum, she absorbs players, who are close in a social optimum. See Figure 4.9 for a schematic drawing of two absorption steps. The idea is that, if it is beneficial for a



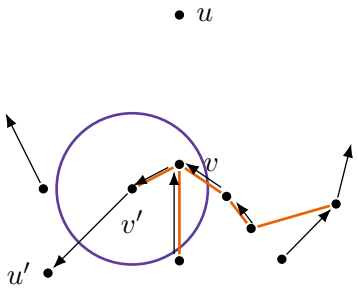
(a) Before the first absorption. v is about to deviate to $\{v, u\}$ (dashed).



(b) After the first absorption. The 4 players in the absorption-ball changed their strategy, removing 4 edges outside of the social optimum.



(c) Before the second absorption. Now v' is about to deviate to $\{v', u'\}$.



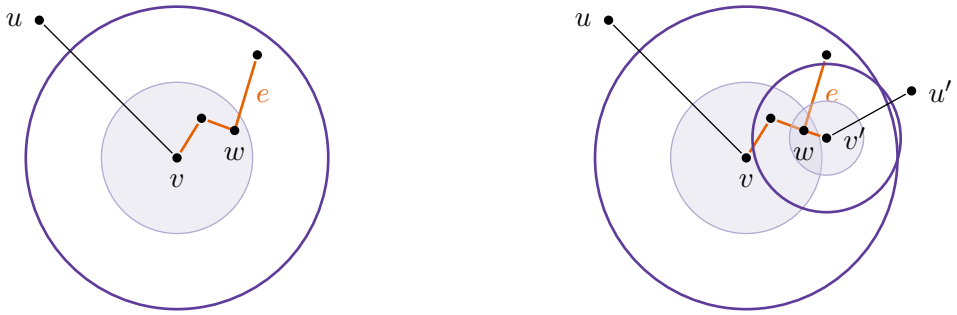
(d) After the second absorption. The edge $\{v, u\}$ is removed.

Figure 4.9: Two absorption steps. Parts of the current support are shown in black. The arrows give the direction to the root in this tree. Parts of a social optimum are shown in orange. The absorption-balls are shown in purple.

player to deviate, then following this player is also good for players who are close in the social optimum. This depends on the cost of the new edge and hence we choose the absorption-radius relative to this cost. The main benefit of absorption is to delete edges outside of the social optimum. Observe that in the second step the previously introduced edge $\{v, u\}$ is deleted again, as v is in the absorption-ball of v'. We show that an absorption step decreases the potential, if the initial profile is homogeneous.

Charging-Scheme. Repeating absorption and homogenization steps terminates in an equilibrium. To bound the social cost of this profile we look at the edges outside of the social optimum and charge them to some edges in the social optimum. In this charging-scheme we have to make sure that the cost of the edge to be distributed is covered while bounding the total charge on social optimum edges. Every edge in the final profile that is not

part of the social optimum has a corresponding absorption-ball. We charge the cost of the edge to edges of the social optimum in a part of this ball. This part is called the charging-ball and its radius is chosen relative to the absorption-radius. See Figure 4.10 for an illustration of the charging-ball relative to the absorption-ball. The cost of $\{u, v\}$



(a) Absorption and charging-ball around v corresponding to edge $\{v, u\}$. The cost on $\{v, u\}$ is charged to the orange edges.

(b) The charging-balls around v and v' corresponding to $\{v, u\}$ and $\{v', u'\}$ overlap in w. Edge e gets charged by both $\{v, u\}$ and $\{v', u'\}$

Figure 4.10: Relation of absorption-balls (purple) and charging-balls (light purple). Edges in a social optimum are shown in orange. (a) shows the two balls for node v. (b) shows two overlapping charging-balls at v and v'.

is charged to the edges of the social optimum inside and the edge leaving this charging-ball, i.e., the orange edges connecting v to w and edge e in Figure 4.10a. The distribution is chosen such that the cost of $\{u, v\}$ is covered. From Figure 4.10b we observe that the radii of overlapping charging-balls have to decrease exponentially. Although an edge of the social optimum can be charged by many balls, their cost has to get small and hence we can bound the total charge on edges of the social optimum.

We remark here that none of the current upper bounds use any property of a social optimum other than the fact that it is a spanning tree. All arguments can be done w.r.t. any spanning tree in the graph.

Research Question 4. Which properties of a social optimum can be used to improve the upper bounds obtained by the homogenization-absorption framework?

We were in particular not able to use the fact that also the social optimum has to pay more compared to fair cost allocation. In fair cost allocation the contribution of an edge to the social cost of a profile does not depend on the number of players using the edge. In contrast to this we can not use edges in the social optimum for free when looking at sharing functions with economies of scale. Hence, we have an additional factor of $\frac{kf(k)}{f(1)}$ in our bound, since it could be the case that in the social optimum only one player is using the edge, but in our construction k players use the edge.

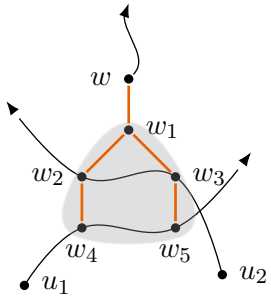
Now we come to the details for the three main parts explained above. We begin with the underlying idea of connecting strategies along the social optimum, which is then used by both homogenization and absorption. Finally we describe the charging-scheme and conclude with the upper bound obtained by this method.

Using the social optimum Both homogenization and absorption are steps where parts of the current support are replaced by parts of the social optimum tree. While in homogenization we use paths as replacements, absorption uses trees inside the absorption-balls. We give a formal definition of this replacement.

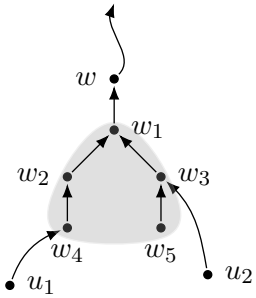
Consider a profile σ with support S which is a tree. Further, let $W \subseteq V$ be a subset of the nodes, let w be any node and let T be a tree spanning $W \cup \{w\}$. We define the quasi-profile $\bar{\sigma}^{T,w}$, where we change the strategies of all players in $D_S(W)$ to connect via T to w and then following w. Formally, we have

$$\bar{\sigma}_u^{T,w} = S[u, w'(u)] \cup T[w'(u), w] \cup S[w, r] \tag{4.1}$$

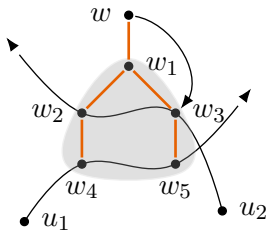
for players $u \in D_S(W) \setminus \{w\}$ and $\bar{\sigma}_v^{T,w} = \sigma_v$ for all other players. Here we denote by $w'(u)$ the first node on the path $S[u, r]$ in W. Consider Figure 4.11 for an example.



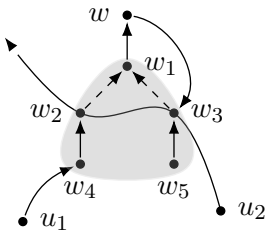
(a) σ_w does not go through W.



(b) The quasi-profile $\bar{\sigma}^{T,w}$ which is a valid profile with tree support.



(c) σ_w goes through $W \setminus \{w\}$.



(d) $\bar{\sigma}^{T,w}$ containing the dashed and solid edges is not a valid strategy profile, as the edge sets for all nodes except w_1 contain cycles. The profile $\sigma^{T,w}$ containing only the solid arcs is valid and has tree support.

Figure 4.11: The two cases of using the orange tree T in $W = \{w_1, \ldots, w_5\}$ and w. Parts of the current profile σ are shown on the left in black. We have $w'(u_1) = w_4$ and $w'(u_2) = w_3$.

Note that this is not necessarily a valid strategy profile, as the given edge set for a player may contain cycles (see Figure 4.11d). However, we have a characterization when this happens and define a very similar profile with tree support.

Lemma 17. *Let σ be a strategy profile with tree support S. Take $W \subseteq V$ and $w \in V$, and let T be a tree spanning $W \cup \{w\}$. Then $\bar{\sigma}^{T,w}$ has tree support, if and only if $\sigma_w \cap (W \setminus \{w\}) = \emptyset$.*

Proof. First assume that $\sigma_w \cap (W \setminus \{w\}) = \emptyset$. Consider the set of edges

$$\bigcup_{u \in D_S(W) \setminus \{w\}} S[u, w'(u)],$$

i.e., the first parts of the new strategies up to W. Since S is a tree this set of edges does not contain a cycle. As $w'(u)$ is the first node in W on $S[u, r]$ and since T is a tree spanning $W \cup \{w\}$, the set of edges formed by the new strategies up to node w, i.e., $\bigcup_{u \in D_S(W) \setminus \{w\}} S[u, w'(u)] \cup T[w'(u), w]$, does not contain a cycle. Since all of the players $u \in D_S(W)$ use the path $S[w, r]$ in $\bar{\sigma}^{T,w}$ and $S[w, r]$ does not intersect W, there can not be a cycle in the union of the new strategies of players in $D_S(W) \cup \{w\}$.

The strategies of the other players $u \notin D_S(W)$ do not change and they do not intersect W. Hence their union does not contain a cycle, since S is a tree. This shows that the support of $\bar{\sigma}^{T,w}$ is a tree.

On the other hand, if $\sigma_w \cap W \setminus \{w\} \neq \emptyset$ there is a $v \in W \setminus \{w\}$ such that $v \in S[w, r]$. Then the edge set $\bar{\sigma}_v^{T,w} = T[v, w] \cup S[w, r]$ and hence the support contains a cycle. □

If σ_w goes through $W \setminus \{w\}$, we do shortcuts as necessary in the strategies $\bar{\sigma}_u^{T,w}$. That is, instead of going to w, players only go to the first node, where they meet $S[w, r]$ in $W \cup \{w\}$. We define $\sigma^{T,w}$ from $\bar{\sigma}^{T,w}$. We interpret $\bar{\sigma}_u^{T,w}$ as walk starting at u and set

$$\sigma_u^{T,w} = \bar{\sigma}_u^{T,w}[u, w''(u)] \cup S[w''(u), r] \tag{4.2}$$

for players $u \in D_S(W) \setminus \{w\}$ and $\sigma_u^{T,w} = \sigma_u$ for all other players. Here $w''(u)$ is the first node on $\bar{\sigma}_u^{T,w}$ in $S[w, r]$. See Figure 4.11d where $w'(u_1) = w_4$ and $w''(u_1) = w_2$, and for w_1 we have $w''(w_1) = w$.

We show that $\sigma^{T,w}$ has tree support, and thus is a valid strategy profile.

Lemma 18. *Let σ be a strategy profile with tree support S. Take $W \subseteq V$ and $w \in V$, and let T be a tree spanning $W \cup \{w\}$. Then, the support of $\sigma^{T,w}$ is a tree.*

Proof. If $\sigma_w \cap (W \setminus \{w\}) = \emptyset$, then the statement follows by Lemma 17 and the observation that $\sigma^{T,w} = \bar{\sigma}^{T,w}$ in this case.

In the case where $\sigma_w \cap (W \setminus \{w\}) \neq \emptyset$ we have as in the proof of Lemma 17 that the edge set

$$\bigcup_{u \in D_S(W) \setminus \{w\}} \bar{\sigma}_u^{T,w}[u, w''(u)] \subseteq \bigcup_{u \in D_S(W) \setminus \{w\}} S[u, w'(u)] \cup T[w'(u), w]$$

does not contain cycles. The remaining edges of the strategies of players in $D_S(W)$ are all part of the path $S[w, r]$. Since $w''(u)$ is the first node on this path, adding this path as strategy of w does not introduce cycles.

As in the proof of Lemma 17 the strategies of the remaining players $u \notin D_S(W) \cup \{w\}$ do not change and they don't intersect W. Hence, the support of $\sigma^{T,w}$ is a tree. □

Observe that by definition $\sigma_u^{T,w} \subseteq \bar{\sigma}_u^{T,w}$ for all players u. If σ_w does not visit $W \setminus \{w\}$ we even have $\sigma_u^{T,w} = \bar{\sigma}_u^{T,w}$ for all players. Thus, $\operatorname{supp}\left(\sigma^{T,w}\right) \subseteq \operatorname{supp}\left(\bar{\sigma}^{T,w}\right)$ and in particular all edges in the support of $\sigma^{T,w}$ are used in the same direction as the edges in the support of $\bar{\sigma}^{T,w}$. Thus we observe the following.

Observation 5.

$$\Phi\left(\sigma^{T,w}\right) \leq \Phi\left(\bar{\sigma}^{T,w}\right)$$

We upper bound the change in the potential when using tree T. We introduce the notation $D_S^W(w)$ for the inverse relation to w'. That is $D_S^W(w)$ is the set of descendants of W, where w is the first node met in W, i.e., $D_S^W(w) = \{v \in D_S(W) : w'(v) = w\}$. Further, we need the restriction of $n_\sigma(e)$ to a subset of players. We define $n_\sigma^U(e) = |\{u \in U : e \in \sigma_u\}|$.

Lemma 19. *Consider a uniform broadcast game with underlying function f. Let σ be a profile with support S which is a tree. Let $W \subseteq V$, $w \in V$ and T be a tree spanning $W \cup \{w\}$. Then*

$$\Phi\left(\sigma^{T,w}\right) - \Phi(\sigma) \leq \Phi\left(\bar{\sigma}^{T,w}\right) - \Phi(\sigma) \leq \sum_{v \in W} \left|D_S^W(v)\right| \left(c_\sigma^{+1}(S[w \nearrow v]) - c_\sigma(S[v \nearrow w])\right) + \sum_{e \in T \setminus \sigma_w} F\left(n_{\bar{\sigma}^{T,w}}^{D_S(W)}(e)\right)\gamma_e.$$

For an intuition of this upper bound consider the case, where $\sigma^{T,w} = \bar{\sigma}^{T,w}$ and none of the edges of T are already used in S. We look at edges of the support that changed. These are edges inside of W and edges above W (closer to the root in S). The edges in W are those of T and they are now used by the players in $D_S(W)$. This gives the second sum. On the other hand we use that Φ is an exact potential, meaning that the change in the potential under a single player deviation is exactly the change in the player cost. Now consider some player $v \in W$ and the profile where only v switched. Then the change in the cost of player v is $c_\sigma^{+1}(S[w \nearrow v]) - c_\sigma(S[v \nearrow w])$. For the descendants of v, which are now connecting to v, the part of the strategy which changes is exactly the same as for v. Since the number of players on edges currently used only decreased by v leaving, and the number of players on new edges increased as v went there, the change in the player cost of a descendant is upper bounded by the respective difference of v. This explains the first sum, where the players in $D_S(W)$ are grouped by their w'. The other case where edges of T have already been used in S has to be handled differently as shown in the following proof.

Proof of Lemma 19. We show the bound for $\Phi\left(\bar{\sigma}^{T,w}\right) - \Phi(\sigma)$. By Observation 5 we get the first inequality.

We abbreviate $\bar{\sigma}^{T,v}$ by σ'. To show the bound we look at the contribution of each edge, i.e., we bound $F(n_{\sigma'}(e)) - F(n_\sigma(e))$. We group edges into three groups. First, we consider σ_w, then $S \setminus (\sigma_w \cup T)$, and finally $T \setminus \sigma_w$.

Edges in σ_w. Every player who changed her strategy, i.e., $\sigma'_u \neq \sigma_u$, is now using all of σ_w by definition of σ'. Thus for edges in σ_w we have $n_{\sigma'}(e) \geq n_\sigma(e)$. Since f is decreasing,

we get for every $e \in \sigma_w$:

$$\begin{aligned} F(n_{\sigma'}(e)) - F(n_\sigma(e)) &= \sum_{k=n_\sigma(e)+1}^{n_{\sigma'}(e)} f(k) \\ &\leq (n_{\sigma'}(e) - n_\sigma(e)) f(n_\sigma(e)+1) \\ &= \left|\{u \in D_S(W) : e \in \sigma'_u \setminus \sigma_u\}\right| f(n_\sigma(e)+1). \end{aligned}$$

Since S is a tree we can group the players in $D_S(W)$ not using e previously by their w'. If w' uses e already in σ, then also all of its descendants use e. On the other hand, if w' did not use e, then none of the descendants can have used e before. Hence, we can rewrite the right hand side as

$$\sum_{\substack{v \in W \\ e \in \sigma'_v \setminus \sigma_v}} \left|D_S^W(v)\right| f(n_\sigma(e)+1). \tag{4.3}$$

Edges in $S \setminus (\sigma_w \cup T)$. For edges in $S \setminus (\sigma_w \cup T)$ we have the opposite relation $n_{\sigma'}(e) \leq n_\sigma(e)$, since no edges outside of T and σ_w are added in σ'. Similarly to the previous case we obtain

$$\begin{aligned} F(n_{\sigma'}(e)) - F(n_\sigma(e)) &= -\sum_{k=n_{\sigma'}(e)+1}^{n_\sigma(e)} f(k) \\ &\leq -(n_\sigma(e) - n_{\sigma'}(e)) f(n_\sigma(e)) \\ &= -\left|\{u \in D_S(W) : e \in \sigma_u \setminus \sigma'_u\}\right| f(n_\sigma(e)) \end{aligned}$$

since f is decreasing. Again, we group players in $D_S(W)$ by their w', since $\sigma_v \setminus \sigma'_v = \sigma_u \setminus \sigma'_u$ for all $v \in W$ and $u \in D_S^W(v)$. Since $e \in S \setminus (\sigma_w \cup T)$ and $\sigma'_v \subseteq \sigma_w \cup T$, we get $e \in \sigma_v \setminus \sigma'_v \Leftrightarrow e \in \sigma_v$. In total, the right hand side can be rewritten to

$$-\sum_{\substack{v \in W \\ e \in \sigma_v}} \left|D_S^W(v)\right| f(n_\sigma(e)). \tag{4.4}$$

Edges in $T \setminus \sigma_w$. For edges e in $T \setminus \sigma_w$ we do not know the relation of $n_{\sigma'}(e)$ and $n_\sigma(e)$. Instead, we consider the strategies of players in $V \setminus D_S(W)$, which are the same for σ' and σ. In particular, $n_{\sigma'}^{V \setminus D_S(W)}(e) = n_\sigma^{V \setminus D_S(W)}(e)$. We have

$$\begin{aligned} F(n_{\sigma'}(e)) - F(n_\sigma(e)) &= F(n_{\sigma'}(e)) - F\left(n_{\sigma'}^{V \setminus D_S(W)}(e)\right) + F\left(n_\sigma^{V \setminus D_S(W)}(e)\right) - F(n_\sigma(e)) \\ &= \sum_{k=n_{\sigma'}^{V \setminus D_S(W)}(e)+1}^{n_{\sigma'}(e)} f(k) \qquad - \sum_{k=n_\sigma^{V \setminus D_S(W)}(e)+1}^{n_\sigma(e)} f(k). \end{aligned}$$

Since $n_{\sigma'}(e) = n_{\sigma'}^{D_S(W)}(e) + n_{\sigma'}^{V \setminus D_S(W)}(e)$, there are exactly $n_{\sigma'}^{D_S(W)}(e)$ terms in the first sum. The same holds for σ and the second sum. Since f is decreasing every $f(k)$ is

upper bounded by $f(k - n_{\sigma'}^{V \setminus D_S(W)}(e))$ in the first sum. For the second sum we bound every $f(k)$ from below by $f(n_\sigma(e))$. Together, we obtain the upper bound

$$F\Big(n_{\sigma'}^{D_S(W)}(e)\Big) - n_\sigma^{D_S(W)}(e) f(n_\sigma(e)) \leq F\Big(n_{\sigma'}^{D_S(W)}(e)\Big) - |\{u \in D_S(W) : e \in \sigma_u\}| f(n_\sigma(e)).$$

For every $v \in W$ using e in σ, also all descendants use e. Thus in total we get

$$F(n_{\sigma'}(e)) - F(n_\sigma(e)) \leq F\Big(n_{\sigma'}^{D_S(W)}(e)\Big) - \sum_{\substack{v \in W \\ e \in \sigma_v}} \Big|D_S^W(v)\Big| f(n_\sigma(e)). \tag{4.5}$$

We combine upper bounds (4.3), (4.4) and (4.5) to

$$\begin{aligned}
&\Phi(\sigma') - \Phi(\sigma) - \sum_{e \in T \setminus \sigma_w} \gamma_e F\Big(n_{\sigma'}^{D_S(W)}(e)\Big) \\
&= \sum_{e \in S \cup T} \gamma_e (F(n_{\sigma'}(e)) - F(n_\sigma(e))) \qquad - \sum_{e \in T \setminus \sigma_w} \gamma_e F\Big(n_{\sigma'}^{D_S(W)}(e)\Big) \\
&\leq \sum_{e \in \sigma_w} \sum_{\substack{v \in W \\ e \in \sigma'_v \setminus \sigma_v}} \gamma_e \Big|D_S^W(v)\Big| f(n_\sigma(e) + 1) \quad - \sum_{e \in (S \cup T) \setminus \sigma_w} \sum_{\substack{v \in W \\ e \in \sigma_v}} \gamma_e \Big|D_S^W(v)\Big| f(n_\sigma(e)).
\end{aligned}$$

Since $\gamma_e f(n_\sigma(e)) = c_\sigma(e)$ and $\gamma_e f(n_\sigma(e) + 1) = c_\sigma^{+1}(e)$ we continue with

$$\sum_{e \in \sigma_w} \sum_{\substack{v \in W \\ e \in \sigma'_v \setminus \sigma_v}} \Big|D_S^W(v)\Big| c_\sigma^{+1}(e) - \sum_{e \in (S \cup T) \setminus \sigma_w} \sum_{\substack{v \in W \\ e \in \sigma_v}} \Big|D_S^W(v)\Big| c_\sigma(e).$$

Since $\sigma_w \subseteq \sigma'_v$ for any $v \in W$ and $\sigma_v \subseteq S \cup T$, swapping the order of summation in both sums yields for the right hand side

$$\sum_{v \in W} \Big|D_S^W(v)\Big| c_\sigma^{+1}(\sigma_w \setminus \sigma_v) - \sum_{v \in W} \Big|D_S^W(v)\Big| c_\sigma(\sigma_v \setminus \sigma_w).$$

From $\sigma_w \setminus \sigma_v = S[w \nearrow v]$ and $\sigma_v \setminus \sigma_w = S[v \nearrow w]$, we obtain the statement by rearranging the terms. □

This is the algorithmic part of homogenization and absorption: using parts of an optimum tree. We gave an upper bound on the change of the potential in such a step. Now we introduce the distance notion used to define homogeneous profiles and to specify the absorption- and charging-radii.

The distance is measured w.r.t. a spanning tree T and a decreasing function. We consider the underlying function f of the broadcast game, and the average of the first values $\bar{f}$. Consider two nodes u and v and the path $T[u, v]$ connecting them in T. Let $e_1, \ldots, e_k$ be the edges of $T[u, v]$ ordered from u to v (that is, e_1 is incident to u and e_k is incident to v). The cost incurred to player u along this path is upper bounded by $\sum_{i=1}^{k} \gamma_{e_i} f(i)$. Since T is a tree at least the players on the path $T[u, v]$ join u, such that at least i players are using edge e_i. This bound is used as one of the two distances of interest.

We define the *unidirectional broadcast distance* $\vec{d}_{T,g}(u,v)$ w.r.t. a spanning tree T and a decreasing function $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$. Let $T[u,v]$ be the path connecting u and v in T and order the edges according to their position in $T[u,v]$, such that $T[u,v] = e_1, \ldots, e_k$ where e_1 is incident to u and e_k is incident to v. Then set

$$\vec{d}_{T,g}(u,v) = \sum_{i=1}^{k} \gamma_{e_i} g(i).$$

For $u = v$ we set $\vec{d}_{T,g}(u,v) = 0$.

Notice that $\vec{d}_{T,g}$ is not necessarily a metric. Consider the path $T[u,v] = u - v - w$ where $\gamma_{\{u,v\}} = 3$ and $\gamma_{\{v,w\}} = 1$ with underlying function $f(i) = \frac{1}{i}$ (fair cost allocation). Observe that $\vec{d}_{T,f}$ is not symmetric, as $\vec{d}_{T,f}(u,w) = 3 + \frac{1}{3}$ while $\vec{d}_{T,f}(w,u) = 1 + \frac{3}{3}$. Further, $\vec{d}_{T,f}$ is not monotone along paths, i.e., $\vec{d}_{T,f}(v,u) = 3 \geq \vec{d}_{T,f}(w,u)$.

Nevertheless, the unidirectional broadcast distance is used to define homogenization. To still use the properties of a metric, we introduce another distance which is an upper bound on the unidirectional distance and a metric.

For a tree T and a decreasing function g as before we define the *bidirectional broadcast distance* $d_{T,g}(u,v)$. Here we order the edges of T by decreasing γ, i.e., we have $\gamma_{e_1} \geq \ldots \geq \gamma_{e_k}$ and set as before

$$d_{T,g}(u,v) = \sum_{i=1}^{k} \gamma_{e_i} g(i).$$

The uni- and bidirectional distances differ only in the order of the edges on path $T[u,v]$ and give an upper bound on the cost incurred to the players on this path.

We show that the bidirectional broadcast distance is a metric.

Lemma 20. *Let T be a spanning tree of V and $g : \mathbb{N} \to \mathbb{R}_{>0}$ a decreasing function. Then for all $u, v, w \in V$ the following are satisfied*

(i) $d_{T,g}(u,v) \geq 0$
(ii) $d_{T,g}(u,v) = d_{T,g}(v,u)$
(iii) $d_{T,g}(u,w) \leq d_{T,g}(u,v) + d_{T,g}(v,w)$
(iv) $\forall v \in T[u,w] : d_{T,g}(u,v) \leq d_{T,g}(u,w)$
(v) $\vec{d}_{T,g}(u,v) \leq d_{T,g}(u,v)$
(vi) $\vec{d}_{T,g}(u,v) \leq \vec{d}_{T,\bar{g}}(u,v)$ *and* $d_{T,g}(u,v) \leq d_{T,\bar{g}}(u,v)$

Proof. (i) and (ii) follow from the definition of $d_{T,g}$. (vi) holds since g is decreasing and hence $g(i) \leq \bar{g}(i)$.

For the remaining items we use the following observation. The term $\gamma_e g(i)$ increases if either γ_e stays the same and i decreases (since g is decreasing) or i stays the same and γ_e increases. For (iii) fix three nodes u, v, w and observe that since T is a tree we have $T[u,w] \subseteq T[u,v] \cup T[v,w]$. Order the edges of $T[u,w]$ by decreasing γ as in the definition of $d_{T,g}(u,w)$ and consider one of the terms $\gamma_{e_i} g(i)$. Since all edges of $T[u,w]$ appear in $T[u,v]$ or $T[v,w]$ there is an index j such that $\gamma_{e_i} g(j)$ appears on the right hand side.

If $j \leq i$, then by our observation in the beginning, $\gamma_{e_i} g(i) \leq \gamma_{e_i} g(j)$. On the other hand, if $j > i$, the edge e_i appears later in one of the paths $T[u, v]$ or $T[v, w]$ w.r.t. γ. That is, there is an edge e' at position i in $d_{T,g}(u, v)$ or $d_{T,g}(v, w)$ with $\gamma_{e'} \geq \gamma$. Again from the discussion above we obtain $\gamma_{e_i} g(i) \leq \gamma_{e'} g(i)$.

This shows that all summands of the left hand side are dominated by terms appearing on the right hand side. As $T[u, w] \subseteq T[u, v] \cup T[v, w]$, the mapping of summands on the left to dominating terms on the right can be made injective, concluding the proof of (iii).

The monotonicity in (iv) follows similarly to the previous item. Since $T[v, w] \subseteq T[u, w]$ the same arguments from above can be applied to $d_{T,g}(u, v)$. The relation of $\vec{d}_{T,g}$ and $d_{T,g}$ in (v) can be shown with the same arguments. □

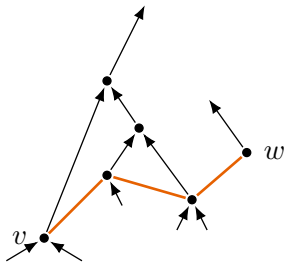
This concludes the preliminaries for our homogenization-absorption framework.

Homogenization Recall, the idea of homogenization is to reduce the potential by adding paths from the social optimum to the current profile.

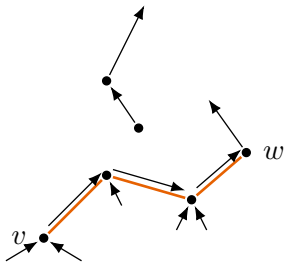
A strategy profile σ with tree support S is called *homogeneous* w.r.t. a spanning tree T, if

$$\forall v, w \in V : \Phi(\sigma) \leq \Phi\left(\sigma^{T[v \to w]}\right),$$

where $\sigma^{T[v \to w]}$ is a short notation for $\sigma^{T[v,w],w}$ as defined in (4.2), where $W = T[v, w]$. See Figure 4.12 for an illustration of $\sigma^{T[v \to w]}$.



(a) Parts of the initial profile σ, where v connects to r not using w.



(b) The profile $\sigma^{T[v \to w]}$, where v connects to w via T and all descendants of $T[v, w]$ join.

Figure 4.12: Moves considered for homogenization. The current support of σ is shown in black, where the directions point to the root. Edges in T are shown in orange.

The key property of homogeneous profiles is that the difference in player costs can be bounded by their distance. Observe that $C_v(\sigma) - C_w(\sigma) = c_\sigma(S[v \nearrow w]) - c_\sigma(S[w \nearrow v])$. Since $c(e) \geq c^{+1}(e)$ we can bound the difference of the player costs as $C_v(\sigma) - C_w(\sigma) \leq c_\sigma(S[v \nearrow w]) - c^{+1}_\sigma(S[w \nearrow v])$.

Lemma 21. *Consider a uniform broadcast game with underlying function f. In a profile σ with tree support S which is homogeneous w.r.t. a spanning tree T, we have*

$$\forall v, w \in V : c_\sigma(S[v \nearrow w]) - c^{+1}_\sigma(S[w \nearrow v]) \leq \vec{d}_{T,\bar{f}}(v, w).$$

Proof. We use the inequalities of Lemma 19 to set up a linear program. The statement then follows by weak duality.

Let v, w be two players and consider the path $T[v,w]$. We order the k edges of $T[v,w]$ from v to w and call the intermediate nodes v_i. In particular, we have $v_0 = v$, $e_i = \{v_{i-1}, v_i\}$ and $v_k = w$.

Since σ is homogeneous w.r.t. T the profile $\sigma^{T[v\to w]}$ does not have smaller potential than σ. This holds for any pair of players. In particular, for all pairs (v, v_i), we have $0 \le \Phi\left(\sigma^{T[v\to v_i]}\right) - \Phi(\sigma)$. On the other hand, we can apply Lemma 19 to each of the sets $W = \{v, \dots, v_i\}$ with $w = v_i$ and $T = T[v, v_i]$ and obtain

$$\begin{aligned} 0 &\le \Phi\left(\sigma^{T[v\to v_i]}\right) - \Phi(\sigma) \qquad (4.6)\\ &\le \sum_{j=0}^{i} \left|D_S^{T[v,v_i]}(v_j)\right| \left(c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])\right) + \sum_{j=1}^{i} \gamma_{e_j} F\left(n_{\bar\sigma^{T[v\to v_i]}}^{D_S(T[v,v_i])}(e_j)\right). \end{aligned}$$

Notice that in the second summand we sum over all edges of $T[v, v_i]$ and do not exclude those contained in σ_{v_i} as in Lemma 19, which gives a weaker but still valid bound.

We set

$$n_j^{(i)} = \left|D_S^{T[v,v_i]}(v_j)\right| \quad \text{and} \quad N_j^{(i)} = \sum_{k=0}^{j} n_k^{(i)}.$$

Observe that $N_{j-1}^{(i)}$ is exactly the number of players in $D_S(T[v,v_i])$ using edge e_j in $\bar\sigma^{T[v\to v_i]}$.

The LP. Define the linear program with $k+1$ variables C_i as

$$\begin{aligned} \max \quad & C_0 - C_k \\ \text{s.t.} \quad & \forall i \in \{1,\dots,k\} : \sum_{j=0}^{i} n_j^{(i)}(C_j - C_i) \le \sum_{j=1}^{i} \gamma_{e_j} F\left(N_{j-1}^{(i)}\right) \\ & \forall i \in \{0,\dots,k\} : \qquad 0 \le C_i. \end{aligned}$$

With the bound $C_{v_i}(\sigma) - C_{v_j}(\sigma) \ge c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])$ we see from (4.6) that the player costs $C_{v_i}(\sigma)$ are a feasible solution to the LP. For this choice we get as objective function value $C_0 - C_k = C_v(\sigma) - C_w(\sigma)$ which is a lower bound on the right hand side of the statement as discussed before the lemma.

The dual LP. We show an upper bound on this value by looking at the dual problem with k variables y_i:

$$\begin{aligned} \min \quad & \sum_{j=1}^{k}\sum_{i=j}^{k} \gamma_{e_j} F\left(N_{j-1}^{(i)}\right) y_i \\ \text{s.t.} \quad & \sum_{i=1}^{k} n_0^{(i)} y_i \ge 1 \end{aligned}$$

$$\begin{aligned} \forall i \in \{1,\dots,k-1\} : \sum_{j=i+1}^{k} n_i^{(j)} y_j - N_{i-1}^{(i)} y_i &\geq 0 \\ -N_{k-1}^{(k)} y_k &\geq -1 \\ \forall i \in \{1,\dots,k\} : \quad 0 &\leq y_i. \end{aligned}$$

Adding all of the constraints except for the non-negativity constraints gives zero on both sides. Hence, a feasible solution has to satisfy all of them with equality. We can thus define the unique solution recursively starting from $\hat{y}_k = \frac{1}{N_{k-1}^{(k)}}$ and for $i \in \{1,\dots,k-1\}$

$$\hat{y}_i = \frac{1}{N_{i-1}^{(i)}} \left(1 - \sum_{j=i+1}^{k} N_{i-1}^{(j)} y_j\right).$$

Adding the inequalities for $j = i,\dots,k$ gives $\sum_{j=i}^{k} N_{i-1}^{(j)} y_j \leq 1$, and hence $\hat{y}$ is indeed feasible for the dual LP. We compute the dual objective function value for $\hat{y}$ as

$$\sum_{j=1}^{k}\sum_{i=j}^{k} \gamma_{e_j} F\left(N_{j-1}^{(i)}\right)\hat{y}_i = \sum_{j=1}^{k}\sum_{i=j}^{k} \gamma_{e_j} \bar{f}\left(N_{j-1}^{(i)}\right) N_{j-1}^{(i)} \hat{y}_i,$$

where we used $F(k) = k\bar{f}(k)$. We use $\max_{i=j,\dots,k} \bar{f}\left(N_{j-1}^{(i)}\right)$ as upper bound for each of the terms $\bar{f}\left(N_{j-1}^{(i)}\right)$. Since f is decreasing also $\bar{f}$ is decreasing. Hence, the maximum is attained at the smallest value $N_{j-1}^{(i)}$ for $i = j\dots,k$. For $j \leq i$ we have $n_\ell^{(j)} \geq n_\ell^{(i)}$ for all $\ell \leq j-1$ by the following argument. $n_\ell^{(j)}$ counts the number of descendants of $T[v,v_j]$ who have v_ℓ as their first node on this path. If we now consider the longer path $T[v,v_i]$ only fewer players can have v_ℓ as their first node on this path, since some of $D_S^{T[v,v_j]}(v_\ell)$ may meet $T[v,v_i]$ on some other node than v_ℓ. See Figure 4.13 for an illustration. Thus, $n_\ell^{(i)}$ is decreasing in i, and hence $\max_{i=j,\dots,k} \bar{f}\left(N_{j-1}^{(i)}\right)$ is attained at $i = k$.

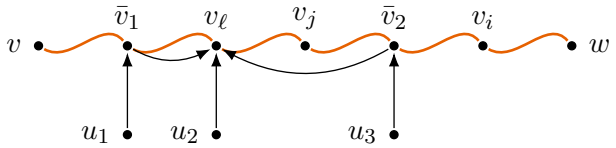


Figure 4.13: Illustration for $n_\ell^{(j)} \geq n_\ell^{(i)}$ for $j \leq i$ used in the proof of Lemma 21. The path $T[v,w]$ is shown in orange. Parts of tree S are drawn in black. u_2 is always in $D_S^{T[v,v_k]}(v_\ell)$ for all $k \geq \ell$, while u_1 is never a member of $D_S^{T[v,v_k]}(v_\ell)$. For u_3 we have $u_3 \in D_S^{T[v,v_j]}(v_\ell)$ since v_ℓ is the first node on $T[v,v_j]$ met by u_3. On the other hand $u_3 \notin D_S^{T[v,v_i]}(v_\ell)$ as u_3 meets $T[v,v_i]$ at $\bar{v}_2$.

Further, we use the bound $D_S^{T[v,w]}(v_\ell) \geq 1$, as in the worst case only v_ℓ is in this set and thus $\bar{f}\left(N_{j-1}^{(k)}\right) \leq \bar{f}(j)$. Together with $\sum_{j=i}^{k} N_{i-1}^{(j)} \hat{y}_j \leq 1$, we obtain the upper bound

$$\sum_{j=1}^{k} \gamma_{e_j} \max_{i=j,\dots,k} \bar{f}\left(N_{j-1}^{(i)}\right) \leq \sum_{j=1}^{k} \gamma_{e_j} \bar{f}(j) = \vec{d}_{T,\bar{f}}(v,w).$$

By weak duality we thus have

$$C_v(\sigma) - C_w(\sigma) \leq \vec{d}_{T,\bar{f}}(v,w),$$

which is almost what we want.

A primal feasible solution. To show the statement, we give another feasible solution to the (primal) LP, whose objective function value is the left hand side of the inequality to be shown, and the bound follows as before from weak duality.

Define

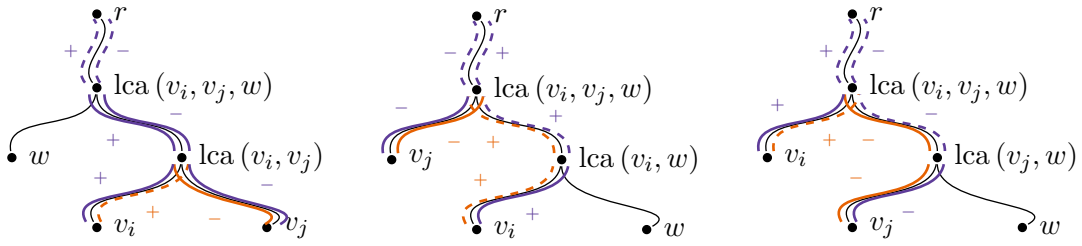
$$\hat{C}_i = c_\sigma^{+1}(\sigma_{v_i} \cap \sigma_w) + c_\sigma(\sigma_{v_i} \setminus \sigma_w)$$

for $i \in \{0,\dots,k\}$. For this choice the primal objective evaluates to

$$\begin{aligned}\hat{C}_0 - \hat{C}_k &= c_\sigma^{+1}(\sigma_v \cap \sigma_w) + c_\sigma(\sigma_v \setminus \sigma_w) - c_\sigma^{+1}(\sigma_w \cap \sigma_w) - c_\sigma(\sigma_w \setminus \sigma_w) \\ &= c_\sigma(\sigma_v \setminus \sigma_w) - c_\sigma^{+1}(\sigma_w \setminus \sigma_v)\end{aligned}$$

which is the left hand side of the statement.

Hence, it remains to show that $\hat{C}$ is feasible for the primal LP. As for feasibility of the vector of the actual player costs, we prove this by showing $\hat{C}_i - \hat{C}_j \geq c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])$ and using (4.6). There are structurally two possibilities how the lowest common ancestors of (v_i, v_j) and (v_i, v_j, w) are placed relative to each other in S. Either $\mathrm{lca}\,(v_i, v_j, w) = \mathrm{lca}\,(v_i, v_j)$, or $\mathrm{lca}\,(v_i, v_j, w) \neq \mathrm{lca}\,(v_i, v_j)$ (see Figure 4.14).



(a) $\mathrm{lca}\,(v_i, v_j, w) \neq \mathrm{lca}\,(v_i, v_j)$

(b) $\mathrm{lca}\,(v_i, v_j, w) = \mathrm{lca}\,(v_i, v_j)$ and $\mathrm{lca}\,(v_i, w)$ is below $\mathrm{lca}\,(v_i, v_j, w)$

(c) $\mathrm{lca}\,(v_i, v_j, w) = \mathrm{lca}\,(v_i, v_j)$ and $\mathrm{lca}\,(v_j, w)$ is below $\mathrm{lca}\,(v_i, v_j, w)$

Figure 4.14: Illustration for proof of $\hat{C}_i - \hat{C}_j \geq c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])$ used in Lemma 21. Paths contributing to terms on the left hand side are shown in purple and paths appearing on the right hand side in orange. Edges where c_σ is considered are marked solid, while edges where we consider c_σ^{+1} are marked with dashes. The label on these marks give the sign of the corresponding term.

Consider the case where $\operatorname{lca}(v_i, v_j, w) \neq \operatorname{lca}(v_i, v_j)$. As $\operatorname{lca}(v_i, v_j, w)$ is closer to the root than $\operatorname{lca}(v_i, v_j)$, we have the situation as shown in Figure 4.14a. In particular, we have $\operatorname{lca}(v_i, v_j, w) = \operatorname{lca}(v_i, w) = \operatorname{lca}(v_j, w)$. Paths appearing on the right hand side $c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])$ are marked with orange. Parts where we use c_σ^{+1} are marked with dashes and parts with c_σ are marked solid. For the left hand side we compute

$$\hat{C}_i - \hat{C}_j = c_\sigma^{+1}(\sigma_{v_i} \cap \sigma_w) + c_\sigma(\sigma_{v_i} \setminus \sigma_w) - c_\sigma^{+1}\Big(\sigma_{v_j} \cap \sigma_w\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_w\Big).$$

The respective paths are shown in purple. For example $c_\sigma\Big(\sigma_{v_j} \setminus \sigma_w\Big)$ corresponds to the solid purple marks with label "$-$" from v_j to $\operatorname{lca}(v_j, v_i)$ and from $\operatorname{lca}(v_j, v_i)$ to $\operatorname{lca}(v_i, v_j, w)$. We observe from the picture that the only difference in the left and right hand side is the path from v_i to $\operatorname{lca}(v_i, v_j, w)$, where we have c_σ on the left and c_σ^{+1} on the right. Since $c_\sigma^{+1}(e) \leq c_\sigma(e)$, this shows that $\hat{C}_i - \hat{C}_j \geq c_\sigma^{+1}(S[v_i \nearrow v_j]) - c_\sigma(S[v_j \nearrow v_i])$ in this case.

Formally, we write

$$\begin{aligned} \sigma_{v_i} \cap \sigma_w &= S[\operatorname{lca}(v_i, w), \operatorname{lca}(v_i, v_j, w)] \cup S[\operatorname{lca}(v_i, v_j, w), r] \\ \text{and} \quad \sigma_{v_i} \setminus \sigma_w &= S[v_i, \operatorname{lca}(v_i, v_j)] \cup S[\operatorname{lca}(v_i, v_j), \operatorname{lca}(v_i, v_j, w)] \end{aligned} \tag{4.7}$$

and similarly for $\sigma_{v_j} \cap \sigma_w$ and $\sigma_{v_j} \setminus \sigma_w$ replacing v_i by v_j. We thus have

$$\begin{aligned} \hat{C}_i - \hat{C}_j =& c_\sigma^{+1}(\sigma_{v_i} \cap \sigma_w) + c_\sigma(\sigma_{v_i} \setminus \sigma_w) - c_\sigma^{+1}\Big(\sigma_{v_j} \cap \sigma_w\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_w\Big) \\ =& c_\sigma(S[v_i, \operatorname{lca}(v_i, v_j)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i)]) \end{aligned}$$

since $\operatorname{lca}(v_i, w) = \operatorname{lca}(v_j, w)$. As $c_\sigma(e) \geq c_\sigma^{+1}(e)$ we get the bound

$$\begin{aligned} \hat{C}_i - \hat{C}_j &= c_\sigma(S[v_i, \operatorname{lca}(v_i, v_j)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i)]) \\ &\geq c_\sigma^{+1}(S[v_i, \operatorname{lca}(v_i, v_j)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i)]) \\ &= c_\sigma^{+1}\Big(\sigma_{v_i} \setminus \sigma_{v_j}\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_{v_i}\Big). \end{aligned}$$

For the other case where $\operatorname{lca}(v_i, v_j, w) = \operatorname{lca}(v_i, v_j)$ we consider two subcases. First, assume $\operatorname{lca}(v_j, w)$ is closer to the root than $\operatorname{lca}(v_i, w)$ (see Figure 4.14b). Then we have $\operatorname{lca}(v_i, v_j, w) = \operatorname{lca}(v_j, w)$, as $\operatorname{lca}(v_i, w)$ is a descendant of $\operatorname{lca}(v_j, w)$ both v_i and w are descendants of $\operatorname{lca}(v_j, w)$. With (4.7) and $S[v_i, \operatorname{lca}(v_i, v_j)] = S[v_i, \operatorname{lca}(v_i, w)] \cup S[(\operatorname{lca}(v_i, w), \operatorname{lca}(v_i, v_j, w))]$ we compute

$$\begin{aligned} \hat{C}_i - \hat{C}_j = \quad & c_\sigma^{+1}(\sigma_{v_i} \cap \sigma_w) + c_\sigma(\sigma_{v_i} \setminus \sigma_w) - c_\sigma^{+1}\Big(\sigma_{v_j} \cap \sigma_w\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_w\Big) \\ = \quad & c_\sigma^{+1}(S[\operatorname{lca}(v_i, w), \operatorname{lca}(v_i, v_j, w)]) + c_\sigma(S[v_i, \operatorname{lca}(v_i, w)]) \\ + \quad & c_\sigma(S[\operatorname{lca}(v_i, w), \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i, w)]) \\ \geq \quad & c_\sigma^{+1}(S[\operatorname{lca}(v_i, w), \operatorname{lca}(v_i, v_j, w)]) + c_\sigma^{+1}(S[v_i, \operatorname{lca}(v_i, w)]) \\ & - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i, w)]) \end{aligned}$$

$$
\begin{aligned}
&= c_\sigma^{+1}(S[v_i, \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, v_i, w)]) \\
&= c_\sigma^{+1}\Big(\sigma_{v_i} \setminus \sigma_{v_j}\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_{v_i}\Big).
\end{aligned}
$$

In the remaining subcase, where $\operatorname{lca}(v_i, v_j, w) = \operatorname{lca}(v_i, v_j)$ and $\operatorname{lca}(v_j, w)$ is a descendant of $\operatorname{lca}(v_i, w)$, we have similar to before $\operatorname{lca}(v_i, v_j, w) = \operatorname{lca}(v_i, w)$ (see Figure 4.14c). Further, $\sigma_{v_j} \setminus \sigma_w = S[v_j, \operatorname{lca}(v_j, w)]$ and thus

$$
\begin{aligned}
\hat{C}_i - \hat{C}_j =& c_\sigma^{+1}(\sigma_{v_i} \cap \sigma_w) + c_\sigma(\sigma_{v_i} \setminus \sigma_w) - c_\sigma^{+1}\Big(\sigma_{v_j} \cap \sigma_w\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_w\Big) \\
=& c_\sigma(S[v_i, \operatorname{lca}(v_i, v_j, w)]) - c_\sigma^{+1}(S[\operatorname{lca}(v_j, w), \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, w)]) \\
\geq& c_\sigma^{+1}(S[v_i, \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[\operatorname{lca}(v_j, w), \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[v_j, \operatorname{lca}(v_j, w)]) \\
=& c_\sigma^{+1}(S[v_i, \operatorname{lca}(v_i, v_j, w)]) - c_\sigma(S[v_j, \operatorname{lca}(v_i, v_j, w)]) \\
=& c_\sigma^{+1}\Big(\sigma_{v_i} \setminus \sigma_{v_j}\Big) - c_\sigma\Big(\sigma_{v_j} \setminus \sigma_{v_i}\Big).
\end{aligned}
$$

This shows that $\hat{C}$ is a feasible solution with objective value being the left hand side of the inequality in the statement. Together with the bound from the feasible solution to the dual LP, the proof is complete. □

Homogenization is the process of repeatedly checking all player pairs (v, w) and changing to $\sigma^{T[v \to w]}$ if this decreases the potential until no improvement is available.

```
1 Function Homogenize(T, σ)                          -- Homogenize σ w.r.t. T
2     while ∃w, z ∈ V : Φ(σ^{T[w→z]}) < Φ(σ) do
3         σ ← σ^{T[w→z]}
4     end
```

Function Homogenize(T, σ)

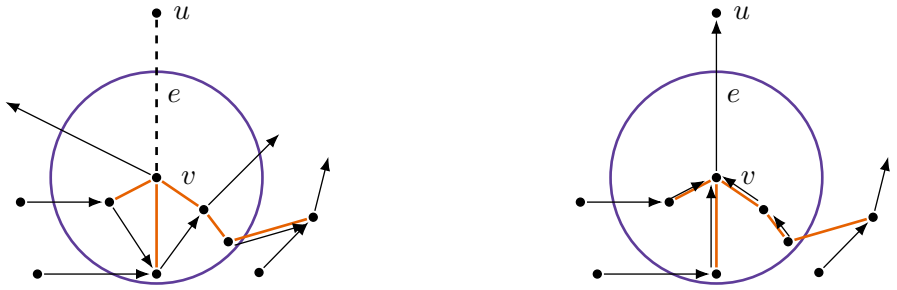
The procedure terminates as in every while-iteration the potential strictly decreases and there are only finitely many profiles. The resulting profile is homogeneous by the condition of the while-loop and has tree support by Lemma 18.

Absorption In contrast to the global nature of homogenization, we consider only local replacements in absorption. If a player deviates and introduces a new edge outside of the social optimum, we absorb nearby players using the social optimum. The goal is to remove edges from the support that are not in the social optimum while not increasing the potential.

Consider a profile σ with tree support S and assume that player v has an improving tree-move (Lemma 11). That is, there is an edge $e = \{v, u\}$ such that $c_\sigma(S[v \nearrow u]) > \gamma_e f(1) + c_\sigma^{+1}(S[u \nearrow v])$. We define the *absorption-ball* at v w.r.t. e in terms of the bidirectional broadcast distance w.r.t. a spanning tree T in $B(v, e)$ as

$$
B(v, e) = \left\{ w \in V : d_{T, \bar{f}}(v, w) \leq \frac{f(1) - f(2)}{2} \gamma_e \right\}.
$$

Note that since $d_{T,\bar{f}}$ is a metric (see Lemma 20) this set is indeed a ball (a connected set without holes). If $w \in B(v,e)$, then also all $w' \in T[w,v]$ are in $B(v,e)$, and hence the subgraph of T induced by nodes in $B(v,e)$ is a tree. We define the short notation $r_{\text{absorb}} = \frac{f(1)-f(2)}{2}$ for the *absorption-radius*.



(a) Before the absorption (σ). v has an improving tree-move using e (dashed).

(b) $\sigma^{T[B(v,e)]}$ after the absorption. The 5 players in the absorption-ball and their descendants changed their strategy, removing 4 edges outside of the social optimum.

Figure 4.15: An absorption step at v w.r.t. $e = \{u,v\}$. Parts of the support of σ are shown in black. The arrows give the direction to the root in this tree. Parts of tree T are shown in orange. The absorption-ball $B(v,e)$ is shown in purple.

Absorption is the process of letting v deviate to u using e and changing the strategy of every player in the absorption-ball and their descendants to connect to v via T. We denote the resulting profile by $\sigma^{T[B(v,e)]}$. The strategy of a player $w \in D_S(B(v,e))$ is

$$S[w,w'(w)] \cup T[w'(w),v] \cup \{e\} \cup S[u,r] \tag{4.8}$$

where we use again the notation $w'(w)$ for the first node met in T by w. The strategies of other players do not change. See Figure 4.15 for an illustration of $\sigma^{T[B(v,e)]}$.

```
1 Function Absorb(T, σ, v, e)          -- Absorb σ at v w.r.t. e and T
2     return σ^{T[B(v,e)]}
```

Function Absorb(T, σ, v, e)

We consider absorption only in homogeneous profiles, since then the support of the new profile is again a tree.

Lemma 22. *For a profile σ with tree support S that is homogeneous w.r.t. T, and an edge $e = \{u,v\} \in E \setminus (S \cup T)$ with $c_\sigma(S[u \nearrow v]) > \gamma_e f(1) + c_\sigma^{+1}(S[v \nearrow u])$, we have*

(i) $\sigma_u \cap B(v,e) = \emptyset$ *and*

(ii) $\sigma^{T[B(v,e)]} = \bar{\sigma}^{T|_W \cup \{e\},u}$ *as defined in (4.1), where $W = B(v,e)$ and $T|_W$ is the subgraph of T induced by nodes in W.*

Proof. For the first part consider any node $w \in \sigma_u$. From Item (vi) of Lemma 20 we have $d_{T,\bar{f}}(v,w) \geq \vec{d}_{T,\bar{f}}(v,w)$. Since we assume σ to be homogeneous w.r.t. T, we have

from Lemma 21 that

$$d_{T,\bar{f}}(v,w) \geq \vec{d}_{T,\bar{f}}(v,w) \geq c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]). \tag{4.9}$$

We show that the right hand side is lower bounded by the same term where we set $w = u$. If $w \in \sigma_v$ then $S[w \nearrow v]$ is empty and $S[v \nearrow u] \subseteq S[v \nearrow w]$. Hence

$$c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) \geq c_\sigma(S[v \nearrow u]) - c_\sigma^{+1}(S[u \nearrow v]).$$

In the other case where $w \notin \sigma_v$, we have $\mathrm{lca}(v,u) = \mathrm{lca}(v,w)$ and hence $c_\sigma(S[v \nearrow w]) = c_\sigma(S[v \nearrow u])$. Further $S[w \nearrow v] \subseteq S[u \nearrow v]$ and thus $c_\sigma^{+1}(S[w \nearrow v]) \leq c_\sigma^{+1}(S[u \nearrow v])$. In total, we also obtain

$$c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) \geq c_\sigma(S[v \nearrow u]) - c_\sigma^{+1}(S[u \nearrow v]).$$

Using this inequality and the fact that e is an improving edge for v, we continue to lower bound (4.9) with

$$c_\sigma(S[v \nearrow u]) - c_\sigma^{+1}(S[u \nearrow v]) > f(1)\gamma_e \geq \frac{f(1)-f(2)}{2}\gamma_e.$$

Putting everything together, we showed that for all $w \in S[u,r]$ the distance $d_{T,\bar{f}}(v,w)$ is strictly larger than $\gamma_e r_{\text{absorb}}$ and thus w can not lie in $B(v,e)$. Thus $\sigma_u \cap B(v,e) = \emptyset$.

For the second part, we need to show that $T|_W \cup \{e\}$ is a tree spanning $B(v,e) \cup \{u\}$, as then the definitions of $\sigma^{T[B(v,e)]}$ (4.8) and of $\bar{\sigma}^{T|_W \cup \{e\},u}$ (4.1) coincide. From the first part we know that in particular u is not in $B(v,e)$. As $T|_W$ is a tree spanning $B(v,e)$ we get that indeed $T|_W \cup \{e\}$ is a tree spanning $B(v,e) \cup \{u\}$. □

Together with Lemma 17 we immediately get the following key properties of absorption.

Observation 6. For a profile σ with tree support S that is homogeneous w.r.t. a tree T and a player v with improving tree-move using e

(i) the support of $\sigma^{T[B(v,e)]}$ is a tree and contains the part of T in $B(v,e)$ as subtree.
(ii) the parent edges $e_S(w) \in E \setminus T$ of nodes w in $B(v,e)$, are not contained in the support of $\sigma^{T[B(v,e)]}$.

We show that absorbing decreases the potential in homogeneous profiles.

Lemma 23. *Consider a uniform broadcast game with underlying function f. Let σ be a homogeneous profile w.r.t. a spanning tree T and let S be its support. Further, let $e = \{u,v\} \in E \setminus (S \cup T)$ with $c_\sigma(S[v \nearrow u]) > \gamma_e f(1) + c_\sigma^{+1}(S[u \nearrow v])$. Then,*

$$\Phi\left(\sigma^{T[B(v,e)]}\right) < \Phi(\sigma).$$

Proof. We show $\Phi\left(\sigma^{T[B(v,e)]}\right) - \Phi(\sigma) < 0$ with the upper bounds of Lemma 19 and Lemma 21. We use the representation of $\sigma^{T[B(v,e)]}$ as $\bar{\sigma}^{T|_{W\cup\{e\},u}}$, where $W = B(v,e)$ from Lemma 22. The following short notations are used in this proof. We set $D = D_S(W)$, $\sigma' = \sigma^{T[B(v,e)]} = \bar{\sigma}^{T|_{W\cup\{e\},u}}$ and $S' = \operatorname{supp}(\sigma')$. From the bound of Lemma 19 we get

$$\begin{aligned}\Phi(\sigma') - \Phi(\sigma) \leq \sum_{w\in W} \left|D_S^W(w)\right| \left(c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma(S[w \nearrow u])\right) \\ + \sum_{e'\in T\setminus\sigma_u} F\left(n_{\sigma'}^D(e')\right)\gamma_{e'} + F\left(n_{\sigma'}^D(e)\right)\gamma_e. \end{aligned} \tag{4.10}$$

To show that the right hand side is upper bounded by zero, we bound both parts – the first sum, and the remaining two summands involving F – by the sum of the unidirectional distance of $w \in W$ to v and γ_e times some $f(k)$. We proceed in two steps. First, we upper bound the difference of the costs appearing in the first sum by the unidirectional distance of w to v as

$$c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma(S[w \nearrow u]) \leq \vec{d}_{T,\bar{f}}(v,w) - \gamma_e f(1). \tag{4.11}$$

This bound is strict for v since e is an improving tree-move. Second, we bound the two remaining terms by

$$\sum_{e'\in T\setminus\sigma_u} F\left(n_{\sigma'}^D(e')\right)\gamma_{e'} + F\left(n_{\sigma'}^D(e)\right)\gamma_e \leq \sum_{w\in W} \left|D_S^W(w)\right| \left(\vec{d}_{T,f}(w,v) + \gamma_e f\left(\operatorname{pos}(e,\sigma'_w)\right)\right), \tag{4.12}$$

where $\operatorname{pos}(e,\sigma'_w)$ denotes the position of edge e in the path σ'_w starting at w. In particular, $\operatorname{pos}(e,\sigma'_v) = 1$. The statement then follows by Lemma 20 and the definition of $W = B(v,e)$, since for any $w \in W \setminus \{v\}$ we have $\operatorname{pos}(e,\sigma'_w) \geq 2$ and

$$\vec{d}_{T,\bar{f}}(v,w) - \gamma_e f(1) + \vec{d}_{T,f}(w,v) + \gamma_e f(2) \leq 2d_{T,\bar{f}}(v,w) + \gamma_e(f(2) - f(1)) \leq 0.$$

Proof of (4.11). On the left hand side the cost of w is compared to the cost of u, instead we want to relate it to the cost of v. We show

$$c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma(S[w \nearrow u]) \leq c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) - \gamma_e f(1). \tag{4.13}$$

There are three cases for the relation of $\operatorname{lca}(v,w)$ and $\operatorname{lca}(u,v,w)$ similar to the proof of Lemma 21. If $\operatorname{lca}(v,w) \neq \operatorname{lca}(u,v,w)$, we have $S[v \nearrow w] = S[v, \operatorname{lca}(v,w)]$ and $S[w \nearrow v] = S[w, \operatorname{lca}(v,w)]$. Since $c_\sigma(e) \geq c_\sigma^{+1}(e)$ we get a lower bound on the right hand side as

$$c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) - \gamma_e f(1) \geq c_\sigma(S[v \nearrow u]) - c_\sigma^{+1}(S[w \nearrow u]) - \gamma_e f(1).$$

As e is an improving tree-move for v we obtain

$$\begin{aligned} c_\sigma(S[v \nearrow u]) - c_\sigma^{+1}(S[w \nearrow u]) - \gamma_e f(1) &\geq c_\sigma^{+1}(S[u \nearrow v]) - c_\sigma^{+1}(S[w \nearrow u]) \\ &\geq c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma(S[w \nearrow u]). \end{aligned}$$

If $\operatorname{lca}(v,w) = \operatorname{lca}(u,v,w)$, consider first the case where $\operatorname{lca}(u,w)$ is a descendant of $\operatorname{lca}(u,v,w)$. We have $\operatorname{lca}(v,w) = \operatorname{lca}(v,u)$ and thus $S[v \nearrow w] = S[v \nearrow u]$. Using the condition on e as improving tree-move for v we obtain

$$c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) - \gamma_e f(1) \geq c_\sigma^{+1}(S[u \nearrow v]) - c_\sigma^{+1}(S[w \nearrow v]).$$

As $S[u \nearrow v]$ and $S[w \nearrow v]$ share the common part $S[\operatorname{lca}(u,w), \operatorname{lca}(u,v,w)]$ and since $c(e) \geq c^{+1}(e)$, we continue with

$$c_\sigma^{+1}(S[u \nearrow v]) - c_\sigma^{+1}(S[w \nearrow v]) \geq c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma(S[w \nearrow u]).$$

For the remaining case we have $\operatorname{lca}(v,w) = \operatorname{lca}(u,v,w)$ and $\operatorname{lca}(u,v)$ is a descendant of $\operatorname{lca}(u,v,w)$. Similar to before $\operatorname{lca}(v,w) = \operatorname{lca}(u,w)$. We write

$$S[v \nearrow w] = S[v \nearrow u] \cup S[\operatorname{lca}(u,v), \operatorname{lca} u, v, w]$$

and use the condition on e for the first term to obtain the bound

$$\begin{aligned}
&c_\sigma(S[v \nearrow w]) - c_\sigma^{+1}(S[w \nearrow v]) - \gamma_e f(1) \\
\geq &c_\sigma^{+1}(S[u \nearrow v]) + c_\sigma(S[\operatorname{lca}(u,v), \operatorname{lca}(u,v,w)]) - c_\sigma^{+1}(S[w \nearrow u]) \\
\geq &c_\sigma^{+1}(S[u \nearrow w]) - c_\sigma^{+1}(S[w \nearrow u]),
\end{aligned}$$

where the last inequality follows from $c(e) \geq c^{+1}(e)$.

We have thus shown (4.13) and since σ is homogeneous w.r.t. T, we get (4.11) from Lemma 21.

Proof of (4.12). For the second bound, we use that f is decreasing and upper bound $F\left(n^D_{\sigma'}(e')\right)$ by

$$\sum_{\substack{w \in D \\ e' \in \sigma'_w}} f(\operatorname{pos}(e', \sigma'_w)).$$

Since the support of σ' is a tree (Observation 6), we have $n_{\sigma'}(e') \geq \operatorname{pos}(e', \sigma'_w)$ for any w as at least all players along σ'_w before e' use e'.

Grouping players in D by their w' we get

$$F\left(n^D_{\sigma'}(e')\right) \leq \sum_{\substack{w \in W \\ e' \in \sigma'_w}} \left|D^W_S(w)\right| f(\operatorname{pos}(e', \sigma'_w)),$$

since the position of e' only increases for descendants of w and f is decreasing. Changing the order of summation on the left hand side yields

$$\begin{aligned}
&\sum_{e' \in T \setminus \sigma_u} F\left(n^D_{\sigma'}(e')\right)\gamma_{e'} + F\left(n^D_{\sigma'}(e)\right)\gamma_e \\
\leq &\sum_{w \in W} \sum_{e' \in T[w,v] \setminus \sigma_u} \left|D^W_S(w)\right| f(\operatorname{pos}(e', \sigma'_w))\gamma_{e'} + \sum_{w \in W} \left|D^W_S(w)\right| f(\operatorname{pos}(e, \sigma'_w))\gamma_e.
\end{aligned}$$

For $w \in W$ observe that

$$\sum_{e' \in T[w,v]} f(\operatorname{pos}(e', \sigma'_w))\gamma_{e'} = \vec{d}_{T,f}(w,v)$$

which proves (4.12). □

Algorithm With the definitions of homogenization and absorption we are now ready to state Algorithm 1 that computes a good equilibrium.

Input: uniform broadcast game $((V, E), r, f, \gamma)$, spanning tree T of V
Output: equilibrium σ

1 compute the stationary point of the following sequence of profiles
2 $\sigma_0 :=$ strategy profile induced by T
3 $\sigma_{i+1} :=$ **if** σ_i *is equilibrium* **then return** σ_i
4 **else**
5 **let** $e_{i+1} = \{u_{i+1}, v_{i+1}\} \in E \setminus \text{supp}(\sigma_i)$ *such that* $c_{\sigma_i}(S_i[v_{i+1} \nearrow u_{i+1}]) > c^{+1}_{\sigma_i}(S_i[u_{i+1} \nearrow v_{i+1}]) + f(1)\gamma_{e_{i+1}}$
6 **return** `Homogenize(`T, `Absorb(`T, σ_i, v_{i+1}, e_{i+1}`))`
7 **end**

Algorithm 1: Homogenization-Absorption framework to compute a good equilibrium

Theorem 8. *Algorithm 1 computes an equilibrium in finitely many steps.*

Proof. From Lemma 11 we have the existence of an edge e_{i+1} as in Line 5 for any profile σ_i not being an equilibrium. Since T is homogeneous w.r.t. itself we always have a homogeneous profile before absorption in Line 6. Hence, absorbing at v_{i+1} is feasible and results in a profile with tree support (Observation 6). The result of homogenization is also a tree (Lemma 18). Thus, the steps can be applied in this form.

From the definition of homogenization and from Lemma 23 we have that the potential strictly decreases in the `else`-branch (Line 5 - Line 6). Thus, no profile can be constructed twice by just using the `else`-branch. As there are only finitely many profiles available, the algorithm computes a stationary point after finitely many steps. This stationary point is an equilibrium from Line 3. □

Although Algorithm 1 terminates, we are not able to say anything about the running time; about how many steps are needed in the worst case. In particular, we want to remark here that the decrease in the potential is chosen to be as small as possible in every step. Intuitively, we want to absorb as many players and use as many edges of the social optimum, while still decreasing the potential. Somehow, this is contrary to fast progress towards a local minimizer of the potential. Hence, our feeling is that this algorithm may take many steps. Further, we are not defining any order of the improving tree-moves in each iteration. We do not know of any characterization of a good improving move in terms of fast stabilization.

To get an upper bound on the PoS we would like to call this algorithm with a social optimum tree. Since it is hard to compute the social optimum for sharing functions with economies of scale we instead use a minimum spanning tree w.r.t. the cost factors γ.

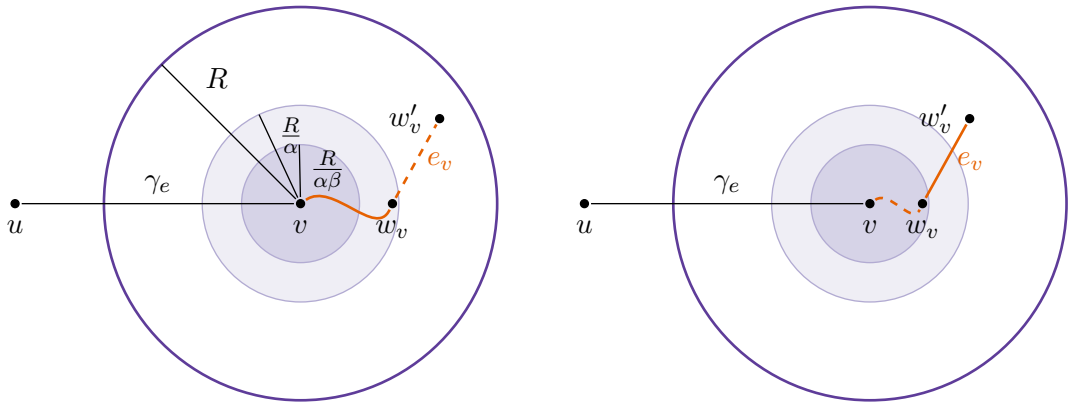
Charging-Scheme To measure the quality of the equilibrium σ computed by Algorithm 1 we compare its social cost to the social cost of tree T in the input. Our analysis

is basically the analysis for fair cost allocation, as we are ignoring sharing effects. On the one hand, we use the lower bound $C(T) \geq \sum_{e \in T} f(1)\gamma_e$ for tree T. On the other hand, we use the upper bound $n_\sigma(e) f(n_\sigma(e)) \leq \sup_{k \in \mathbb{N}} k f(k)$ for the social cost of σ. Note that for fair cost allocation both of these bounds are tight. The relation of the social costs of T and σ is obtained by splitting $C(\sigma)$ in two parts:

$$C(\sigma) \leq \sup_{k \in \mathbb{N}} \frac{k f(k)}{f(1)} \left(\sum_{e \in \mathrm{supp}(\sigma) \cap T} f(1)\gamma_e + \sum_{e \in \mathrm{supp}(\sigma) \setminus T} f(1)\gamma_e \right). \tag{4.14}$$

While the first summand is upper bounded by $C(T)$ we show that the second summand can be bounded by $C(T)$ times some factor independent of T and σ.

An edge e outside of T has been introduced to σ by an improving tree-move of some v. We charge γ_e to edges in T which are close to v. We define the *charging-radius* to be an $\alpha > 2$ fraction of the absorption-radius around v, i.e., the charging-radius is $r_{\text{charge}} = \frac{r_{\text{absorb}}}{\alpha} = \frac{f(1)-f(2)}{2\alpha}\gamma_e$ where we use again $d_{T,\bar{f}}$ as distance. In particular, we charge edges of the path from v to r in T. It may happen that the part of this path in the charging-ball has very small cost (sum of γ's) or is even empty. In this case, we charge γ_e to the edge on this path leaving the charging-ball. We introduce another parameter $\beta > 1$, and charge the leaving edge in the case, where the path in the charging-ball has cost only a β fraction of the charging-radius. See Figure 4.16 for an illustration of the two cases.



(a) The path from v to w_v in the charging-ball is large enough and gets charged.

(b) The path from v to w_v in the charging-ball is too small, hence edge e_v leaving the charging-ball gets charged.

Figure 4.16: The two charging situations. The absorption-ball of radius $R = r_{\text{absorb}}\gamma_e$ is drawn in purple, the charging-ball is drawn in light purple and has radius $\frac{R}{\alpha}$. Parts of the path $T[v, r]$ are shown in orange. w_v is the last node of $T[v, r]$ in the charging-ball.

Depending on the underlying function f different choices of α and β yield good upper bounds.

Theorem 9. *For a uniform broadcast game $((V,E),r,f,\gamma)$ and a spanning tree T, let σ be the profile returned by Algorithm 1. For any $\alpha > 2$ and $\beta > 1$, we have*

$$\frac{C(\sigma)}{C(T)} \leq \left(\sup_{k\in\mathbb{N}} \frac{kf(k)}{f(1)}\right)(1 + X(\alpha,\beta))$$

where

$$X(\alpha,\beta) = \frac{2\alpha\beta^2(\alpha-1)}{(f(1)-f(2))f(1)(\alpha-2)} \sum_{j=1}^{\infty} \bar{f}(j)^2 + \frac{2f(1)\alpha\beta(\alpha\beta-1)}{(f(1)-f(2))(\beta-1)(\alpha\beta-2)}$$

Proof. As explained above we use (4.14) to bound the social cost of σ and focus on edges in $\mathrm{supp}(\sigma) \setminus T$. Let $\bar{E} = (e_{j_1}, \dots, e_{j_k})$ be the edges in $\mathrm{supp}(\sigma) \setminus T$ indexed by their last appearance in Line 5 of Algorithm 1. That is, e_{j_k} is added to σ_{j_k-1} and $\sigma = \sigma_{j_k}$. This defines the sequence of nodes $\bar{V} = (v_{j_1}, \dots, v_{j_k})$ introducing the edges in $\mathrm{supp}(\sigma) \setminus T$ (see also Line 5 of Algorithm 1).

For each of those nodes $v = v_j \in \bar{V}$ we define edge e_v as the edge on $T[v,r]$ leaving the charging-ball, i.e., $e_v = \{w_v, w'_v\}$ where $d_{T,\bar{f}}(v,w_v) \leq \frac{r_{\mathrm{absorb}}}{\alpha}\gamma_{e_j}$ but $d_{T,\bar{f}}(v,w'_v) > \frac{r_{\mathrm{absorb}}}{\alpha}\gamma_{e_j}$ (see Figure 4.16). To show that e_v exists, we show that the root r is not in the charging-ball of v. Let $v = v_j$ be one of the nodes in $\bar{V}$ and consider the profile $\bar{\sigma} = \sigma_{j-1}$. Since $\bar{\sigma}$ is homogeneous w.r.t. T, we have from Lemma 21 and Lemma 20

$$d_{T,\bar{f}}(v,r) \geq \vec{d}_{T,\bar{f}}(v,r) \geq c_{\bar{\sigma}}\Big(\bar{S}[v \nearrow r]\Big).$$

With the property of e_j from Line 5, we continue with

$$c_{\bar{\sigma}}\Big(\bar{S}[v \nearrow r]\Big) \geq c_{\bar{\sigma}}\Big(\bar{S}[v \nearrow u]\Big) > c^{+1}_{\bar{\sigma}}\Big(\bar{S}[u \nearrow v]\Big) + f(1)\gamma_{e_j} \geq f(1)\gamma_{e_j}.$$

Since $\alpha > 2$, this shows that r is not in the charging-ball of v_j w.r.t. e_j and hence w_{v_j} exists.

Charging-Scheme. We define the charging function $\Gamma : \bar{E} \times T \to \mathbb{R}$, where $\Gamma(e_j, e)$ specifies how much of γ_{e_j} is charged to e. Let $\bar{E}_{\mathrm{in}}$ be those edges $e_j \in \bar{E}$ where $d_{T,\bar{f}}(v_j, w_{v_j}) \geq \frac{r_{\mathrm{absorb}}}{\alpha\beta}\gamma_{e_j}$, and $\bar{E}_{\mathrm{out}}$ be the remaining edges of $\bar{E}$, i.e., those where $d_{T,\bar{f}}(v_j, w_{v_j}) < \frac{r_{\mathrm{absorb}}}{\alpha\beta}\gamma_{e_j}$. For an edge $e_j \in \bar{E}_{\mathrm{in}}$ we charge edges e in $T[v_j, w_{v_j}]$ with

$$\Gamma(e_j, e) = \left(\frac{\alpha\beta}{r_{\mathrm{absorb}}}\right)^2 \left(\sum_{i=1}^{\infty} \bar{f}(i)^2\right) \frac{\gamma_e^2}{\gamma_{e_j}}.$$

For an edge $e_j \in \bar{E}_{\mathrm{out}}$ we charge e_{v_j} with γ_{e_j}. All other pairs (e_j, e) are mapped to 0.

The proof consists of three parts. Firstly, we show that each γ_{e_j} is covered by the charging factors, i.e., for every $e_j \in \bar{E}$:

$$\gamma_{e_j} \leq \sum_{e\in T} \Gamma(e_j, e) \tag{4.15}$$

Secondly, we show that the edges in T are not charged too much, i.e., for every $e \in T$:

$$\sum_{e_j \in \bar{E}} \Gamma(e_j, e) \leq X(\alpha, \beta)\gamma_e. \tag{4.16}$$

Using these two bounds, we finally show the overall bound on the ratio of the social costs.

Proof of (4.15). For an edge $e_j \in \bar{E}_{\text{out}}$, we charge e_{v_j} with γ_{e_j} and hence the inequality is satisfied. For an edge $e_j \in \bar{E}_{\text{in}}$, we have

$$\gamma_{e_j} \leq \left(\frac{\alpha\beta}{r_{\text{absorb}}}\right)^2 \frac{d_{T,\bar{f}}(v_j, w_{v_j})^2}{\gamma_{e_j}}$$

from the square of the defining inequality of $e_j \in \bar{E}_{\text{in}}$. Applying the Cauchy-Schwarz inequality to $d_{T,\bar{f}}(v_j, w_{v_j})^2$, we obtain

$$d_{T,\bar{f}}(v_j, w_{v_j})^2 \leq \left(\sum_{i=1}^{\infty} \bar{f}(i)^2\right)\left(\sum_{e \in T[v_j, w_{v_j}]} \gamma_e^2\right).$$

Plugging this into the previous inequality, we arrive at

$$\gamma_{e_j} \leq \sum_{e \in T} \Gamma(e_j, e).$$

Proof of (4.16). Consider an edge $e \in T$. We bound the amount charged to e for the two cases $\bar{E}_{\text{in}}$ and $\bar{E}_{\text{out}}$ separately. For edges $e_j \in \bar{E}_{\text{in}}$ charging e we upper bound $\frac{\gamma_e}{\gamma_{e_j}}$, and for edges $e_j \in \bar{E}_{\text{out}}$ charging e we upper bound $\frac{\gamma_{e_j}}{\gamma_e}$ to obtain a bound by

$$\sum_{e_j \in \bar{E}} \Gamma(e_j, e) \leq \sum_{\substack{e_j \in \bar{E}_{\text{in}} \\ e \in T[v_j, w_{v_j}]}} \left(\frac{\alpha\beta}{r_{\text{absorb}}}\right)^2 \left(\sum_{i=1}^{\infty} \bar{f}(i)^2\right) \frac{\gamma_e}{\gamma_{e_j}} \gamma_e + \sum_{\substack{e_j \in \bar{E}_{\text{out}} \\ e = e_{v_j}}} \frac{\gamma_{e_j}}{\gamma_e} \gamma_e. \tag{4.17}$$

Consider an edge $e_i \in \bar{E}_{\text{in}}$ charging e. On the one hand we have $d_{T,\bar{f}}(v_i, w_{v_i}) \leq \frac{r_{\text{absorb}}}{\alpha}\gamma_{e_i}$ and on the other hand since $e \in T[v_i, w_{v_i}]$ we also have $f(1)\gamma_e \leq d_{T,\bar{f}}(v_i, w_{v_i})$. Putting both together we get

$$\frac{\gamma_e}{\gamma_{e_i}} \leq \frac{r_{\text{absorb}}}{f(1)\alpha}. \tag{4.18}$$

Now take an edge $e_i \in \bar{E}_{\text{out}}$ charging e, that is, $d_{T,\bar{f}}(v_i, w_{v_i}) < \frac{r_{\text{absorb}}}{\alpha\beta}\gamma_{e_i}$. Let w be the other node of $e = e_{v_i}$ different from w_{v_i}, then we have $d_{T,\bar{f}}(v_i, w) > \frac{r_{\text{absorb}}}{\alpha}\gamma_{e_i}$, since w is not in the charging-ball of v_i. By the triangle inequality for $d_{T,\bar{f}}$ we further have

$$d_{T,\bar{f}}(v_i, w) \leq d_{T,\bar{f}}(v_i, w_{v_i}) + d_{T,\bar{f}}(w_{v_i}, w) = d_{T,\bar{f}}(v_i, w_{v_i}) + f(1)\gamma_e.$$

Combining all bounds results in

$$\frac{\gamma_{e_i}}{\gamma_e} \leq \frac{f(1)\alpha\beta}{r_{\text{absorb}}(\beta-1)}. \tag{4.19}$$

We will now see that the cost factors of edges with overlapping charging-balls have to decrease exponentially, making our previous bounds stronger. Consider two nodes v_i and v_j in $\bar{V}$ with $i < j$ whose charging-balls intersect. I.e., there is a node v in the charging-ball around v_i that is also in the charging-ball around v_j. From the triangle inequality and the monotonicity of $d_{T,\bar{f}}$ (Lemma 20) we get:

$$d_{T,\bar{f}}(v_i, v_j) \leq d_{T,\bar{f}}(v_i, v) + d_{T,\bar{f}}(v, v_j) \leq d_{T,\bar{f}}(v_i, w_{v_i}) + d_{T,\bar{f}}(w_{v_j}, v_j).$$

From the definitions of w_{v_i} and w_{v_j} we obtain

$$d_{T,\bar{f}}(v_i, v_j) \leq \frac{f(1)-f(2)}{2\alpha}\Big(\gamma_{e_i} + \gamma_{e_j}\Big).$$

On the other hand, we have

$$d_{T,\bar{f}}(v_i, v_j) > \frac{f(1)-f(2)}{2}\gamma_{e_j},$$

as v_i was not absorbed by v_j, since e_i is still in the support of σ. Combining both bounds on $d_{T,\bar{f}}(v_i, v_j)$ gives

$$\gamma_{e_j} < \frac{1}{\alpha-1}\gamma_{e_i}.$$

The cost factors decrease even more if both e_i and e_j are in $\bar{E}_{\text{out}}$. We then have the stronger bound

$$d_{T,\bar{f}}(v_i, v_j) \leq \frac{f(1)-f(2)}{2\alpha\beta}\Big(\gamma_{e_i} + \gamma_{e_j}\Big)$$

which gives

$$\gamma_{e_j} < \frac{1}{\alpha\beta-1}\gamma_{e_i}.$$

We can thus bound

$$\sum_{\substack{e_j \in \bar{E}_{\text{in}} \\ e \in T[v_j, w_{v_j}]}} \frac{\gamma_e}{\gamma_{e_j}} \leq \sum_{\ell=0}^{\infty} \frac{1}{(\alpha-1)^\ell}\frac{\gamma_e}{\gamma_{e_k}} \leq \sum_{\ell=0}^{\infty} \frac{1}{(\alpha-1)^\ell}\frac{r_{\text{absorb}}}{f(1)\alpha}$$

with (4.18) and

$$\sum_{\substack{e_j \in \bar{E}_{\text{out}} \\ e = e_{v_j}}} \frac{\gamma_{e_j}}{\gamma_e} \leq \sum_{\ell=0}^{\infty} \frac{1}{(\alpha\beta-1)^\ell}\frac{\gamma_{e_1}}{\gamma_e} \leq \sum_{\ell=0}^{\infty} \frac{1}{(\alpha\beta-1)^\ell}\frac{f(1)\alpha\beta}{r_{\text{absorb}}(\beta-1)}$$

with (4.19). Plugging both into (4.17) and evaluating the geometric series shows (4.16).

Overall bound. With (4.15) and (4.16) we finish the proof by

$$\begin{aligned} C(\sigma) &\leq \left(\sup_{k\in\mathbb{N}} \frac{kf(k)}{f(1)}\right)\left(\sum_{e\in \mathrm{supp}(\sigma)\cap T} f(1)\gamma_e + \sum_{e\in\bar{E}} f(1)\gamma_e\right) \\ &\leq \left(\sup_{k\in\mathbb{N}} \frac{kf(k)}{f(1)}\right)\left(\sum_{e\in T} f(1)\gamma_e + \sum_{e_j\in\bar{E}} f(1)\sum_{e\in T}\Gamma(e_j,e)\right) \\ &\leq \left(\sup_{k\in\mathbb{N}} \frac{kf(k)}{f(1)}\right)\left(\sum_{e\in T} f(1)\gamma_e + \sum_{e\in T} X(\alpha,\beta)f(1)\gamma_e\right) \\ &\leq \left(\sup_{k\in\mathbb{N}} \frac{kf(k)}{f(1)}\right)(1+X(\alpha,\beta))C(T). \end{aligned}$$

□

This gives a constant PoS for any sharing function f with economies of scale where $\sup_{k\in N} \frac{kf(k)}{f(1)}$ and $\sum_{j=1}^{\infty} \bar{f}(j)^2$ are bounded.

For fair cost allocation $\sup_{k\in N} \frac{kf(k)}{f(1)} = 1$ and $\sum_{j=1}^{\infty} \bar{f}(j)^2$ is bounded ([BB95]). We can thus apply Theorem 9 and numerically optimize the values for α and β giving the smallest bound.

Corollary 3. *In broadcast games with fair cost allocation the values* $\alpha \approx 3.22$ *and* $\beta \approx 1.28$ *give the minimal upper bound on the PoS of approximately* 264.25.

4.6 Price of Stability – Lower Bounds for Broadcast Games

In this section, we look at lower bounds for the Price of Stability (PoS) in broadcast games. We give instances of broadcast games with functions in $\mathcal{F}_{\text{lin}}$ and $\mathcal{F}_{\text{poly}}$ and compute explicit bounds on the PoS in these instances. We consider the graph used in [Bil+13] which gives the best known lower bound for fair cost allocation. We give some evidence that the choices made in [Bil+13] are optimal to some extent. From computational experiments we observe that the same instance gives good bounds also in the $\mathcal{F}_{\text{lin}}$ case for small s. For larger s the best bound we found is a small instance with 3 nodes. For functions in $\mathcal{F}_{\text{poly}}$ we additionally look at another graph which gives better bounds for large α.

When searching for lower bound instances we restrict ourselves to graphs being the union of two trees (the tree of a social optimum and the tree corresponding to the best equilibrium). At first, this seems like a natural choice, since equilibria are maintained under edge deletion. That is, every equilibrium is still an equilibrium in a subgraph. However, the opposite is not true: Deleting edges may introduce new equilibria. In particularly interesting for the PoS, the social optimum can become an equilibrium. Consider the graph in Figure 4.17 for an example with fair cost allocation. If all edges are present, the social optimum shown in purple is not an equilibrium as player 3 has an incentive to deviate to $\{r,3\}$: $C_3(\text{purple}) = 4+2+3-\frac{\varepsilon}{3} > 9-\varepsilon$. One can check that the

profile shown in orange is the best equilibrium. Hence the PoS in this instance is $\approx \frac{22}{17}$. However, after removing the dotted edge $\{r, 3\}$ the purple profile is an equilibrium and the PoS drops to 1.

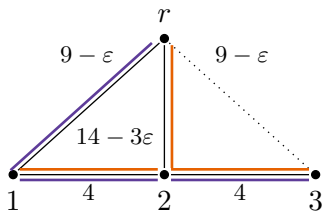


Figure 4.17: Instance of a broadcast game with fair cost allocation. A social optimum is shown in purple, the best equilibrium is shown in orange. The edges $\{1, 2\}$ and $\{2, 3\}$ are used by both the social optimum and the equilibrium. After removing the dotted edge $\{r, 3\}$, the purple profile is an equilibrium.

Thus, it is not clear why looking only at the union of two trees should give the highest lower bound for the PoS. In our experiments we observe, however, that using more edges does not give higher bounds. To determine the PoS precisely, a better understanding of this behavior is needed.

Research Question 5. Does the worst case instance for the PoS in broadcast games need more edges than the ones contained in a social optimum and a best equilibrium?

The structure of this section is as follows. First, we describe the graph we are going to use and how we choose the edge costs to obtain a high PoS. We will then look at the two classes $\mathcal{F}_{\text{lin}}$ and $\mathcal{F}_{\text{poly}}$ and give bounds on the PoS depending on the parameters s and α. For fair cost allocation we show that the costs chosen in [Bil+13] are optimal in a restricted class of instances. The instances we consider for sharing functions with economies of scale are inspired by computational experiments which we briefly explain in the last part of this section.

4.6.1 The Fan Instance

We consider the *fan graph* $F(n)$ as used in [Bil+13] for the currently best lower bounds on the PoS in the fair cost allocation case. The fan graph $F(n)$ consists of a star rooted at r spanning n nodes $1, \dots, n$, and a path connecting nodes 1 to n. The (uniform) *fan instance* is denoted by $\mathcal{F}_f(n, x, y)$, where $n \in \mathbb{N}_{>0}$ is the number of nodes, f is the underlying function, and the vectors $x \in \mathbb{R}^{n-1}_{\geq 0}$ and $y \in \mathbb{R}^{n}_{\geq 0}$ give the scaling factors for the edge costs. We index x beginning with 2, i.e., we think of $x \in \mathbb{R}^{n}_{\geq 0}$ where x_1 is not used. See Figure 4.18 for an illustration of the fan instance.

Some spanning trees in a fan graph are of special interest. We refer to the star rooted at r and spanning all nodes $1, \dots, n$ as *the star* S_n. A spanning tree using all edges $1 - 2 - \cdots - n$ is characterized by the (unique) edge incident to the root. Such a tree is denoted by T_t, where the root is connected to node t. A special case of those trees is T_1 consisting of the edges $r - 1 - 2 - \cdots - n$ which is a path and hence will be called *the path*. The set of all trees using all edges $1 - 2 - \cdots - n$ is denoted by $\mathcal{T}_n$.

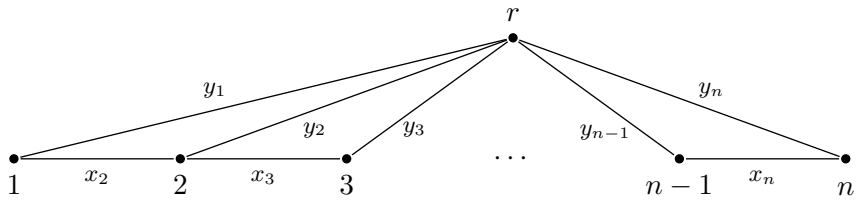


Figure 4.18: The fan instance $\mathcal{F}_f(n, x, y)$ consist of an n-star rooted at r and a path connecting nodes 1 to n. The scaling factors for the cost functions are given as edge labels.

We consider a subclass $\mathcal{F}^*$ of fan instances where the edge costs are such that the star is one of the best equilibria. For a fixed underlying function f the PoS in this class can be expressed as

$$\mathrm{PoS}^f_{\mathcal{F}^*}(n) = \max_{n' \le n} \quad \max_{\text{tree } T \text{ in } F(n')} \quad \max_{\substack{x,y \in \mathbb{R}^{n'}_{\ge 0} \\ S_{n'} \text{ is best equilibrium}}} \frac{C(S_{n'})}{C(T)}.$$

In contrast to the definition of $\mathrm{PoS}(n)$ in Section 2.2 where we fix OPT and minimize over all equilibria, here we fix the best equilibrium to be the star and maximize over all trees for the social optimum. We briefly discuss the first maximum in the above expression. When looking at subclasses of graphs it is not clear why the value of the PoS (without the first maximum) should be increasing. In particular, if the class is not closed under the operation of adding a node, it could be the case that the value is higher for smaller number of nodes. That is why we have to include the maximum over smaller instances.

The key ingredient to compute $\mathrm{PoS}^f_{\mathcal{F}^*}(n)$ is to formulate the innermost maximization problem as linear program. With this we get a lower bound on the PoS by computing a feasible solution to this LP.

Characterization of a subset of $\mathcal{F}^*$ We start by looking closer at instances in $\mathcal{F}^*$. If the underlying function f satisfies two monotonicity properties, we have a characterization for instances $\mathcal{F}_f(n, x, y)$ being in $\mathcal{F}^*$.

Lemma 24. *Let $y > 0$ and f be a sharing function with economies of scale satisfying*

$$\forall k \ge 1 : f(1)\left(\frac{f(2)}{f(1)}\right)^k \le f(2k) \qquad \text{and} \tag{4.20}$$

$$\forall k \ge 1 : f(3)\left(\frac{f(2)}{f(1)}\right)^{k-1} \le f(2k+1). \tag{4.21}$$

Then, $\mathcal{F}_f(n, x, y)$ lies in $\mathcal{F}^*$, if and only if for x and y all of the following hold:

$$\forall i \in \{1, \dots, n-1\} : f(1)y_i < f(1)x_{i+1} + f(2)y_{i+1} \tag{4.22}$$

$$\forall i \in \{2, \dots, n\} : f(1)y_i < f(1)x_i + f(2)y_{i-1} \tag{4.23}$$

$$\forall i \in \{2,\dots,n-1\}: f(1)y_{i-1} < f(1)x_i + f(3)y_i \quad \text{or} \tag{4.24}$$
$$f(1)y_{i+1} < f(1)x_{i+1} + f(3)y_i \quad \text{or} \tag{4.25}$$
$$f(1)y_{i-1} + f(1)y_i + f(1)y_{i+1} \leq f(1)x_i + f(1)x_{i+1} + 3f(3)y_i \tag{4.26}$$

Proof. Let f be a sharing function with economies of scale satisfying (4.20) and (4.21). "$\Leftarrow$": Assume f additionally satisfies (4.22) to (4.26). We show that the star S_n is one of the best equilibria. Let σ be the profile corresponding to S_n, that is, for every player $i \in \{1,\dots,n\}$ we have $\sigma_i = \{\{i,r\}\}$. First, we show that σ is an equilibrium. The current cost of player i is $C_i(\sigma) = f(1)y_i$. Any other strategy of i is of the form: go to some $j \neq i$ and then to r. There are two possibilities: go to the right or to the left. We show that none of these strategies is improving for i in σ.

If $j > i$, consider the strategy $s' = \{\{k,k+1\} : k = i,\dots,i-j-1\} \cup \{j,r\}$. Using (4.22) for $j-1$ and that f is decreasing we can bound the cost of s' as

$$\begin{aligned} C_i(s',\sigma_{-i}) &= f(1)x_i + \dots + f(1)x_{j-1} + f(1)x_j + f(2)y_j \\ &> f(1)x_i + \dots + f(1)x_{j-1} + f(1)y_{j-1} && (4.22) \\ &> f(1)x_i + \dots + f(1)x_{j-1} + f(2)y_{j-1}. && (f \text{ decr.}) \end{aligned}$$

We can continue this chain of inequalities using (4.22) for indices $k = j-2,\dots,i+1$ to obtain

$$C_i(s',\sigma_{-i}) > f(1)x_{i+1} + f(2)y_{i+1} > f(1)y_i = C_i(\sigma)$$

showing that s' is not improving for i. In the other case, where $j < i$, we consider $s' = \{\{k,k-1\} : k = i,\dots,i-j-1\} \cup \{j,r\}$. Taking now inequalities (4.23) for indices $k = j+1,\dots,i$ in the arguments from above we get that $C_i(s',\sigma_{-i}) > C_i(\sigma)$. Thus we established that σ is an equilibrium. We now have to argue that σ is one of the best equilibria in $\mathcal{F}_f(n,x,y)$. Actually we show the stronger statement that σ is the *unique* equilibrium.

Let $\bar{\sigma} \neq \sigma$ be any profile with tree support. We show that $\bar{\sigma}$ can not be an equilibrium, and hence by Lemma 10 there are no other equilibria in $\mathcal{F}_f(n,x,y)$. As $\bar{\sigma} \neq \sigma$ some edge $\{i,r\}$ is used by at least 2 players. For this edge let i_ℓ and i_r be the player with smallest and largest index using $\{i,r\}$. Since $\mathrm{supp}(\bar{\sigma})$ is a tree, all players between i_ℓ and i_r are using edge $\{i,r\}$ and we have $n_{\bar{\sigma}}(\{i,r\}) = i_r - i_\ell + 1 \geq 2$. There are three cases: either i_ℓ is farther away from i than i_r, or the other way around, or both have the same distance to i.

We start with the case $i - i_\ell > i_r - i$, where $i_r - i_\ell + 1 \leq 2(i - i_\ell)$ and in particular $1 \leq i - i_\ell$. We consider player i_ℓ and show that there is an improving move. The current cost of player i_ℓ is

$$C_{i_\ell}(\bar{\sigma}) = \sum_{k=1}^{i-i_\ell} f(k)x_{i_\ell+k} + f(i_r - i_\ell + 1)y_i. \tag{4.27}$$

Chaining inequalities (4.22) for indices $k = i_\ell,\dots,i-1$ yields

$$f(1)y_{i_\ell} < \sum_{k=1}^{i-i_\ell} \left(\frac{f(2)}{f(1)}\right)^{k-1} f(1)x_{i_\ell+k} + \left(\frac{f(2)}{f(1)}\right)^{i-i_\ell} f(1)y_i. \tag{4.28}$$

We compare the coefficients in (4.27) and (4.28). From (4.20) we have

$$\left(\frac{f(2)}{f(1)}\right)^{k-1} f(1) \leq f(k) \quad \text{and} \tag{4.29}$$
$$\left(\frac{f(2)}{f(1)}\right)^{i-i_\ell} f(1) \leq f(2(i-i_\ell)) \leq f(i_r - r_\ell + 1)$$

and hence $C_{i_\ell}(\bar{\sigma}) > f(1)y_{i_\ell}$. So $\{i_\ell, r\}$ is an improving move for i_ℓ in $\bar{\sigma}$. The case $i_r - i > i - i_\ell$ is symmetric to the previous case. Consider player i_r and use inequalities (4.23) to observe that $\{i_r, r\}$ is an improving move.

We are left with the case $i - i_\ell = i_r - i$, where $i_r - i_\ell + 1 = 2(i - i_\ell)$. Since $i_r - r_\ell + 1 \geq 2$ we have $i - i_\ell, i_r - i \geq 1$ and hence $i \in \{2, \ldots, n-1\}$. Assume now that (4.24) holds at i. As before we have for the current cost of player i_ℓ

$$C_{i_\ell}(\bar{\sigma}) = \sum_{k=1}^{i-i_\ell} f(k)x_{i_\ell+k} + f(i_r - i_\ell + 1).$$

Also as previously done, we chain the inequalities (4.22) for indices $k = i_\ell, \ldots, i-2$ but now for index $i-1$ we use the stronger inequality (4.24) for i to get

$$f(1)y_{i_\ell} < \sum_{k=1}^{i-i_\ell} \left(\frac{f(2)}{f(1)}\right)^{k-1} f(1)x_{i_\ell+k} + \left(\frac{f(2)}{f(1)}\right)^{i-i_\ell-1} f(3)y_i.$$

To compare the coefficients we still have (4.29) and using (4.21) we get

$$\left(\frac{f(2)}{f(1)}\right)^{i-i_\ell-1} f(3) \leq f(2(i-i_\ell)+1) = f(i_r - i_\ell + 1).$$

Again we obtain $C_{i_\ell}(\bar{\sigma}) > f(1)y_{i_\ell}$ showing that i_ℓ has an improving move switching to $\{i_\ell, r\}$. If (4.24) does not hold at i but (4.25) does, again consider i_r and use (4.23) together with (4.25) in the above reasoning to see that i_r has an incentive to switch to $\{i_r, r\}$.

To conclude the first direction of this proof we show that for x, y with $y > 0$ satisfying (4.24), (4.25) and (4.26) at least one of (4.24) or (4.25) has to hold. Assume (4.24) and (4.25) are violated at some $i \in \{2, \ldots, n-1\}$, that is,

$$f(1)y_{i-1} \geq f(1)x_i + f(3)y_i \qquad \text{and} \qquad f(1)y_{i+1} \geq f(1)x_{i+1} + f(3)y_i$$

hold. Then we have

$$f(1)y_{i-1} + f(1)y_i + f(1)y_{i+1} \geq f(1)x_i + f(1)x_{i+1} + (2f(3) + f(1))y_i.$$

Since $y_i > 0$ and $f(3) < f(1)$ this is a contradiction to (4.26).

We finished the first part of the proof and showed: If x, y with $y > 0$ satisfy (4.22) to (4.26), than the star σ is the unique equilibrium in $\mathcal{F}_f(n, x, y)$.

"⇒": For the other direction, take again f satisfying (4.20) and (4.21) and assume now that the star σ is one of the best equilibria in $\mathcal{F}_f(n,x,y)$. We show that x,y satisfy all constraints (4.22) to (4.26).

Since σ is an equilibrium no player has an improving move. For every player i going to one of her neighbors $i-1$ or $i+1$ and then to r is a feasible strategy. From this we already know that (4.22) and (4.23) have to hold with $\leq$ instead of the strict inequality. Now assume that for some $i \in \{1,\dots,n-1\}$ (4.22) is not satisfied with strict inequality, that is, $f(1)y_i = f(1)x_{i+1} + f(2)y_{i+1}$ holds. Consider the strategy $s' = \{\{i,i+1\},\{i+1,r\}\}$ for i and define $\sigma^i = (s',\sigma_{-i})$. Comparing the social cost of σ^i and σ as

$$\begin{aligned} C\left(\sigma^i\right) - C(\sigma) &= f(1)x_{i+1} - f(1)y_i + (2f(2) - f(1))y_{i+1} \\ &< f(1)x_{i+1} - f(1)y_i + f(2)y_{i+1} = 0 \end{aligned}$$

shows that σ^i can not be an equilibrium in $\mathcal{F}_f(n,x,y)$ since σ is an equilibrium of minimal social cost. Starting improving-dynamics from σ^i ends in an equilibrium σ' with strictly smaller potential $\Phi(\sigma') < \Phi(\sigma^i)$ (since there is a player in σ^i with an improving move). We compute the potential of σ^i as

$$\Phi\left(\sigma^i\right) = \sum_{\substack{k=1 \\ k\neq i}}^{n} f(1)y_k + f(1)x_{i+1} + f(2)y_{i+1} = \sum_{k=1}^{n} f(1)y_k.$$

For the star σ we have $\sum_{k=1}^{n} f(1)y_k = C(\sigma)$. Combining all inequalities we obtain

$$C(\sigma') \leq \Phi(\sigma') < \Phi\left(\sigma^i\right) = C(\sigma).$$

This is a contradiction, as σ is an equilibrium of minimal social cost. Thus, (4.22) can not be satisfied with equality. For (4.23) take i and consider the symmetric strategy $\{\{i,i-1\},\{i-1,r\}\}$. The same arguments as above show that (4.23) has to hold with strict inequality.

Now assume (4.24), (4.25) and (4.26) are not satisfied at some $j \in \{2,\dots,n-2\}$. In particular, we have $f(1)y_{j-1} \geq f(1)x_j + f(3)y_j$ and $f(1)y_{j+1} \geq f(1)x_{j+1} + f(3)y_j$. Consider the profile $\bar{\sigma}^j$ which differs from the star σ only in the strategies of player $j-1$ and $j+1$ who go via j. Formally $\bar{\sigma}^j_{j-1} = \{\{j-1,j\},\{j,r\}\}$ and $\bar{\sigma}^j_{j+1} = \{\{j+1,j\},\{j,r\}\}$. We show that $\bar{\sigma}^j$ is an equilibrium with smaller social cost than σ which gives a contradiction. To see that $\bar{\sigma}^j$ is an equilibrium we check (NECond) from Lemma 11. For an edge $e = \{k,k+1\} \notin \operatorname{supp}(\bar{\sigma}^j)$ where $k \in \{1,\dots,n\}\setminus\{j-3,j-2,j-1,j,j+1,j+2,j+3\}$ we have from (4.22) and (4.23)

$$c_{\bar{\sigma}^j}(T[k \nearrow k+1]) = f(1)y_k < f(1)x_{k+1} + f(2)y_{k+1} = c_{\{k,k+1\}}(1) + c^{+1}_{\bar{\sigma}^j}(T[k+1 \nearrow k])$$

and

$$c_{\bar{\sigma}^j}(T[k+1 \nearrow k]) = f(1)y_{k+1} < f(1)x_{k+1} + f(2)y_k = c_{\{k,k+1\}}(1) + c^{+1}_{\bar{\sigma}^j}(T[k \nearrow k+1]).$$

For $e = \{j-2, j-1\}$ we have

$$\begin{aligned} c_{\bar{\sigma}^j}(T[j-2 \nearrow j-1]) &= f(1)y_{j-2} \\ &\leq f(1)x_{j-1} + f(2)x_j + \left(\frac{f(2)}{f(1)}\right)^2 f(1)y_j && ((4.22)) \\ &\leq f(1)x_{j-1} + f(2)x_j + f(4)y_j && ((4.20)) \\ &= c_{\{j-2,j-1\}}(1) + c^{+1}_{\bar{\sigma}^j}(T[j-1 \nearrow j-2]) \end{aligned}$$

and

$$\begin{aligned} c_{\bar{\sigma}^j}(T[j-1 \nearrow j-2]) &= f(1)x_j + f(3)y_j \\ &\leq f(1)y_{j-1} \\ &< f(1)x_{j-1} + f(2)y_{j-2} && ((4.23)) \\ &\leq f(1)x_{j-1} + f(1)y_{j-2} \\ &= c_{\{j-2,j-1\}}(1) + c^{+1}_{\bar{\sigma}^j}(T[j-2 \nearrow j-1]) \end{aligned}$$

For $e = \{j+1, j+2\}$ the same arguments hold exchanging (4.22) and (4.23). The remaining cases are $\{j-1, r\}$ and $\{j+1, r\}$ where we have the desired inequalities from our assumption that (4.24) and (4.25) do not hold. In total, we obtain that $\bar{\sigma}^j$ is an equilibrium. Comparing the social cost to σ gives

$$\begin{aligned} C\left(\bar{\sigma}^j\right) - C(\sigma) &= f(1)x_j + f(1)x_{j+1} - f(1)y_{j-1} + f(1)y_{j+1} + (3f(3) - f(1))y_j \\ &< f(1)x_j + f(1)x_{j+1} - f(1)y_{j-1} + f(1)y_{j+1} + 2f(3)y_j \leq 0. \end{aligned}$$

This is a contradiction to σ having smallest social cost among all equilibria.

We have shown that if the star σ is one of the best equilibria in $\mathcal{F}_f(n, x, y)$, then all constraints (4.22) to (4.26) have to be satisfied which concludes the proof □

Observe from the proof of Lemma 24 that the star is the *unique* equilibrium, if it is one of the best. From the same proof we also get that dropping (4.26) gives a characterization for the star being the unique equilibrium.

Corollary 4. *In a fan instance $\mathcal{F}_f(n, x, y)$ where $y > 0$ and f satisfies (4.20) and (4.21), the star is the unique equilibrium, if and only if all of the conditions (4.22), (4.23), (4.24), and (4.25) from Lemma 24 hold.*

Maximizing over smaller instances allows to further restrict our attention to instances where x and y are strictly positive and consider only trees in $\mathcal{T}$.

Lemma 25. *For a tree T in $F(n)$ not containing edge $\{i, i+1\}$ we have the bound*

$$\max_{\substack{x,y \in \mathbb{R}^n_{\geq 0} \\ S_n \text{ is best equilibrium}}} \frac{C(S_n)}{C(T)} \leq \max\left\{\mathrm{PoS}^f_{\mathcal{F}^*}(i), \mathrm{PoS}^f_{\mathcal{F}^*}(n-i)\right\}$$

Proof. We split the instance into two smaller instances left and right of edge $\{i, i+1\}$ by removing $\{i, i+1\}$. The left part is then induced by nodes $\{1, \dots, i, r\}$ and the right part by nodes $\{i+1, \dots, n, r\}$. Note that these are again fan instances. Now consider $x, y \in \mathbb{R}^n_{\geq 0}$ giving the maximum on the left hand side of the statement of the lemma. Define x^L, y^L and x^R, y^R to be the entries of x and y in the left and right part respectively. Again, we think of $x^R \in \mathbb{R}^{n-i}_{\geq 0}$ by adding some first component which is not used.

Let S^L be the part of the star S_n in the left part with the edges $\{\{j, r\} : j = 1, \dots, i\}$, and S^R the part of S_n in the right part. Then we have $C(S) = C\Big(S^L\Big) + C\Big(S^R\Big)$. Similarly define T^L and T^R to be the parts of tree T in the left and right instance. As $\{i, i+1\} \notin T$ we also have $C(T) = C\Big(T^L\Big) + C\Big(T^R\Big)$ and in total

$$\frac{C(S_n)}{C(T)} = \frac{C\Big(S^L\Big) + C\Big(S^R\Big)}{C(T^L) + C(T^R)} \leq \max\left\{ \frac{C\Big(S^L\Big)}{C(T^L)}, \frac{C\Big(S^R\Big)}{C(T^R)} \right\}.$$

Observe that S^L is one of the best equilibria in the left instance: First, since the star S_n is an equilibrium in the original instance, S^L is an equilibrium in the small instance. Further, if there was another equilibrium σ' in the left instance with smaller social cost, we could replace S^L by σ' in the original instance and obtain an equilibrium which is better than the star. Similar arguments hold for S^R in the right instance. Hence, we split the original instance in two smaller instances again in $\mathcal{F}^*$ and thus

$$\frac{C(S_n)}{C(T)} \leq \max\Big\{ \mathrm{PoS}^f_{\mathcal{F}^*}(i), \mathrm{PoS}^f_{\mathcal{F}^*}(n-i) \Big\}.$$

□

We use this lemma to show that we can assume x and y to be strictly positive.

Lemma 26. *In a fan instance $\mathcal{F}_f(n, x, y)$ in $\mathcal{F}^*$ where some $y_i = 0$ or some $x_i = 0$ we have the bound*

$$\frac{C(S_n)}{C(T)} \leq \max\Big\{ \mathrm{PoS}^f_{\mathcal{F}^*}(i), \mathrm{PoS}^f_{\mathcal{F}^*}(n-i) \Big\}$$

for every tree T in $F(n)$.

Proof. Let T be some tree in $F(n)$. If $T \notin \mathcal{T}_n$ we get the bound immediately from Lemma 25. Now assume $T \in \mathcal{T}_n$, that is, $T = T_t$ for some $t \in \{1, \dots, n\}$. First consider the case where $y_i = 0$ for some $i \in \{1, \dots, n\}$. We build a new tree T' from T of smaller social cost. The idea is to have $\{i, i+1\} \notin T'$ and hence $T' \notin \mathcal{T}_n$ so that we can use Lemma 25.

If $t \leq i$, we replace $\{i, i+1\}$ by $\{i+1, r\}$. This results in a tree T'. Since the star is an equilibrium and $y_i = 0$ we have $x_{i+1} \geq y_{i+1}$ and hence we replaced the edge by an edge of no greater cost. By this replacement the number of players using an edge $e \neq \{i+1, r\}$

in T' has only decreased. For $\{i+1,r\} \notin T$ we have $n_{T'}(\{i+1,r\}) \leq n_T(\{i,i+1\})$. Since the total cost of an edge $kf(k)$ is non-decreasing in the number of players, we get

$$\begin{aligned} C(T') &= \sum_{\substack{e \in T' \\ e \neq \{i+1,r\}}} n_{T'}(e) f(n_{T'}(e))\gamma_e + n_{T'}(\{i+1,r\}) f(n_{T'}(\{i+1,r\}))y_{i+1} \\ &\leq \sum_{\substack{e \in T' \\ e \neq \{i+1,r\}}} n_T(e) f(n_T(e))\gamma_e + n_T(\{i,i+1\}) f(n_T(\{i,i+1\}))x_{i+1} \\ &= C(T). \end{aligned}$$

Otherwise, if $t \geq i+1$, we replace $\{i,i+1\}$ by $\{i,r\}$. As before this decreases the number of players on edges other than $\{i,r\}$. With $y_i = 0$ we thus have $C(T') \leq C(T)$.

In all cases we get

$$\frac{C(S_n)}{C(T)} \leq \frac{C(S_n)}{C(T')}$$

and the desired bound from Lemma 25 as $\{i,i+1\} \notin T'$.

If some $x_i = 0$, then we also have $y_i = 0$: since the star is an equilibrium we have $f(1)y_{i-1} \leq f(2)y_i$ and $f(1)y_i \leq f(2)y_{i-1}$. Since f is decreasing this can only be true if y_i and y_{i-1} are 0. Hence, we get the bound from the above discussion. □

Combining Lemma 24, Lemma 25 and Lemma 26 we get

$$\mathrm{PoS}^f_{\mathcal{F}^*}(n) = \max_{n' \leq n} \max_{T \in \mathcal{T}_{n'}} \max_{\substack{x,y \in \mathbb{R}^{n'}_{>0} \\ S_{n'} \text{ unique equilibrium}}} \frac{C(S_{n'})}{C(T)}$$

for a sharing function f with economies of scale satisfying (4.20) and (4.21). Thus, for computing $\mathrm{PoS}^f_{\mathcal{F}^*}(n)$ (or lower bounds) we can restrict our attention to fan instances where $x, y > 0$ and the star is the unique equilibrium. From now on we consider underlying functions f satisfying (4.20) and (4.21) where we have the characterizations of Lemma 24 and Corollary 4 for the star being the unique equilibrium.

The LP for $\mathrm{PoS}^f_{\mathcal{F}^*}(n)$. Let f be a sharing function with economies of scale satisfying (4.20) and (4.21). Using Lemma 24 and Corollary 4 we write $\max_{S_n \text{ unique equilibrium}} \frac{C(S_n)}{C(T)}$ for a fixed tree T in $F(n)$ as the following optimization problem.

$$\begin{aligned} \sup \quad & \frac{C(S_n)}{C(T)} && && (P_n^T) \\ \text{s.t.} \forall i \in \{1,\dots,n-1\}: \quad & f(1)y_i < f(1)x_{i+1} + f(2)y_{i+1} \\ \forall i \in \{2,\dots,n\}: \quad & && f(1)y_i \;\; < f(1)x_i + f(2)y_{i-1} \\ \forall i \in \{2,\dots,n-1\}: \quad & f(1)y_{i-1} < f(1)x_i + f(3)y_i && \text{or } f(1)y_{i+1} < f(1)x_{i+1} + f(3)y_i \\ & x, y \geq 0 && && (4.30) \end{aligned}$$

We approximate the optimal value of (P_n^T) by a maximization over several linear programs, where we fix the choices of the satisfied disjunct in the or-constraint in (4.30). For this, we introduce binary variables $z \in \{0,1\}^n$. Setting $z_i = 0$ corresponds to choosing $f(1)y_{i-1} < f(1)x_i + f(3)y_i$ and setting $z_i = 1$ to choosing $f(1)y_{i+1} < f(1)x_{i+1} + f(3)y_i$ for $i \in \{2,\dots,n-1\}$. We fix $z_1 = 0$ and $z_n = 1$. Observe that for a fixed $i \in \{2,\dots,n-1\}$ we either have the inequalities $f(1)y_i < f(1)x_{i+1} + f(2)y_{i+1}$ and $f(1)y_i < f(1)x_{i+1} + f(3)y_{i+1}$; or $f(1)y_i < f(1)x_i + f(2)y_{i-1}$ and $f(1)y_i < f(1)x_i + f(3)y_{i-1}$. In both cases the second inequality is stronger than the first, as f is decreasing. We can thus combine the corresponding inequalities and obtain an LP with $2n-1$ variables and $2(n-1)+1+(2n-1)$ inequalities for a fixed choice of z.

$$\max \quad C(S_n) \tag{LP_z^T}$$

$$\text{s.t.} \quad \forall i \in \{2,\dots,n\} : f(1)y_{i-1} \le f(1)x_i + (z_i f(2) + (1-z_i)f(3))y_i \tag{4.31}$$

$$\forall i \in \{1,\dots,n-1\} : f(1)y_{i+1} \le f(1)x_{i+1} + (z_i f(3) + (1-z_i)f(2))y_i \tag{4.32}$$

$$C(T) = 1 \tag{4.33}$$

$$x, y \ge 0. \tag{4.34}$$

If we ignore the normalization constraint (4.33), any feasible solution can be scaled by a positive factor staying feasible. Any solution to this new LP can be transformed to a feasible solution of (LP_z^T) by scaling with $\frac{1}{C(T)}$. We will thus ignore (4.33) and consider the new objective function value $\frac{C(S_n)}{C(T)}$ for any solution. We then have

Remark 1. The optimal value of (P_n^T) and the maximum $\max_{\substack{z\in\{0,1\}^n \\ z_1=0, z_n=1}} (\mathrm{LP}_z^T)$ are arbitrarily close.

Proof. First, consider an optimal solution to (P_n^T). This solution is feasible for (LP_z^T) for some choice of $z \in \{0,1\}^n$. Hence, the optimal value of (P_n^T) is upper bounded by $\max_{z\in\{0,1\}^n} (\mathrm{LP}_z^T)$. On the other hand, consider $z \in \{0,1\}^n$ maximizing $\max_{z\in\{0,1\}^n} (\mathrm{LP}_z^T)$ and an optimal solution (x,y) to (LP_z^T). We add a small $\varepsilon > 0$ to x and obtain $(\hat{x},y)$. Since x's only appear on the right hand sides, $(\hat{x},y)$ is a feasible solution for (P_n^T). We bound the social cost of T w.r.t. $(\hat{x},y)$ by

$$C_{(\hat{x},y)}(T) \le C_{(x,y)}(T) + n\varepsilon.$$

If we choose $\varepsilon > 0$ small enough then $\frac{C_{(\hat{x},y)}(S_n)}{C_{(\hat{x},y)}(T) - n\varepsilon}$ is close to the objective function value of (P_n^T) for $(\hat{x},y)$. And hence, $\max_{z\in\{0,1\}^n} (\mathrm{LP}_z^T) = \frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T)}$ is a lower bound on (P_n^T). □

With these LPs we can write the PoS informally as

$$\mathrm{PoS}_{\mathcal{F}^*}^f(n) = \max_{n' \le n} \max_{T \in \mathcal{T}_{n'}} \max_{\substack{z\in\{0,1\}^{n'} \\ z_1=0, z_{n'}=1}} (\mathrm{LP}_z^T).$$

At the basic feasible solutions of (LP_z^T) $2n-1$ constraints are active: Either all of (4.31) and (4.32) (additionally to the normalization constraint), or some x_i or y_i is 0 . The objective function value of a solution where some x_i or y_i is 0 is bounded by the PoS of a smaller instance by Lemma 26. Thus, it suffices to look at the basic solution where all of (4.31) and (4.32) are active. We denote this solution by (x_z^*, y_z^*) and obtain

$$\mathrm{PoS}_{\mathcal{F}^*}^f(n) = \max_{n' \leq n} \max_{T \in \mathcal{T}_{n'}} \max_{\substack{z \in \{0,1\}^{n'} \\ z_1=0, z_{n'}=1}} \frac{C_{(x_z^*, y_z^*)}(S_{n'})}{C_{(x_z^*, y_z^*)}(T)}. \tag{4.35}$$

The basic solution (x_z^*, y_z^*) The system of linear equations given by constraints (4.31) and (4.32) of (LP_z^T) has a unique solution (x_z^*, y_z^*) (or (x^*, y^*) when z is clear from the context) given by the recursion

$$y_i = \frac{f(1) + z_{i-1}f(3) + (1 - z_{i-1})f(2)}{f(1) + z_i f(2) + (1 - z_i)f(3)} y_{i-1} \qquad \text{and} \tag{4.36}$$

$$x_i = \frac{f(1)^2 - (z_{i-1}f(3) + (1 - z_{i-1})f(2))(z_i f(2) + (1 - z_i)f(3))}{f(1)(f(1) + z_{i-1}f(3) + (1 - z_{i-1})f(2))} y_i \tag{4.37}$$

for $i \in \{2, \ldots, n\}$. Note that this actually describes a family of feasible solutions as we are free to choose one of the y values. However, all these solutions give the same objective value for trees $T_t \in \mathcal{T}$ as both

$$C_{(x_z^*, y_z^*)}(S_n) = \sum_{i=1}^{n} f(1) y_i^* \quad \text{and} \tag{4.38}$$

$$C_{(x_z^*, y_z^*)}(T_t) = \sum_{i=2}^{t} (i-1) f(i-1) x_i^* + n f(n) y_t^* + \sum_{i=t+1}^{n} (n+1-i) f(n+1-i) x_i^* \tag{4.39}$$

contain the same number of terms and implicit occurrences of the chosen common y parameter and hence get scaled by the same factor when changing the parameter.

We often interpret the objective function value for a tree $T_t \in \mathcal{T}$ as weighted harmonic mean. We use y_t as free parameter in the basic solution (x^*, y^*) and express all other y's and x's in terms of y_t. The objective value can be written as

$$\frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_t)} = \frac{\sum_{i=1}^{t-1} f(1) y_i + f(1) y_t + \sum_{i=t+1}^{n} f(1) y_i}{\sum_{i=1}^{t-1} i f(i) \frac{x_{i+1}}{y_i} y_i + n f(n) y_t + \sum_{i=t+1}^{n} (n+1-i) f(n+1-i) \frac{x_i}{y_i} y_i}. \tag{4.40}$$

This is the weighted harmonic mean of the elements $\frac{f(1)}{i f(i)} \frac{y_i}{x_{i+1}}$ with weight $f(1)y_i$ for $i \in \{1, \ldots, t-1\}$, the element $\frac{f(1)}{n f(n)}$ of weight $f(1)y_t$, and the elements $\frac{f(1)}{(n+1-i)f(n+1-i)} \frac{y_i}{x_i}$ with weight $f(1)y_i$ for $i \in \{t+1, \ldots, n\}$. For fair cost allocation this simplifies as all scalar factors are 1. In particular, for the path we have

$$\frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_1)} = \mathcal{H}\left((y_1, 1), \left(y_2, \frac{y_2}{x_2}\right), \ldots \left(y_n, \frac{y_n}{x_n}\right)\right). \tag{4.41}$$

We focus on a particular choice of z and define

$$z_n^*(\zeta) = (\zeta, 0, \dots, 0, 1) \in \{0,1\}^n \text{ for } \zeta \in \{0,1\}. \tag{4.42}$$

For $z_n^*(0)$ the recursion in (4.36) and (4.37) evaluates to

$$\begin{aligned} y_i &= \frac{f(1)+f(2)}{f(1)+f(3)} y_{i-1} \quad \text{and} \quad x_i = \frac{f(1)^2 - f(2)f(3)}{f(1)(f(1)+f(2))} y_i \quad \text{for } i = 2, \dots, n-1 \\ y_n &= y_{n-1} \quad \text{and} \quad x_n = \frac{f(1)-f(2)}{f(1)} y_n. \end{aligned} \tag{4.43}$$

We observe a symmetry in the basic solutions w.r.t. z.

Lemma 27. *For $t \in \mathbb{N}$ and $z \in \{0,1\}^t$ define $\overleftarrow{z}$ by $\overleftarrow{z}_i = 1 - z_{t-i+1}$. Let $(x,y) = (x_z^*, y_z^*)$ and $(x',y') = (x_{\overleftarrow{z}}^*, y_{\overleftarrow{z}}^*)$. Then*

$$\frac{\sum_{i=1}^{t-1} y'_i}{\sum_{i=1}^{t-1} x'_{i+1}} = \frac{\sum_{i=2}^{t} y_i}{\sum_{i=2}^{t} x_i}.$$

Proof. We use the representation of (4.40)

$$\frac{\sum_{i=1}^{t-1} y'_i}{\sum_{i=1}^{t-1} x'_{i+1}} = \frac{\sum_{i=1}^{t-1} \frac{y'_i}{y'_t}}{\sum_{i=1}^{t-1} \frac{y'_i}{y'_t} \cdot \frac{x'_{i+1}}{y'_i}} \quad \text{and} \quad \frac{\sum_{i=2}^{t} y_i}{\sum_{i=2}^{t} x_i} = \frac{\sum_{i=2}^{t} \frac{y_i}{y_1}}{\sum_{i=2}^{t} \frac{y_i}{y_t} \cdot \frac{x_i}{y_i}}$$

and the transformation $i \leftrightarrow t-i+1$. We show

$$\frac{y_i}{y_1} = \frac{y'_{t-i+1}}{y'_t} \quad \text{and} \quad \frac{x_i}{y_i} = \frac{x'_{t-i+2}}{y'_{t-i+1}}. \tag{4.44}$$

For the first equality consider $\frac{y_i}{y_{i-1}}$ and (4.36). We have $z_{i-1} = 1 - \overleftarrow{z}_{t-i+2}$ and $z_i = 1 - \overleftarrow{z}_{t-i+1}$ from the definition of $\overleftarrow{z}$. Hence

$$\frac{y_i}{y_{i-1}} = \frac{y'_{t-i+1}}{y'_{t-1+2}}$$

which shows the first equality in (4.44) by induction. For the second equality we write

$$\frac{x'_{t-i+2}}{y'_{t-i+1}} = \frac{x'_{t-i+2}}{y'_{t-i+2}} \cdot \frac{y'_{t-i+2}}{y'_{t-i+1}}.$$

With (4.36), (4.37), $z_{i-1} = 1 - \overleftarrow{z}_{t-i+2}$ and $z_i = 1 - \overleftarrow{z}_{t-i+1}$, we obtain the second equality in (4.44). □

This lemma is especially helpful for the following choice of z. Define

$$z_n^t = (0, 1, \dots, 1, 0, \dots, 0, 1) \in \{0,1\}^n, \tag{4.45}$$

where $z_1 = 0$ and $z_t, \dots, z_{n-1} = 0$ and all other entries are 1. For example $z_5^1 = (0,0,0,0,1)$, $z_5^2 = (0,0,0,0,1)$ and $z_5^3 = (0,1,0,0,1)$.

Notice that the last part of z_n^t after t is $z^*_{n-t+1}(0)$ and $\overleftarrow{(z_1, \dots, z_{t-1})} = z_t^*(0)$. Thus from Lemma 27 we know that the solution $(x^*_{z_n^t}, y^*_{z_n^t})$ is the same on both sides of t. In particular, the ratios $\frac{y_{t-i}}{x_{t-i+1}}$ and $\frac{y_{t+i}}{x_{t+i}}$ are the same.

We use these feasible solutions for z_n^t to obtain lower bounds on $\mathrm{PoS}^f_{\mathcal{F}^*}(n)$ and hence on $\mathrm{PoS}(n)$ for functions in $\mathcal{F}_{\text{lin}}$ and $\mathcal{F}_{\text{poly}}$. For fair cost allocation we show that the solution for $z_n^*(0)$ gives the highest objective value if the tree T is the path. The resulting instance is exactly the instance used in [Bil+13] yielding the currently best known lower bounds. Thus, we show that in the class $\mathcal{F}^*$ the instance of [Bil+13] determines the PoS.

4.6.2 Lower Bounds for $\mathcal{F}_{\text{lin}}$

For functions $f_s \in \mathcal{F}_{\text{lin}}$, where $f_s(k) = s + \frac{(1-s)}{k}$, we give lower bounds on $\mathrm{PoS}^{f_s}_{\mathcal{F}^*}(n)$ by computing

$$\frac{C_{(x_z^*, y_z^*)}(S_n)}{C_{(x_z^*, y_z^*)}(T)}$$

for trees $T = T_t$ in $\mathcal{T}_n$ and $z = z_n^t$. We denote by

$$R(T_t) = \frac{C_{(x^*_{z_n^t}, y^*_{z_n^t})}(S_n)}{C_{(x^*_{z_n^t}, y^*_{z_n^t})}(T_t)}$$

the objective function value for a tree $T_t \in \mathcal{T}$.

Before doing the computations, we show that functions in $\mathcal{F}_{\text{lin}}$ satisfy the conditions of Lemma 24. We use the following facts. For $k \geq 1$ we have $k \leq 2^{k-1}$. The function $x \mapsto \frac{1+s(x-1)}{x}$ is decreasing in x for any $s \in [0,1)$. Further, any $s \in [0,1)$ satisfies $(1+s)^k \leq 1 + s(2^k - 1)$. This follows from the binomial theorem, the fact that $s \leq 1$ and that $\sum_{i=1}^k \binom{k}{i} = 2^k - 1$. From these we get (4.20) by

$$f(1)\left(\frac{f(2)}{f(1)}\right)^k = \left(\frac{1+s}{2}\right)^k \leq \frac{1+s(2^k-1)}{2^k} \leq \frac{1+s(2k-1)}{2k} = f(2k).$$

For (4.21), we additionally use $k+1 \leq 2^k$ and thus together with $k \leq 2^{k-1}$ we have $2k+1 \leq 3 \cdot 2^{k-1}$. For $k = 1$ (4.21) is trivially satisfied, hence we consider $k \geq 2$. In total we get

$$\begin{aligned}
f(3)\left(\frac{f(2)}{f(1)}\right)^{k-1} &= \frac{1}{3} s f(2)^{k-1} + \frac{2}{3} f(2)^k \\
&\leq \frac{1}{3} s \cdot \frac{1+s(2^{k-1}-1)}{2^{k-1}} + \frac{2}{3} \cdot \frac{1+s(2^k-1)}{2^k} \\
&= \frac{1+s(2^k + s(2^{k-1}-1))}{3 \cdot 2^{k-1}} \\
&\leq \frac{1+s(2^k + 2^{k-1}-1)}{3 \cdot 2^{k-1}} = \frac{1+s(3 \cdot 2^{k-1}-1)}{3 \cdot 2^{k-1}}
\end{aligned}$$

$$\leq \frac{1 + s(2k + 1 - 1)}{2k + 1} = f(2k + 1)$$

which shows that f_s satisfies (4.21).

We can thus use (LP_z^T) and consider the solution (x_z^*, y_z^*) for z_n^t (see (4.36) and (4.37)). The objective function value, i.e., the lower bound on the PoS can be computed from (4.38) and (4.39). The remaining choice is which $T \in \mathcal{T}$ to consider.

The Path First, we take the path and look at the objective function values depending on s. Note that $z_n^1 = z_n^*(0)$. Figure 4.19 shows the plots of $R(T_1)$ depending on parameter s for some values of $n \in \{2, \ldots, 100\}$. A zoom into small values of s is shown in Figure 4.20. Note that we compute only a lower bound on the PoS by taking $\frac{C(S_n)}{C(T)}$.

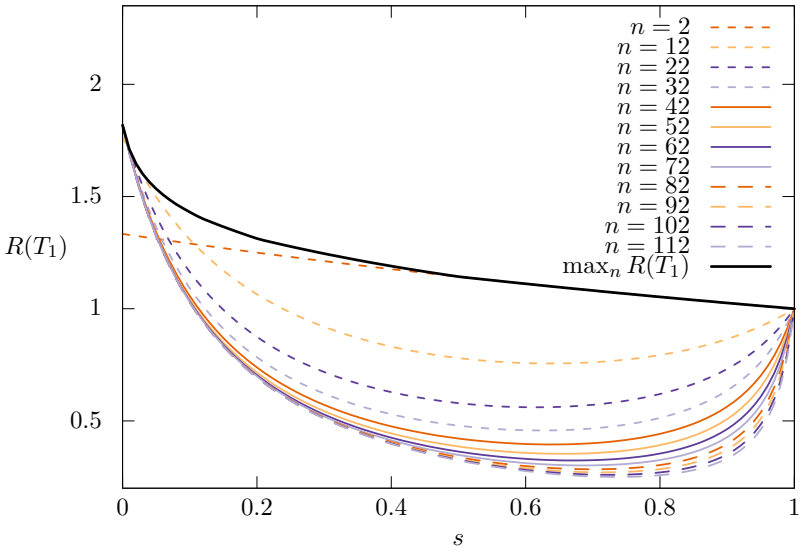


Figure 4.19: Values of $R(T_1)$ depending on $s \in [0, 1)$ for different values of n. The black curve shows the maximum over $n \in \{2, \ldots, 120\}$ for each s. In the main part, the lines are ordered from top to bottom by increasing n, i.e., $n = 22$ for the top most dashed purple line at $s = 0.4$.

We do not check that T is actually a social optimum, we just use the fact that the cost of a social optimum is at most that of T. Hence, it is possible that $R(T)$ drops below 1, this happens in particular if $C(S_n) \leq C(T)$. The values at the boundary are already known. For $s = 0$, the value is the currently highest lower bound from [Bil+13] for fair cost allocation (approx. 1.81). For $s = 1$ the value is 1 as there are no sharing effects and every player chooses a shortest path which is the social optimum. Intuitively, the PoS should decrease with decreasing sharing effects (increasing s) as the players get closer to choosing shortest paths.

From the plots we observe that for fixed n the ratio $R(T_1)$ first decreases and later increases when s goes from 0 to 1. While we do not fully understand the behavior, there are three aspects influencing the ratio which partly work against each other.

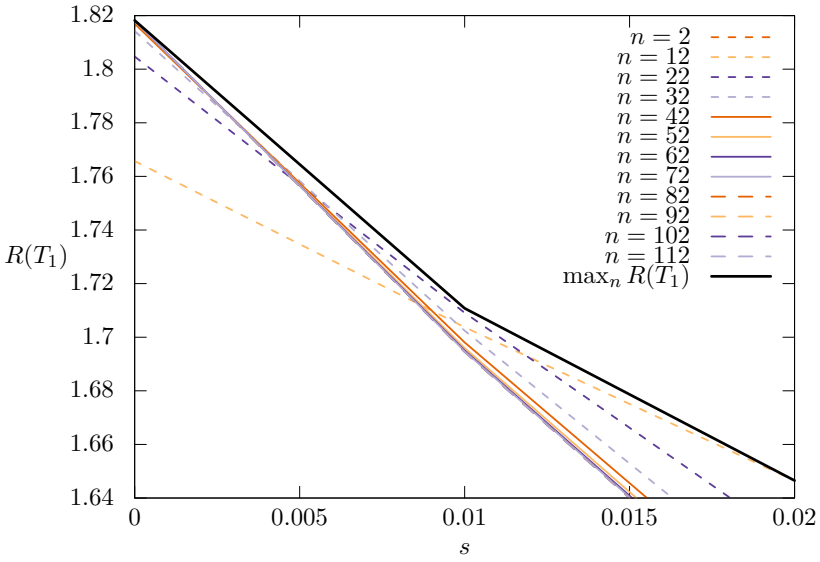
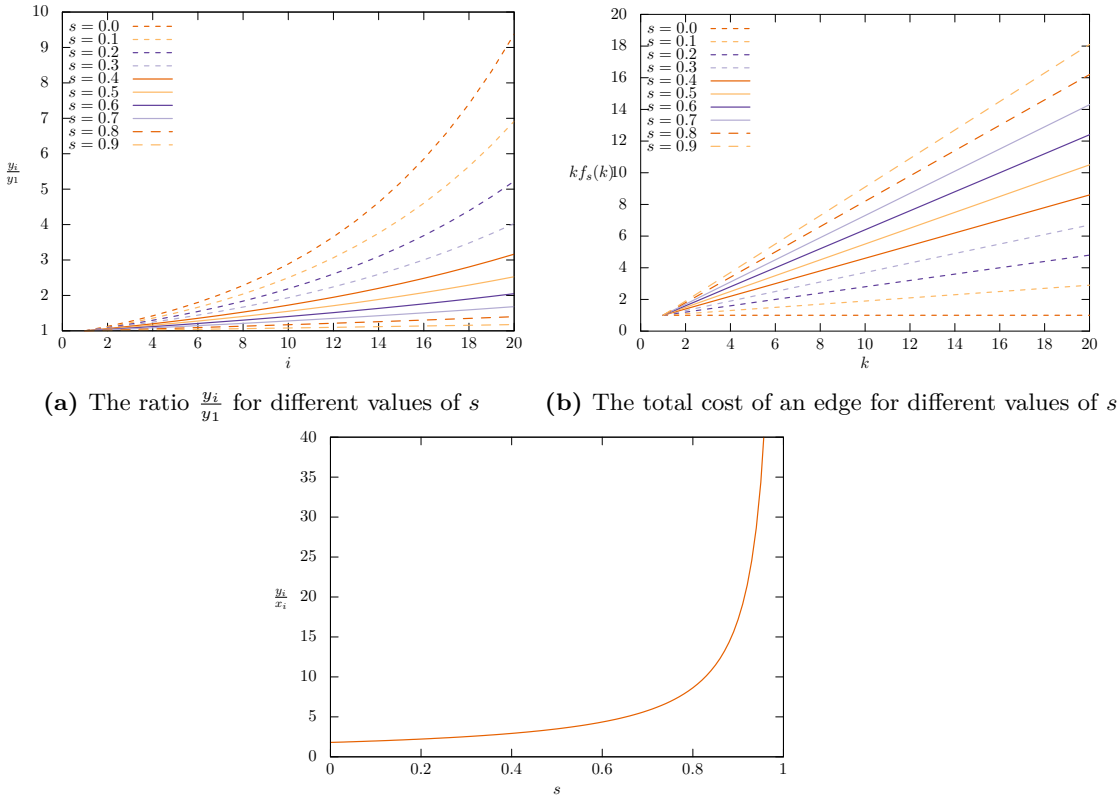


Figure 4.20: Zoom into Figure 4.19 for small values $s \in [0, 0.02]$. Note that here the relative order of lines changes compared to Figure 4.19 (while still having the same legend). The bottom dashed purple line at $s = 0.005$ corresponds to $n = 22$.

First of all we consider the structure of the social cost. For fair cost allocation the contribution of an edge to the social cost is independent of the number of players using the edge. This is not the case for sharing functions with economies of scale. In particular, every edge gets the factor $n_\sigma(e)f(n_\sigma(e))$. For fixed n and increasing s this factor gets larger as can be seen in Figure 4.21b. Especially when looking at the star as the best equilibrium, these factors only appear in the social cost of tree T, as in the star every edge is used by only one player. Hence, intuitively the ratio $R(T_1)$ should be decreasing in s.

Ignoring the factors $n_\sigma(e)f(n_\sigma(e))$, we express $R(T_1)$ as a weighted harmonic mean of $\frac{y_i}{x_i}$ with weight y_i and 1 with weight y_1 as in (4.41). We will now discuss how the weights y_i and the ratios $\frac{y_i}{x_i}$ change with s. From (4.43) we see an exponential increase in the y values. Every y_i is a factor of $\frac{f(1)+f(2)}{f(1)+f(3)} = \frac{3}{4} + \frac{3}{2(s+1)}$ larger than the previous y_{i-1}. In particular, we have $y_i = y_1\left(\frac{f(1)+f(2)}{f(1)+f(3)}\right)^{i-1}$ for $i \in \{2, \dots, n-1\}$. If we look at these factors in dependence of s (Figure 4.21a), we see that they get smaller for larger s. Hence, for large s we have small weights in the weighted harmonic mean. If x was independent of y also the values of the harmonic mean decrease, and hence intuitively $R(T_1)$ should decrease.

On the other hand, we have $x_i = \frac{f(1)^2 - f(2)f(3)}{f(1)(f(1)+f(2))} y_i$. Note that the ratio $\frac{y_i}{x_i} = \frac{3(3+s)}{6-(1+s)(2s+1)}$ is independent of i. But for s going near 1 the denominator approaches 0 and hence the ratio grows very large (see Figure 4.21c). Thus for s near 1 also $R(T_1)$ grows. Since $\frac{y_i}{x_i}$ grows very slowly in the beginning, this increasing effect is overtaken by the decreasing effects explained above for small s.



(a) The ratio $\frac{y_i}{y_1}$ for different values of s

(b) The total cost of an edge for different values of s

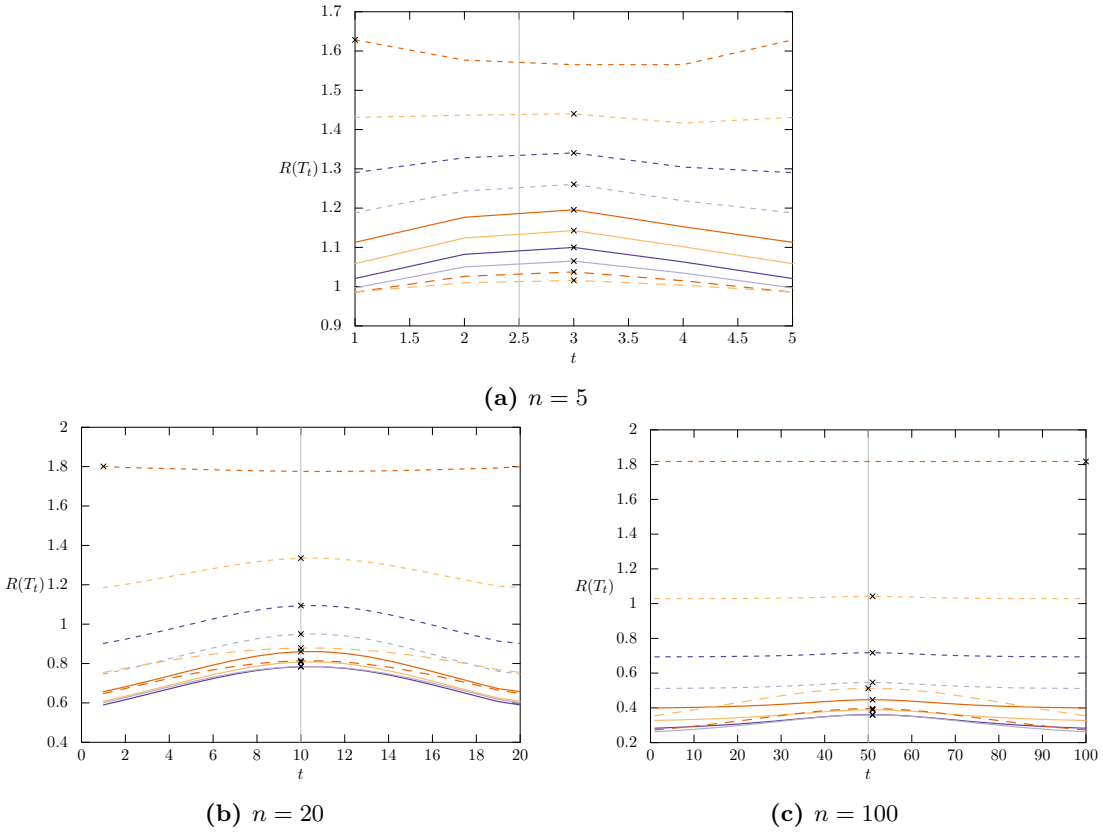
(c) Ratio $\frac{y_i}{x_i}$ depending on s

Figure 4.21: The three building blocks $\frac{y_i}{x_i}$, $\frac{y_i}{y_1}$ and $kf(k)$ of $R(T)$ and their dependency on parameter $s \in [0, 1)$.

Now consider a fixed s and varying values of n. For large n we have more values $\frac{y_i}{x_i}$ in the weighted harmonic mean. Since these values grow fast for large s, the slope at $s = 1$ is steeper for large n as can be observed in Figure 4.19. On the other hand, if s is close to 1 there are almost no sharing effects occurring. But in the path there are many edges used by more players. In fact, the path is the unique tree with highest number of shares, i.e., maximizing $\sum_{e \in \text{supp}(T)} n_\tau(e)$ among all trees with the same number of nodes. Hence, it is no surprise that for large s the ratio $R(T_1)$ drops below 1. In other words, if s is close to 1 the cost of a social optimum is the sum of the shortest paths for every player. Using the path as a shortest path for every player gives a bad bound on a social optimum. Especially for many players.

For s close to 0 we have the opposite effect. Now sharing is advantageous and as mentioned before the path is a tree with maximal sharing. For small s the social cost of a profile is independent of the numbers of players on edges. Hence using the path as an upper bound on a social optimum seems promising. Further observe from Figure 4.21a that for small s the growth of the y values is large while the ratio $\frac{y_i}{x_i}$ is small. Hence, it is beneficial to introduce many y edges.

In total we observe that for large s the maximal lower bound we get by this is achieved at small n, while for small s this changes and using more players is better.



(a) $n = 5$

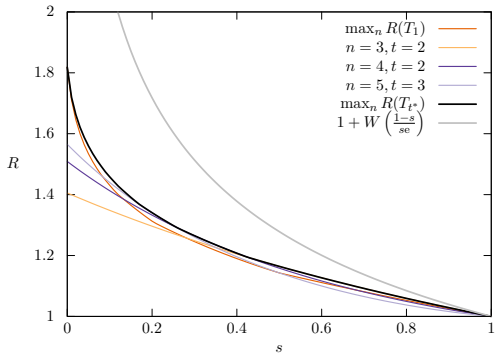
(b) $n = 20$

(c) $n = 100$

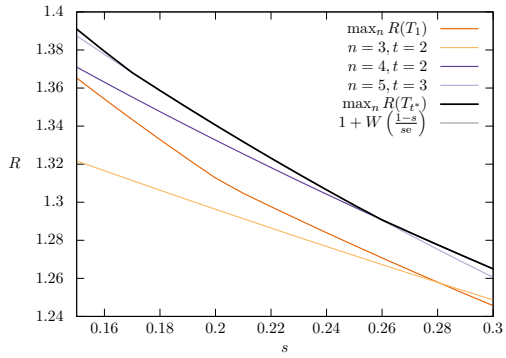
Figure 4.22: Different choices of $t \in \{1, \ldots, n\}$ depending on s for fixed values of n. We use 10 equally distributed values for $s \in [0, 1)$. The black x marks indicate a value for t giving the highest ratio $R(T_t)$ for each s. The vertical gray line indicates $\frac{n}{2}$. The color code is the same as in Figure 4.19. s increases from top to bottom in (a). The dotted orange line corresponds to $s = 0$ and the solid purple line to $s = 0.6$.

Trees $T_t \in \mathcal{T}$ The above arguments hold similarly for other trees $T_t \in \mathcal{T}$. We consider the right part from t to n as a path and have the same behavior of y and x in this part. Similarly for the left part from 1 to t, starting at t. There is a slight asymmetry as we set $z_t = 0$, hence $y_{t-1} = y_t$ and the increase in the y values starts only from there. But for large n we have the same relations on each of the sides of t. However, the stem $\{t, r\}$ is only accounted for once in $R(T_t)$. In the interpretation of $R(T_t)$ as weighted harmonic mean, we have the ratios $\frac{y_i}{x_i}$ scaled by some $kf(k)$ with weight y_i for $i \in \{1, \ldots, n\} \setminus \{t\}$, and value $\frac{f(1)}{nf(n)}$ of weight y_t for the stem $\{t, r\}$ of the tree. Since the total edge cost is non-decreasing, the value for the stem is less than any of the other values. It is thus beneficial to put a small weight on the stem relative to the other weights. Since the weights y_i grow exponentially, choosing a small t for the stem seems good. Especially

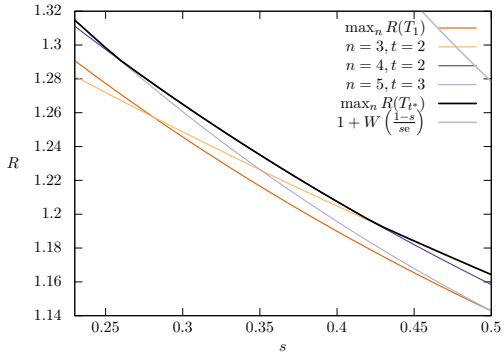
for small s where the growth in the y values is large. For large s the y values do not differ that much and hence we have to consider the factors $kf(k)$. As mentioned before the sharing effects are maximal in the path. If we want to minimize sharing effects among trees $T_t \in \mathcal{T}$, we choose t to be roughly at $\frac{n}{2}$. We always have all n players using the stem $\{t, r\}$ and some sharing effects on the paths connecting 1 and n to t. Making both paths as short as possible at the same time leads to choosing $t \approx \frac{n}{2}$. We observe from Figure 4.22 that choosing t to be approximately $\frac{n}{2}$ has a slight offset, preferring larger t's, due to the asymmetry of setting $z_t = 0$.



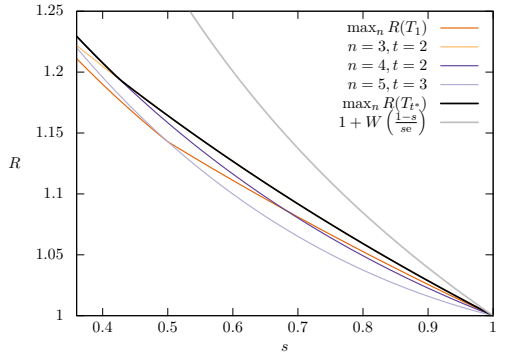
(a) Five lower bounds and the upper bound (gray) are shown for $s \in [0, 1]$.



(b) Zoom to $s \in [0.15, 0.3]$. The light purple line for $n = 5, t = 3$ gives the maximum for s in the approximate interval $[0.17, 0.26]$.



(c) Zoom to $s \in [0.23, 0.5]$. The purple line for $n = 4, t = 2$ gives the maximum for s in the approximate interval $[0.25, 0.44]$.



(d) Zoom to $s \in [0.36, 1]$. The light orange line for $n = 3, t = 2$ gives the maximum for s in the approximate interval $[0.41, 1]$.

Figure 4.23: Lower bounds on the PoS given by ratios $R(T)$ for $T \in \mathcal{T}$ together with the upper bound from Lemma 15.

Conclusion A combined plot of the lower bounds obtained by looking at $R(T)$ can be found in Figure 4.23. The maximal lower bound we computed is shown in black. For every $s \in [0, 1)$, we computed the values $R(T_{t^*})$ for $n \in \{2, \ldots, 120\}$, where t^* denotes the value for t giving the maximal ratio $R(T_t)$ for fixed s and n. We then take the maximum of all these values. For s we plot $\max_{n=2,\ldots,120} \max_{t=1,\ldots,n} R(T_t)$. Additionally

we show the maximum curve of the ratios $R(T_1)$ for paths in orange. That is, we plot the function $s \mapsto \max_{n=2,\dots,120} R(T_1)$. The plot also contains the ratio $R(T_2)$ for $n = 3$ in light orange, $R(T_2)$ for $n = 4$ in purple, and $R(T_3)$ for $n = 5$ in light purple. Finally, we include the upper bound as computed in Section 4.5 (Lemma 15) in gray.

We observe that paths already give a good approximation of the maximal lower bound, especially for small s. For larger s the maximum is given by small instances. These results are also supported by our computational experiments, where we enumerate small instances. In these experiments we fix the star to be one of the best equilibria and consider all other trees for T in the lower bound. We want to emphasize that in the experiments we consider *all* trees T and not only trees in $\mathcal{T}$. For functions in $\mathcal{F}_{\text{lin}}$, we observe that for $s \geq 0.5$ the small instance with $n = 2$ and $t = 3$ gives the highest ratio. For $s \in [0.3, 0.4]$ the instance with $n = 4$ and $t = 2$, that is the small instance from before with one more player, gives the highest ratio. Finally, for $s = 0.2$ the instance with highest ratio has $n = 5$ and $t = 3$. Since we ran experiments only for $s \in \{0.1, 0.2, \dots, 0.9\}$, we do not explicitly know, for which s the worst case instance switches. To summarize, we observe that for functions $f_s \in \mathcal{F}_{\text{lin}}$ where $s \geq 0.2$, small instances with $3, 4$ and 5 nodes give the highest lower bound on the PoS if we fix the star to be one of the best equilibria. In contrast to that, paths give high bounds for very small s.

It would be interesting to know, whether this behavior holds for all $s > 0$. That is, is the case $s = 0$ the only case, where we have to take the limit of n growing large to obtain the PoS?

Research Question 6. For functions $f_s \in \mathcal{F}_{\text{lin}}$, where $s > 0$, is the PoS given by a finite instance?

4.6.3 Lower Bounds for $\mathcal{F}_{\text{poly}}$

As in the case of $\mathcal{F}_{\text{lin}}$, we compute $R(T)$ for $T \in \mathcal{T}$ as a lower bound on the PoS. First we show that $f_\alpha \in \mathcal{F}_{\text{poly}}$, where $f_\alpha(k) = k^{\alpha-1}$, satisfies the conditions of Lemma 24.

From $k \leq 2^{k-1}$ and $2k + 1 \leq 3 \cdot 2^{k-1}$ as for $f_s \in \mathcal{F}_{\text{lin}}$ and $\alpha - 1 \leq 0$ we get

$$f(1)\left(\frac{f(2)}{f(1)}\right)^k = \left(2^k\right)^{\alpha-1} \leq (2k)^{\alpha-1} = f(2k)$$

and

$$f(3)\left(\frac{f(2)}{f(1)}\right)^{k-1} = \left(3 \cdot 2^{k-1}\right)^{\alpha-1} \leq (2k+1)^{\alpha-1} = f(2k+1).$$

Comparison to $\mathcal{F}_{\text{lin}}$ for the Fan Instance Again, we consider $R(T)$ as a weighted harmonic mean built by the fractions $\frac{y_i}{y_1}, \frac{y_i}{x_i}$ and the factors $kf(k)$. The dependence of the three building blocks on parameter α is similar to functions in $\mathcal{F}_{\text{lin}}$. That is, $\frac{y_i}{y_1}$ is increasing and $kf(k)$ is decreasing in α, and $\frac{y_i}{x_i}$ grows slowly for small α and fast for α close to 1. Hence, the curves for $R(T_t)$ and $R(T_1)$ in particular, are similar for $f_s \in \mathcal{F}_{\text{lin}}$ and $f_\alpha \in \mathcal{F}_{\text{poly}}$ (compare Figure 4.24 to Figure 4.19).

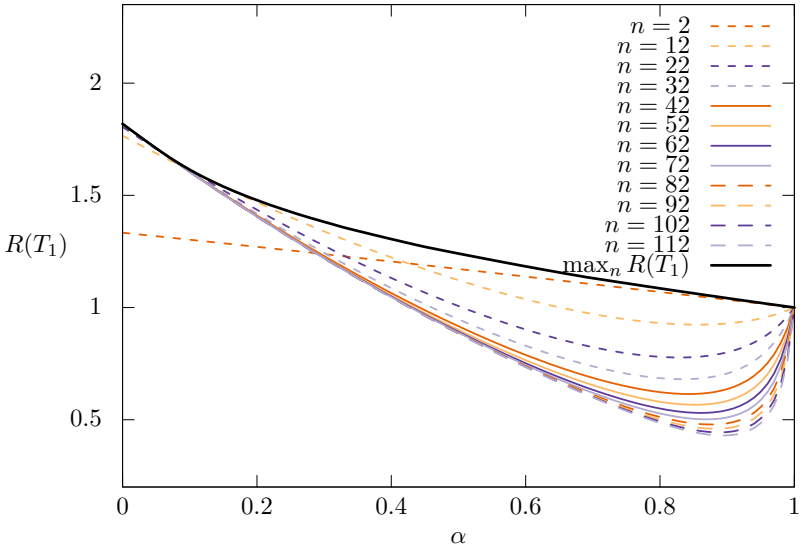
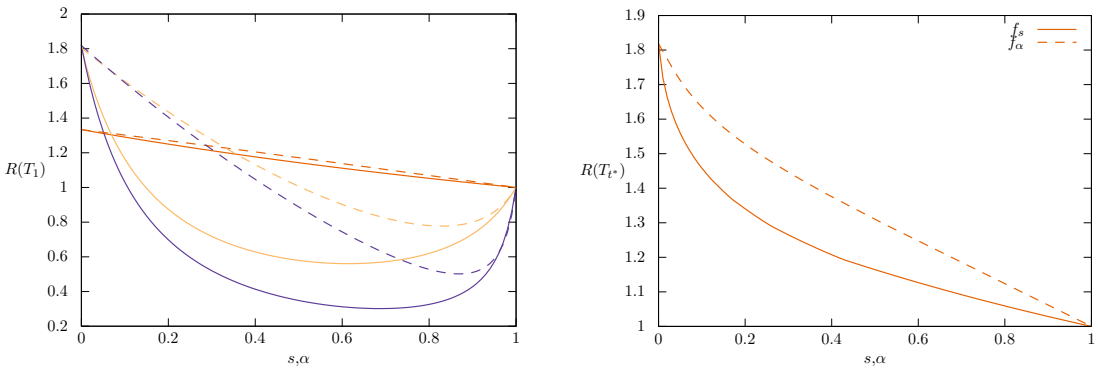


Figure 4.24: Values of $R(T_1)$ depending on $\alpha \in [0, 1)$ for different values of n. The black curve shows the maximum over $n \in \{2, \ldots, 120\}$ for each α. In the main part, the lines are ordered from top to bottom by increasing n, i.e., $n = 22$ for the top most dashed purple line at $\alpha = 0.4$.

To explain the differences in the curves for fixed n and varying α, we compare the building blocks of $R(T)$ in Figure 4.26. We observe that for a fixed parameter $p \in [0, 1)$ the factor $kf(k)$ is smaller for $f_p \in \mathcal{F}_{\text{poly}}$ than for $f_p \in \mathcal{F}_{\text{lin}}$. This hints to the shallower decrease of $R(T)$ for small α. Further, we see that the values $\frac{y_i}{x_i}$ are almost the same for p up to 0.6 and the steep increase begins later for $f_p \in \mathcal{F}_{\text{poly}}$. Together with the fact that the weights $\frac{y_i}{y_1}$ are larger for $f_p \in \mathcal{F}_{\text{poly}}$, these are further reasons explaining the shallow decrease for $f_p \in \mathcal{F}_{\text{poly}}$. In total, we observe from Figure 4.25a that the same path instance gives a higher ratio $R(T_1)$ for $\mathcal{F}_{\text{poly}}$.

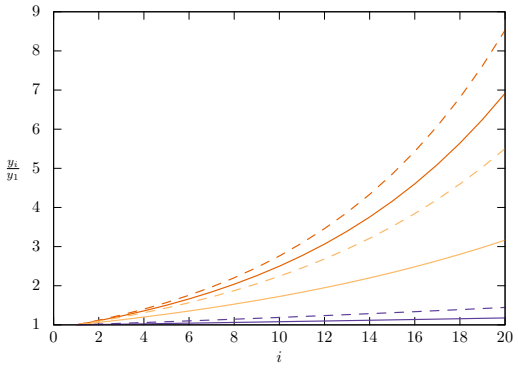


(a) $R(T_1)$ for $n = 2$ (orange), $n = 22$ (light orange) and $n = 72$ (purple).

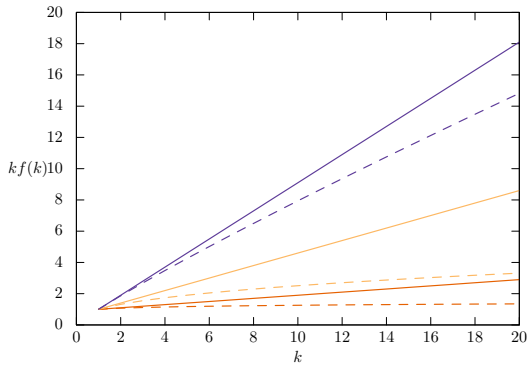
(b) $\max_n R(T_{t^*})$

Figure 4.25: Comparison of the lower bounds $R(T)$ for $f_s \in \mathcal{F}_{\text{lin}}$ (solid) and $f_\alpha \in \mathcal{F}_{\text{poly}}$ (dashed).

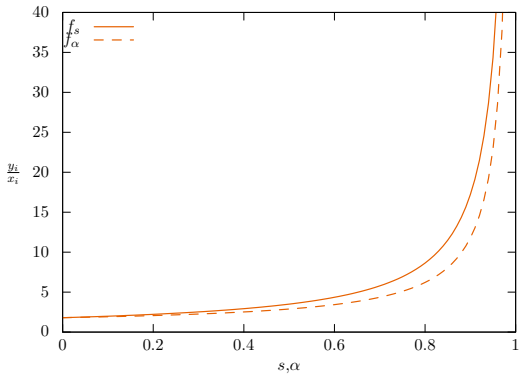
Again, we also consider other trees $T_t \in \mathcal{T}$ and compute the optimal t^* for fixed n and α. Comparing $p \mapsto \max_{n=2,\dots,120} \max_{t=1,\dots,n} R(T_t)$ in Figure 4.25b shows that the lower bound from these fan instances is higher for functions in $\mathcal{F}_{\text{poly}}$.



(a) The ratio $\frac{y_i}{y_1}$ for different values of s



(b) The total cost of an edge for different values of s



(c) Ratio $\frac{y_i}{x_i}$ depending on s

Figure 4.26: Comparison of the three building blocks $\frac{y_i}{x_i}$, $\frac{y_i}{y_1}$ and $kf(k)$ of $R(T)$ for functions $f_s \in \mathcal{F}_{\text{lin}}$ (solid lines) and $f_\alpha \in \mathcal{F}_{\text{poly}}$ (dashed lines). In (a) and (b) we consider $s, \alpha = 0.1$ (orange) $s, \alpha = 0.4$ (light orange), and $s, \alpha = 0.9$ (purple).

The Bridge Instance While the fan instance already gives good lower bounds, we additionally consider another class of instances for functions in $\mathcal{F}_{\text{poly}}$, which is better for large α. The graph B_n consists of a root node r, a node 1 and nodes $\{2, \dots, n\}$. The edges are the union of the star S_r rooted at r and the star S_1 rooted at 1. See Figure 4.27 for an illustration. The scaling factors on edges of S_r are called y and those of S_1 are called x as shown in Figure 4.27. Note that S_r and S_1 share edge $\{1, r\}$ and hence as before we consider $x \in \mathbb{R}^n$ where x_1 is never used. A *bridge instance* is given by the bridge graph B_n and scaling factors $x, y \in \mathbb{R}^n_{\geq 0}$, formally $\mathcal{B}_f(n, x, y) = (B_n, r, f, (x, y))$.

As for the fan instance, we consider bridge instances where the star rooted at r is the unique equilibrium. To get a lower bound on the PoS, we compare the cost of S_r to the cost of S_1 in these instances.

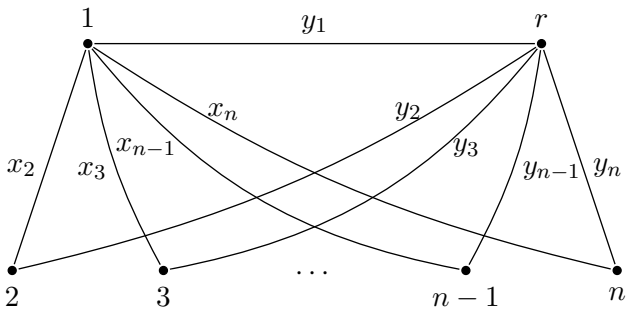


Figure 4.27: Bridge instance B_n consisting of two stars rooted at r and at 1

Lemma 28. *In a bridge instance $\mathcal{B}_f(n, x, y)$ the star S_r is the unique equilibrium, if for x and y all of the following hold:*

$$\forall i \in \{2, \dots, n\} : f(1)y_1 \leq f(1)x_i + f(2)y_i \tag{4.46}$$
$$\forall i \in \{2, \dots, n\} : f(1)y_i < f(1)x_i + f(i)y_1 \tag{4.47}$$
$$\forall 2 \leq i < j \leq n : f(1)y_i < f(1)x_i + f(i)x_j + f(i+1)y_j \tag{4.48}$$
$$\forall 2 \leq j < i \leq n : f(1)y_i < f(1)x_i + f(i-1)x_j + f(i)y_j \tag{4.49}$$
$$\tag{4.50}$$

Proof. Let $\mathcal{B}_f(n, x, y)$ be a bridge instance where x and x satisfy all conditions stated above. We show that the star S_r is an equilibrium by checking (NECond) for every edge outside of S_r. These edges are of the form $\{1, i\}$ for $i \in \{2, \dots, n\}$. From (4.46) we have

$$f(1)y_1 \leq f(1)x_i + f(2)y_i$$

and from (4.47)

$$f(1)y_i \leq f(1)x_i + f(i)y_i \leq f(1)x_i + f(2)y_i$$

which shows that (NECond) is satisfied for any edge $\{1, i\}$ outside of S_r.

Now let σ be a strategy profile in $\mathcal{B}_f(n, x, y)$ with $\text{supp}(\sigma)$ being a tree different from the star S_r. The means there is an edge $\{i, r\}$ used by more than one player. Let $\{i, r\}$ be one of those edges and denote by ℓ_i the largest index of a player using $\{i, r\}$ additionally to player i, in particular $\ell_i \neq i$. We show that ℓ_i has an improving move and together with Lemma 10 we get that there are no other equilibria than S_r.

Switching to $\{\ell_i, r\}$ incurring a cost of at most $f(1)y_{\ell_i}$ is an improving move for ℓ_i. Consider the case where $i = 1$, that is, edge $\{1, r\}$ is used at least by player 1 and player ℓ_1. The current strategy of ℓ_1 has cost at least

$$f(1)x_{\ell_i} + f(\ell_i)y_1$$

since at most all players $j \leq \ell_i$ could be using $\{1, r\}$. Hence, (4.47) shows that playing $\{\ell_i, r\}$ is an improving move.

If $i \neq 1$, we first look at the case where $\ell_i < i$. The current cost incurred to ℓ_i is at least

$$f(1)x_{\ell_i} + f(\ell_i)x_i + f(\ell_i + 1)y_i.$$

Again, at most all of the players $j \leq \ell_i$ use edge $\{i, r\}$ additionally to i. From (4.48) we see that ℓ_i has an improving move $\{\ell_i, r\}$. If $\ell_i > j$, we have to exclude player i from the above computation and obtain a lower bound on the current cost of

$$f(1)x_{\ell_i} + f(\ell_i - 1)x_i + f(\ell_i)y_i.$$

Using (4.49) concludes the proof. □

Note that as for the conditions in the fan instance x's only appear on the right hand side and thus we can weaken the inequalities in Lemma 28. For any (x, y) satisfying this weakened conditions, we can add a small $\varepsilon > 0$ to x to obtain a feasible solution to the original conditions.

For $f_\alpha \in \mathcal{F}_{\text{poly}}$ we give two vectors x and y satisfying the weak conditions of Lemma 28.

Lemma 29. *The vectors $x', y' \in \mathbb{R}^n$ defined by*

$$x'_i = y'_1 \frac{1 - (2i)^{\alpha-1}}{1 + 2^{\alpha-1}} \quad \text{and} \quad y'_i = y'_1 \frac{1 + i^{\alpha-1}}{1 + 2^{\alpha-1}}$$

for $i \in \{2, \ldots, n\}$ and $x'_1 = y'_1 > 0$ satisfy the weak inequalities of Lemma 28 for $f_\alpha \in \mathcal{F}_{poly}$.

Proof. For x', y' as defined above, the weakened inequalities (4.46) and (4.47) are satisfied with equality. For $1 \leq i < j \leq n$ we have

$$f(1)y'_i \leq f(1)x'_i + f(i)y'_1 \iff 0 \leq \left(1 + j^{\alpha-1}\right)\left((i+1)^{\alpha-1} - (2i)^{\alpha-1}\right).$$

Since $\alpha - 1 < 0$ and $i + 1 \leq 2i$ the right inequality is satisfied. For $1 \leq j < i \leq n$ we have

$$\begin{aligned} &f(1)y'_i \leq f(1)x'_i + f(i-1)x'_j + f(i)y'_j \iff \\ &0 \leq (i-1)^{\alpha-1} - (2i)^{\alpha-1} + j^{\alpha-1}\left(i^{\alpha-1} - (2(i-1))^{\alpha-1}\right). \end{aligned}$$

Again, the inequality on the right hand side is satisfied as $i - 1 \leq 2i$ and for $i \geq 2$ also $i \leq 2(i-1)$. □

For the feasible vectors x', y' from Lemma 29 we compute

$$R(\mathcal{B}) = \frac{C_{(x',y')}(S_r)}{C_{(x',y')}(S_1)} = \frac{\sum_{i=1}^n f(1)y'_i}{nf(n)y'_1 + \sum_{i=2}^n f(1)x'_i} = \frac{1 + \sum_{i=2}^n \frac{1+i^{\alpha-1}}{1+2^{\alpha-1}}}{n^\alpha + \sum_{i=2}^n \frac{1-(2i)^{\alpha-1}}{1+2^{\alpha-1}}} \tag{4.51}$$

as a lower bound for the PoS in $\mathcal{B}_f(n, x, y)$. Figure 4.28 shows this bound for different values of n depending on α.

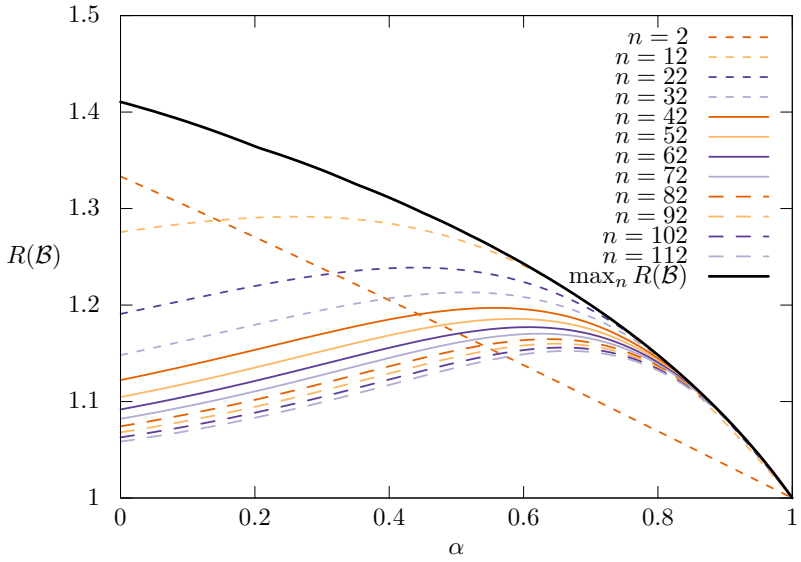


Figure 4.28: Lower bound $R(\mathcal{B})$ (from (4.51)) given by a bridge instance $\mathcal{B}$ depending on α for different values of n. The black curve shows the maximum over $n \in \{2, \dots, 120\}$ for each α.

In the combined plot in Figure 4.29 we observe that the bridge instance gives higher bounds than the fan instance for $\alpha > 0.6$. In contrast to functions in $\mathcal{F}_{\text{lin}}$, the path in the fan instance is not giving good lower bounds compared to other choices of t.

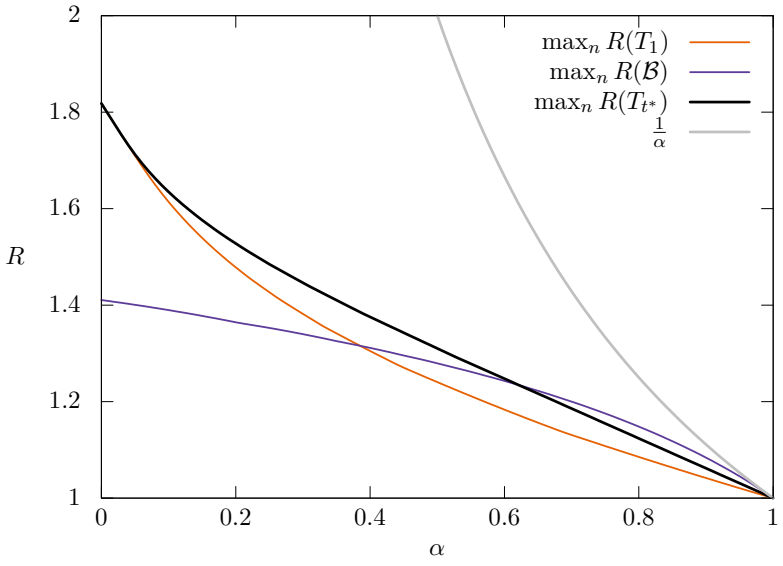


Figure 4.29: Lower bounds on the PoS for functions $f_\alpha \in \mathcal{F}_{\text{poly}}$ together with the upper bound from Lemma 16 (gray). For the fan instance we show $R(T_1)$ in orange and $R(T_t)$ for the best choice of t in black. The lower bound $\max_n R(\mathcal{B})$ from the bridge instance is shown in purple.

4.6.4 Fan Instance for Fair Cost Allocation

We are now looking at the special case of fair cost allocation. The goal of this section is to show that $\mathrm{PoS}^{\mathrm{fair}}_{\mathcal{F}^*}(n)$ is given by the path and the costs used in [Bil+13] and thus

Theorem 10.

$$\forall n \in \mathbb{N}: \quad \mathrm{PoS}^{fair}_{\mathcal{F}^*}(n) = \frac{20\left(\frac{9}{8}\right)^{n-2} - 16}{11\left(\frac{9}{8}\right)^{n-2} - 8} \leq \frac{20}{11}$$

Unfortunately, this does not yet solve the question of computing the PoS in general fan instances with fair cost allocation, since we only look at instances where the star is the best equilibrium.

Research Question 7. Is the highest PoS in a fan instance with fair cost allocation achieved by an instance where the star is one of the best equilibria?

Computational experiments suggest that this is true, even for other graph classes and general sharing functions with economies of scale. Since the star is the only profile without sharing of edges it is a natural choice for the best equilibrium in instances with high PoS, where one tries to make the social cost large. Unfortunately, we were not able to proof this intuition.

To show Theorem 10 we use the characterization of Lemma 24 and (LP^T_z). First note that for $k \geq 1$ we have $2k \leq 2^k$ and $2k + 1 \leq 3 \cdot 2^{k-1}$, and hence fair cost allocation satisfies (4.20) and (4.21) of Lemma 24. We can thus use (4.35) to write

$$\mathrm{PoS}^{\mathrm{fair}}_{\mathcal{F}^*}(n) = \max_{n' \leq n} \max_{T \in \mathcal{T}_{n'}} \max_{\substack{z \in \{0,1\}^{n'} \\ z_1 = 0, z_{n'} = 1}} \frac{C_{(x^*_z, y^*_z)}(S_{n'})}{C_{(x^*_z, y^*_z)}(T)}.$$

The proof of Theorem 10 consists of two parts. First, we determine the best choice of z for the path which gives the value as stated in the theorem. Then we show that the maximal value for any other tree is bounded by the value of the path.

Optimal choice of z for the path The social cost of the star w.r.t. (x^*, y^*) is given by the sum over all y's, while the social cost of the path is the sum over all x's. We have

$$\frac{C_{(x^*_z, y^*_z)}(S_n)}{C_{(x^*_z, y^*_z)}(T_1)} = \frac{y^*_1 + \sum_{i=2}^n y^*_i}{y^*_1 + \sum_{i=2}^n x^*_i}.$$

We maximize this value over all choices of z. We show that $z^*_n(0)$ maximizes the ratio.

We begin with the part where the star and the path are disjoint, i.e., ignoring edge $\{r, 1\}$. Define

$$\rho(z) = \frac{\sum_{i=2}^n y^*_{z,i}}{\sum_{i=2}^n x^*_{z,i}}.$$

First, we compute $\rho(z^*_n(0))$ and $\rho(z^*_n(1))$.

$\frac{y_i}{y_{i-1}}$	$z_i = 0$	$z_i = 1$
$z_{i-1} = 0$	$\frac{9}{8}$	1
$z_{i-1} = 1$	1	$\frac{8}{9}$

$\frac{x_i}{y_i}$	$z_i = 0$	$z_i = 1$
$z_{i-1} = 0$	$\frac{5}{9}$	$\frac{1}{2}$
$z_{i-1} = 1$	$\frac{2}{3}$	$\frac{8}{5}$

Table 4.1: The ratios appearing in the basic solution (x_z^*, y_z^*) for fair cost allocation depending on the entries of z.

We evaluate the appearing ratios $\frac{y_i}{y_{i-1}}$ and $\frac{x_i}{y_i}$ from (4.36) and (4.37) for the possible choices of z_{i-1} and z_i in Table 4.1. With these values we get

$$\rho(z_n^*(0)) = \frac{\sum_{i=2}^{n-1}\left(\frac{9}{8}\right)^{i-1} y_1^* + \left(\frac{9}{8}\right)^{n-2} y_1^*}{\sum_{i=2}^{n-1}\frac{5}{9}\left(\frac{9}{8}\right)^{i-1} y_1^* + \frac{1}{2}\left(\frac{9}{8}\right)^{n-2} y_1^*} = \frac{10\left(\frac{9}{8}\right)^{n-2} - 9}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-2} - 5}. \tag{4.52}$$

For $\zeta = 1$ we first consider $n \geq 3$

$$\rho(z_n^*(1)) = \frac{y_1^* + \sum_{i=3}^{n-1}\left(\frac{9}{8}\right)^{i-2} y_1^* + \left(\frac{9}{8}\right)^{n-3} y_1^*}{\frac{2}{3}y_1^* + \sum_{i=3}^{n-1}\frac{5}{9}\left(\frac{9}{8}\right)^{i-2} y_1^* + \frac{1}{2}\left(\frac{9}{8}\right)^{n-3} y_1^*} = \frac{10\left(\frac{9}{8}\right)^{n-3} - 8}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-3} - \frac{13}{3}} \tag{4.53}$$

and for $n = 2$ we get

$$\rho(z_2^*(1)) = \frac{y_2^*}{x_2^*} = \frac{\frac{8}{9}y_1^*}{\frac{5}{8}\frac{8}{9}y_1^*} = \frac{8}{5} = \frac{10\left(\frac{9}{8}\right)^{2-3} - 8}{\frac{11}{2}\left(\frac{9}{8}\right)^{2-3} - \frac{13}{3}}.$$

Observation 7. We have the following properties:

- $\rho(z_n^*(0))$ is decreasing in n and lower bounded by $\frac{20}{11}$
- $\rho(z_n^*(1))$ is increasing in n and upper bounded by $\frac{20}{11}$
- for every $n \geq 2$ we have $\rho(z_n^*(1)) \leq \rho(z_n^*(0))$.

We show that $z_n^*(\zeta)$ maximizes $\rho(z)$. The proofs are different for $\zeta = 1$ and $\zeta = 0$.

Lemma 30. *For $n \in \mathbb{N}_{\geq 2}$ both*

$$\max_{\substack{z\in\{0,1\}^n \\ z_1=1, z_n=1}} \frac{\sum_{i=2}^{n} y_{z,i}^*}{y_{z,1}^*} \qquad \textit{and} \qquad \max_{\substack{z\in\{0,1\}^n \\ z_1=1, z_n=1}} \rho(z)$$

are maximized for $z_n^*(1)$.

Proof.

First Maximum. For the first maximum, observe from Table 4.1 that the ratio $\frac{y_z^*, i}{y_{z,i-1}^*}$ is maximized for $z_{i-1} = z_i = 0$ for $i \in \{3, \ldots, n-1\}$. Also for $z_2 = 0$ and $z_{n-1} = 0$ the ratios are maximal, since we fixed $z_1 = 1 = z_n$.

Second Maximum. Since $z_n^*(1)$ is a valid choice for the maximum, we only need to show that no other z gives a higher ratio. Let $n \geq 2$ and take some $z \in \{0,1\}^n$ with $z_1 = 1$ and $z_n = 1$ which is different from $z_n^*(1)$.

We decompose z into parts of the form $1, 0, \ldots, 0, 1$ as shown in Figure 4.30. Let $1 \leq i_\ell \leq n$ be the positions of 1 in z, i.e., $z_{i_1} = z_1 = 1$. Define $\bar{z}^\ell$ to be the part of z between two 1s, i.e., $\bar{z}^\ell = (z_{i_\ell}, \ldots, z_{i_{\ell+1}})$. With this we have $\bar{z}^\ell = z^*_{|\bar{z}^\ell|}(1)$ for every $\ell \geq 1$, where we denote by $|\bar{z}^\ell|$ the length of part $\bar{z}^\ell$,i.e., $i_{\ell+1} - i_\ell + 1$. Let K be the number of parts in this decomposition, i.e., $i_{K+1} = n$. Note that since $z \neq z_n^*(1)$ this decomposition divides z in at least two parts and every part has length less than n.

$$\begin{array}{rccccccccccccccc} & & i_1 & & & & i_2 & & & & i_3 & i_4 & & & i_5 & \\ z = (& & 1, & 0, \ldots, 0, & & & 1, & 0, \ldots, 0, & & & 1, & 1, & 0, \ldots, 0, & & 1 &) \\ & & \underbrace{\qquad\qquad\qquad} & & & & \underbrace{\qquad\qquad\qquad} & & & & \underbrace{\qquad} & \underbrace{\qquad} & & & \underbrace{\qquad} & \\ & & \bar{z}^1 & & & & \bar{z}^2 & & & & \bar{z}^3 & & \bar{z}^4 & & & \end{array}$$

Figure 4.30: z is decomposed into four parts of the form $1, 0 \ldots, 0, 1$.

We write

$$\rho(z) = \frac{\sum_{j=i_1+1}^{i_2} y_j^* + \ldots + \sum_{j=i_K+1}^{i_{K+1}} y_j^*}{\sum_{j=i_1+1}^{i_2} x_j^* + \ldots + \sum_{j=i_K+1}^{i_{K+1}} x_j^*} \leq \max_{\ell=1,\ldots,K} \frac{\sum_{j=i_\ell+1}^{i_{\ell+1}} y_j^*}{\sum_{j=i_\ell+1}^{i_{\ell+1}} x_j^*}.$$

Observe that each of the ratios on the right hand side corresponds to $\rho\left(\bar{z}^\ell\right)$. If we set y_1^* to y_{z,i_ℓ}^* for every part $\bar{z}^\ell$, the basic solution $(x^*_{\bar{z}^\ell}, y^*_{\bar{z}^\ell})$ coincides with the respective part of (x_z^*, y_z^*), i.e., $y^*_{\bar{z}^\ell} = (y^*_{z,i_\ell}, \ldots, y^*_{z,i_{\ell+1}})$ and $x^*_{\bar{z}^\ell} = (x^*_{z,i_\ell}, \ldots, x^*_{z,i_{\ell+1}})$.

Since $\rho(z_n^*(1))$ is increasing in n (Observation 7) we obtain

$$\rho(z) \leq \max_{\ell=1,\ldots,K} \rho\left(z^*_{|\bar{z}^\ell|}(1)\right) \leq \rho\left(z^*_{\max_{\ell=1,\ldots,K}|\bar{z}^\ell|}(1)\right) \leq \rho(z_n^*(1)).$$

□

Since $\rho(z_n^*(0))$ is decreasing in n, the above proof can not be used for $\zeta = 0$, but the analogous statement holds nevertheless.

Lemma 31. *For $n \in \mathbb{N}_{\geq 2}$ both*

$$\max_{\substack{z \in \{0,1\}^n \\ z_1=0, z_n=1}} \frac{\sum_{i=2}^n y^*_{z,i}}{y^*_{z,1}} \qquad \textit{and} \qquad \max_{\substack{z \in \{0,1\}^n \\ z_1=0, z_n=1}} \rho(z)$$

are maximized for $z_n^*(0)$.

Proof.

First Maximum. As in Lemma 30, we observe from Table 4.1 that the ratio $\frac{y^*_{z,i}}{y^*_{z,i-1}}$ is maximized for $z_{i-1} = z_i = 0$ for $i \in \{2, \ldots, n-1\}$. Since we again fixed $z_n = 1$, also the choice of $z_{n-1} = 0$ remains the best.

Second Maximum. We show that no $z \neq z_n^*(0)$ gives a higher ratio than $z_n^*(0)$. Recall from Observation 7 that $\rho(z_n^*(0)) > \frac{20}{11}$. Now assume $z \neq z_n^*(0)$ gives the maximum and has a higher ratio. We split z in two parts. Let i be the index of the first 1 in z. We consider the first part $\bar{z}^1 = (z_1, \dots, z_{i+1})$ and the second part $\bar{z}^2 = (z_{i+1}, \dots, z_n)$. Since $z \neq z_n^*(0)$ we have $i < n$ and hence these parts are well-defined. We show that replacing z_i by 0 increases the ratio and hence is a contradiction to z being optimal. Denote by z' the vector resulting from z by replacing z_i to be 0. We split z' in the same way as z and have $\bar{z}'^1 = (z'_1, \dots, z'_{i+1})$ and $\bar{z}'^2 = (z'_{i+1}, \dots, z'_n) = \bar{z}^2$. Let $(x, y) = (x_z^*, y_z^*)$ and $(x', y') = (x_{z'}^*, y_{z'}^*)$.

We consider $\rho(z)$ as weighted harmonic mean of $\rho(\bar{z}^1)$ and $\rho(\bar{z})^2$ with weights $\sum_{j=2}^{i+1} y_j$ and $\sum_{j=i+2}^{n} y_j$ respectively. For $\rho(z')$ we take the analogous representation. We have $\rho(\bar{z}^2) = \rho(\bar{z}'^2)$.

We show $\rho(\bar{z}^1) \leq \rho(\bar{z}^2)$, $\rho(\bar{z}^1) \leq \rho(\bar{z}'^1)$ and $\sum_{j=2}^{i+1} y_j \leq x \sum_{j=2}^{i+1} y'_j$ and $\sum_{j=i+2}^{n} y_j = x \sum_{j=i+2}^{n} y'_j$ for some $x \in \mathbb{R}_{\geq 0}$. With Lemma 2 this gives the contradiction $\rho(z) \leq \rho(z')$.
Case $z_{i+1} = 0$: First observe that changing z_i to 0 increases y_i and y_{i+1} by a factor of $\frac{9}{8}$ and $\left(\frac{9}{8}\right)^2$ respectively, and hence all y_j in the second part by a factor of $\left(\frac{9}{8}\right)^2$. Thus

$$\sum_{j=2}^{i+1} y_j \leq \left(\frac{9}{8}\right)^2 \sum_{j=2}^{i+1} y'_j \qquad \text{and} \qquad \sum_{j=i+2}^{n} y_j = \left(\frac{9}{8}\right)^2 \sum_{j=i+2}^{n} y'_j.$$

Consider $\rho(\bar{z}^1)$ and $\rho(\bar{z}'^1)$ as weighted harmonic mean of the corresponding $\frac{y_j}{x_j}$ with weight y_j. The mean of the first $i-1$ elements is $\frac{9}{5}$, since $z_j = z'_j = 0$ for all $j \in \{1, \dots, i-1\}$ with (4.37).

We compute the mean of elements i and $i+1$ in $\bar{z}^1$ as

$$\frac{y_i + y_{i+1}}{y_i \frac{x_i}{y_i} + y_{i+1} \frac{x_{i+1}}{y_{i+1}}} = \frac{2}{\frac{1}{2} + \frac{2}{3}} = \frac{12}{7}.$$

For $\bar{z}'^1$ this mean is again $\frac{9}{5}$. As $\frac{12}{7} < \frac{9}{5} < \frac{20}{11}$, we have

$$\rho\left(\bar{z}^1\right) < \rho\left(\bar{z}'^1\right) < \frac{20}{11}.$$

From our choice of z we have $\rho(z) > \frac{20}{11}$. Since the first part has ratio less than $\frac{20}{11}$, the ratio of the second part has to be larger than $\frac{20}{11}$. We thus have,

$$\rho\left(\bar{z}^1\right) < \rho\left(\bar{z}'^1\right) < \rho\left(\bar{z}^2\right) = \rho\left(\bar{z}'^2\right).$$

From Lemma 2 we get the contradiction $\rho(z) \leq \rho(z')$.
Case $z_{i+1} = 1$: As in the previous case observe that changing z_i to 0 increases y_i and y_{i+1} by a factor of $\frac{9}{8}$, and also all y's in the second part by $\frac{9}{8}$. Thus

$$\sum_{j=2}^{i+1} y_j \leq \frac{9}{8} \sum_{j=2}^{i+1} y'_j \qquad \text{and} \qquad \sum_{j=i+2}^{n} y_j = \frac{9}{8} \sum_{j=i+2}^{n} y'_j.$$

Again, the mean of the first $i-1$ elements is $\frac{9}{5}$ and we compute the mean of elements i and $i+1$ for $\bar{z}^1$ as

$$\frac{y_i + y_{i+1}}{y_i \frac{x_i}{y_i} + y_{i+1} \frac{x_{i+1}}{y_{i+1}}} = \frac{1 + \frac{8}{9}}{\frac{1}{2} + \frac{8}{9} \cdot \frac{5}{8}} = \frac{306}{171}$$

and for $\bar{z}'^1$ as

$$\frac{y'_i + y'_{i+1}}{y'_i \frac{x'_i}{y'_i} + y'_{i+1} \frac{x'_{i+1}}{y'_{i+1}}} = \frac{2}{\frac{5}{9} + \frac{1}{2}} = \frac{36}{19}.$$

We have $\frac{306}{171} < \frac{20}{11} < \frac{36}{19}$. Thus, as before we have $\rho(\bar{z}^1) \leq \rho(\bar{z}^2)$ and $\rho(\bar{z}^1) \leq \rho(\bar{z}'^1)$ and get the contradiction by Lemma 2. □

With these lemmas we give the best choice of z for the path.

Lemma 32. *For $n \in \mathbb{N}_{\geq 2}$ and $\zeta \in \{0,1\}$ the maximum*

$$\max_{\substack{z \in \{0,1\}^n \\ z_1 = \zeta, z_n = 1}} \frac{C_{(x_z^*, y_z^*)}(S_n)}{C_{(x_z^*, y_z^*)}(T_1)}$$

is attained at $z_n^*(\zeta)$. The value is

$$\frac{10\left(\frac{9}{8}\right)^{n-2} - 8}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-2} - 4} \text{ for } \zeta = 0 \quad \text{and} \quad \frac{10\left(\frac{9}{8}\right)^{n-3} - 7}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-3} - \frac{10}{3}} \text{ for } \zeta = 1.$$

Proof. The values for $z_n^*(\zeta)$ follow directly from (4.52) and (4.53) as we add y_1^* in the numerator and denominator. Observe that both ratios are increasing in n and > 1.

To show that $z_n^*(\zeta)$ maximizes the ratio, take any $z \in \{0,1\}^n$ with $z_1 = \zeta$ and $z_n = 1$. Let $(x, y) = (x_z^*, y_z^*)$ and $(x', y') = (x_{z_n^*(\zeta)}^*, y_{z_n^*(\zeta)}^*)$. We consider $\frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_1)}$ as weighted harmonic mean of 1 with weight y_1 and $\rho(z)$ with weight $\sum_{i=2}^n y_i$.

If $\rho(z) \leq 1$, we immediately have $\frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_1)} \leq 1 < \frac{C_{(x',y')}(S_n)}{C_{(x',y')}(T_1)}$.

In the other case, Lemma 30 and Lemma 31 give $\rho(z) \leq \rho(z_n^*(\zeta))$ and $\frac{\sum_{i=2}^n y_i}{y_1} \leq \frac{\sum_{i=2}^n y'_i}{y'_1}$. Since scaling the weights in the harmonic mean by the same factor does not change the mean, and increasing both the value and the weight of the larger element increases the mean (see Lemma 1), we obtain

$$\begin{aligned}
\frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_1)} &= \mathcal{H}\left((y_1, 1), \left(\sum_{i=2}^n y_i, \rho(z)\right)\right) &&= \mathcal{H}\left((1,1), \left(\frac{\sum_{i=2}^n y_i}{y_1}, \rho(z)\right)\right) \\
&\leq \mathcal{H}\left((1,1), \left(\frac{\sum_{i=2}^n y'_i}{y'_1}, \rho(z_n^*(\zeta))\right)\right) &&= \frac{C_{(x',y')}(S_n)}{C_{(x',y')}(T_1)}.
\end{aligned}$$

□

Observation 8. For fixed $n \geq 2$ the value of the maximum in Lemma 32 for $\zeta = 1$ is smaller than the value for $\zeta = 0$.

Other trees than the path We will now show that no other tree T_t in $\mathcal{T}$ than the path gives a higher ratio. Using the symmetry from Lemma 27 we are able to apply Lemma 30 and Lemma 31 to the left part of T_t, i.e., for $\{1,\dots,t\}$. For these lemmas we need $z_n = 1$ and hence, we are now fixing z_1 to 0 so that after the transformation of Lemma 27 $\overleftarrow{z}_n = 1$.

Lemma 33. *For $n \in \mathbb{N}_{\geq 2}$ the maximum*

$$\max_{T\in\mathcal{T}_n} \max_{\substack{z\in\{0,1\}^n \\ z_1=0, z_n=1}} \frac{C_{(x_z^*,y_z^*)}(S_n)}{C_{(x_z^*,y_z^*)}(T)}$$

is attained for T_1 and $z_n^*(0)$.

Proof. We show that all $T_1 \neq T_t \in \mathcal{T}$ have smaller ratio than T_1. Recall from Lemma 32 that the ratio for T_1 (and $\zeta = 0$) is $\frac{10\left(\frac{9}{8}\right)^{n-2}-8}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-2}-4}$, which is greater than 1 and increasing in n.

Consider T_t for some $t \in \{2,\dots,n-1\}$ and a $z \in \{0,1\}^n$ with $z_1 = 0$ and $z_n = 1$. Let $(x,y) = (x_z^*, y_z^*)$. We introduce the short notations

$$R(n) = \frac{10\left(\frac{9}{8}\right)^{n-2} - 8}{\frac{11}{2}\left(\frac{9}{8}\right)^{n-2} - 4} \qquad \text{and} \qquad R_{T_t} = \frac{C_{(x,y)}(S_n)}{C_{(x,y)}(T_t)}.$$

We split the instance in three parts: the left part $L = \{1,\dots,t-1\}$, the middle part (the stem) $M = \{t\}$, and the right part $R = \{t+1,\dots,n\}$. Define the ratios

$$\rho_L = \frac{\sum_{i=1}^{t-1} y_i}{\sum_{i=1}^{t-1} x_{i+1}} \quad \text{and} \quad \rho_M = \frac{y_t}{y_t} \quad \text{and} \quad \rho_R = \frac{\sum_{i=t+1}^{n} y_i}{\sum_{i=t+1}^{n} x_i},$$

with weights $w_L = \frac{\sum_{i=1}^{t-1} y_i}{y_t}$, $w_M = 1$, and $w_R = \frac{\sum_{i=t+1}^{n} y_i}{y_t}$ such that

$$R_{T_t} = \mathcal{H}((w_L,\rho_L),(w_M,\rho_M),(w_R,\rho_R)) \leq \max\{\rho_L,\rho_M,\rho_R\}.$$

If this maximum is $\rho_M = 1$, then we immediately have $R_{T_t} < R(n)$.

In the case where the maximum is larger than 1 assume $\rho_L \leq \rho_R$. Applying Lemma 30 and Lemma 31 to the right part yields $\rho_R \leq \rho(z^*_{n-t+1}(z_t))$ and $w_R \leq w'_R$, where w'_R is the ratio of the y's to y_t for $z^*_{n-t+1}(z_t)$ (the first ratio in the statement of Lemma 30 and Lemma 31). Thus increasing the value and the weight of the largest element gives (see Lemma 1)

$$R_{T_t} \leq \mathcal{H}((w_L,\rho_L),(w_M,\rho_M),(w'_R,\rho(z^*_{n-t+1}(z_t)))).$$

Now consider the right part together with the stem and define $R' = \{t,\dots,n\}$ and

$$\rho_{R'} = \frac{y_t + \sum_{i=t+1}^{n} \bar{y}_i}{y_t + \sum_{i=t+1}^{n} \bar{x}_i} \qquad \text{and} \qquad w_{R'} = 1 + \frac{\sum_{i=t+1}^{n} \bar{y}_i}{y_t},$$

where $(\bar{x}, \bar{y})$ is the basic solution to $z^*_{n-t+1}(z_t)$. We have

$$\rho_{R'} = \mathcal{H}((w_M, \rho_M), (w'_R, \rho(z^*_{n-t+1}(z_t))))$$

and thus $R_{T_t} \leq \mathcal{H}((w_L, \rho_L), (w_{R'}, \rho_{R'}))$.

If $\rho_L \leq \rho_{R'}$, then we get the bound $R_{T_t} < R(n)$ from Lemma 32 as the right part R' is a path instance of size $2 \leq n-t+1 \leq n-1$ and the ratio for $\zeta = 0$ is pointwise larger than the ratio for $\zeta = 1$ (Observation 8).

If on the other hand $\rho_L > \rho_{R'}$, we apply Lemma 30 and Lemma 31 to the left part using Lemma 27. This gives $\rho_L \leq \rho(z^*_t(z_t))$ and $w_L \leq w'_L$, where w'_L is again the ratio of the y's to y_t for $z^*_t(z_t)$.

Similarly to before, we get

$$R_{T_t} \leq \mathcal{H}((w'_L, \rho(z^*_t(z_t))), (w_M, \rho_M), (w'_R, \rho(z^*_{n-t+1}(z_t)))).$$

Evaluating the right hand side for $z_t = 0$ and $z_t = 1$ using (4.36) and (4.37), finally shows $R_{T_t} < R(n)$ also in this case.

If $\rho_l > \rho_R$ the arguments from above hold, when exchanging L and R everywhere and using Lemma 27. □

This concludes the proof of

Theorem 10.

$$\forall n \in \mathbb{N}: \quad \mathrm{PoS}^{fair}_{\mathcal{F}^*}(n) = \frac{20\left(\frac{9}{8}\right)^{n-2} - 16}{11\left(\frac{9}{8}\right)^{n-2} - 8} \leq \frac{20}{11}$$

We wrote

$$\mathrm{PoS}^{\mathrm{fair}}_{\mathcal{F}^*}(n) = \max_{n' \leq n} \quad \max_{T \in \mathcal{T}_{n'}} \quad \max_{\substack{z \in \{0,1\}^{n'} \\ z_1 = 0, z_{n'} = 1}} \frac{C_{(x^*_z, y^*_z)}(S_{n'})}{C_{(x^*_z, y^*_z)}(T)}$$

with (4.35) and showed that no $T \in \mathcal{T} \setminus \{T_1\}$ has higher ratio than T_1 (Lemma 33), and that the maximum for T_1 is exactly the given right hand side (Lemma 32).

4.6.5 Computational Experiments

The instances studied in the previous sections for functions in $\mathcal{F}_{\mathrm{lin}}$ and $\mathcal{F}_{\mathrm{poly}}$ are the result of computational experiments. The experiments show that those instances are among those with highest PoS in a restricted class of instances. For fair cost allocation and small number of players (up to 4) the experiments confirm that the instances considered in [Bil+13] give the highest PoS among all instances.

Method Similarly to the previous sections (see Section 4.6.1) we fix the underlying graph and try to find edge costs such that the PoS of the resulting instance is as high

as possible. For input n and underlying cost function f the implementation computes a lower bound on $\mathrm{PoS}^f(n)$ in uniform broadcast games by bounding

$$\max_{\substack{\text{graph } G \text{ with } n \text{ nodes} \\ \text{rooted at } R}} \max_{\text{tree } T_{\mathrm{OPT}} \text{ in } G} \max_{\text{tree } T_{\mathrm{NE}} \text{ in } G} \tag{4.54}$$

$$\max\{r \in \mathbb{R} \mid \exists \text{ scaling factors } \gamma \in \mathbb{R}^E : \frac{C(T_{\mathrm{NE}})}{C(T_{\mathrm{OPT}})} > r \wedge$$

$$T_{\mathrm{OPT}} \text{ is a social optimum and } T_{\mathrm{NE}} \text{ is one of the best equilibria in } (G, R, f, \gamma)\}.$$

We do not enumerate all graphs G with n nodes but rather take the complete graph K_n. For a (not complete) graph G we can add edges with high scaling factor compared to the factors on edges in G without changing the value of the PoS. Hence, looking at the complete graph removes the outer maximization and does not change the value of (4.54). The remaining maximizations are realized by enumeration. A program enumerates the two trees T_{OPT} and T_{NE} in the complete graph and the value r. For each choice it constructs a satisfiability problem for the conditions on the scaling factors and calls a satisfiability solver. As long as the formula is satisfiable, r is increased and the solver is called again for the new formula. With this procedure (see Algorithm 2) we get a lower bound on the innermost maximization and hence a lower bound on the total value. In the following we describe the method in more detail.

Maximizing Over Trees. For the tree T_{OPT} in the outer maximization we enumerate all unlabeled rooted trees on n nodes, of which there are $1, 1, 2, 4, 9, 20, 48, 115$ for $n = 1, \ldots, 8$ [Inc21]. For T_{NE} we have to consider all n^{n-2} labeled trees on n nodes. In total we enumerate all $6, 64, 1125$ pairs of trees for $n = 3, \ldots, 5$. As the number of trees gets large quickly (the value for $n = 6$ is 25920), we reduce the search space for larger n. Firstly, we fix T_{NE} to be the star rooted at R and enumerate all unlabeled rooted trees for T_{OPT}. Secondly, we do not consider the complete graph, but only the union of the two trees T_{NE} and T_{OPT}. This gives only lower bounds on $\mathrm{PoS}^f(n)$ but we believe that our restrictions are not decreasing the value.

For every pair of T_{OPT} and T_{NE} we have to solve the inner maximization, that is, find scaling factors for the edges such that T_{OPT} is a social minimum, T_{NE} is one of the best equilibria, and the ratio of their social costs is maximal. We propagate the currently highest achieved ratio to the remaining pairs of trees. If we found scaling factors for the trees $(T_{\mathrm{OPT}}, T_{\mathrm{NE}})$ giving a ratio of r_1, we try to find scaling factors for the remaining tree pairs that achieve a ratio of at least r_1 (see the recursive calls in Lines 7 and 13). In this way we reduce the number of constraint systems to be solved.

Maximizing the Ratio r. For a pair of trees $(T_{\mathrm{OPT}}, T_{\mathrm{NE}})$ the inner maximization is lower bounded by iteratively increasing r (see Function `inner()`). If the constraint system is satisfiable for r we compute the PoS of the instance corresponding to the satisfying assignment. Note that we are guaranteed that T_{OPT} is a social optimum and T_{NE} is one of the best equilibria and hence the PoS of the instance is just the ratio of the social cost of both trees and can easily be computed from the satisfying assignment. In the next call of the procedure we replace r by the maximum of the new PoS and r plus some step size (see Line 19). On the one hand, we want to use the new PoS as new bound

```
Function Bound(n, step)
1  let opts = unlabeled trees on n nodes rooted at R
2  let nes = labeled trees on n nodes rooted at R
3  let maxOpts(ts, r) =                                  -- maximum over opt trees
4      if null ts then r
5      else
6          let (T_OPT, os) = (head(ts), tail(ts))
7          maxOpts(os, maxNEs(T_OPT, nes, r))
8      end
9  let maxNEs(T_OPT, ts, r) =                            -- maximum over ne trees
10     if null ts then r
11     else
12         let (T_NE, ns) = (head(ts), tail(ts))
13         maxNEs(T_OPT, ns, inner(T_OPT, T_NE, r, step))
14     end
15 maxOpts(opts, 1)

Function inner(T_OPT, T_NE, r, step)                     -- maximum over r
16 repeat
17     let γ = solve constraint system for T_OPT, T_NE, and r   -- call SMT solver
18     if sat then                                       -- γ is a satisfying assignment
19         Update r to max{ C^γ(T_NE)/C^γ(T_OPT), r + step }
20     end
21 until unsat
22 return r
```

Algorithm 2: Computing a lower bound on $\mathrm{PoS}^f(n)$ by enumerating the trees T_{OPT} and T_{NE}, and using an SMT-solver.

since we know that an instance with this value exists. On the other hand, if we only use this value, the procedure may progress in very small improvements of the ratio (it does). Hence we add a step size to decrease the number of steps taken to terminate. However, this does not solve the maximization problem exactly. The step size gives the accuracy of the returned bound. We end with a ratio r for which an instance with PoS r exists and the guarantee that the PoS of the graph is smaller than r plus the step size.

Since the PoS is greater than 1, we start with $r = 1$. If we assume that the PoS is less than 2 (no larger values have been observed in the currently known lower bounds) then each inner maximization takes at most $\frac{1}{\text{step size}}$ iterations. Notice, that we have to solve at least one system for every pair of trees. This is the unsatisfiable system giving us an upper bound on the PoS in the instance. This upper bound allows us to continue with the next pair of trees.

The Constraint System. We now describe the key step of generating the constraint system for the inner maximization, i.e., Line 17. We express the three conditions on the scaling factors (T_{OPT} is a social minimum, T_{NE} is one of the best equilibria, and the ratio of their social costs is strictly larger than r) in QF_LRA, the language of quantifier free formulas over the linear fragment of the theory of reals. The variables are the scaling factors.

The first condition is modeled as

$$\forall \text{ tree } T : C(T) \geq C(T_{\text{OPT}}),$$

where the universal quantifier is expressed by the conjunction of the inequalities for every tree T. Observe, that for uniform games the social cost is indeed linear in the scaling factors.

The condition of T_{NE} being one of the best equilibria is represented by a formula of the form

$$\forall \text{ tree } T : T \text{ is equilibrium } \implies C(T_{\text{NE}}) \leq C(T).$$

For the premise of this implication we use the characterization of Lemma 11, where we express the universal quantifier by combining all instances of the subformula. For every edge we take the inequalities (NECond) and combine all of those with a conjunction.

In the following we show parts of the realization of the constraints in QF_LRA for $n = 4$, underlying function $f_{0.5} \in \mathcal{F}_{\text{lin}}$, $r \approx 1.163$, and the trees T_{OPT} and T_{NE} rooted at 1 and consisting of the edges $\{\{1,2\},\{2,3\},\{2,4\}\}$ and $\{\{1,2\},\{1,3\},\{1,4\}\}$, respectively.

```
1  (declare-fun var () Real)      -- edge {1,2}
2  (declare-fun var_1 () Real)    -- edge {1,3}
3  (declare-fun var_2 () Real)    -- edge {1,4}
4  (declare-fun var_3 () Real)    -- edge {2,3}
5  (declare-fun var_4 () Real)    -- edge {2,4}
6  (declare-fun var_5 () Real)    -- edge {3,4}
7
8  -- T_OPT is social minimum (for any tree T: C(T) ≥ C(T_OPT))
9  (assert
10   (>=
11     (+ (* (/ 3 2) var) (* (/ 1 1) var_2) (* (/ 1 1) var_3))
12     (+ (* (/ 2 1) var) (* (/ 1 1) var_3) (* (/ 1 1) var_4)) ))
13 ...
14
15 -- T_NE is equilibrium using (NECond)
16 (assert (>= (+ var_3 (* (/ 3 4) var_1)) (* (/ 1 1) var)))   -- 2 -> 3
17 (assert (>= (+ var_3 (* (/ 3 4) var)) (* (/ 1 1) var_1)))   -- 3 -> 2
18 ...
19
20 -- T_NE is best equilibrium (for any tree T: T is equilibrium ⟹ C(T_NE) ≤ C(T))
21 (assert
22   (=>
23     (and
24       (and
```

```
25          (>= (+ var_1 (+ (* (/ 2 3) var) (* (/ 3 4) var_3))) (/ 0 1))
26          (>= (+ var_1 (/ 0 1)) (+ (* (/ 3 4) var) (* (/ 1 1) var_3))) )
27        (and
28          (>= (+ var_4 (* (/ 3 4) var_2)) (* (/ 3 4) var))
29          (>= (+ var_4 (* (/ 2 3) var)) (* (/ 1 1) var_2)) )
30        (and
31          (>= (+ var_5 (* (/ 3 4) var_2)) (+ (* (/ 3 4) var) (* (/ 1 1) var_3)))
32          (>= (+ var_5 (+ (* (/ 2 3) var) (* (/ 3 4) var_3))) (* (/ 1 1) var_2)) ) )
33      (<=
34        (+ (* (/ 1 1) var) (* (/ 1 1) var_1) (* (/ 1 1) var_2))
35        (+ (* (/ 3 2) var) (* (/ 1 1) var_2) (* (/ 1 1) var_3)) ) ) )
36  ...
37
38  -- C(T_NE) > r · C(T_OPT)
39  (assert
40    (> (+ (* (/ 1 1) var) (* (/ 1 1) var_1) (* (/ 1 1) var_2))
41       (* (/ 4131348405888421 3552935091602100)
42          (+ (* (/ 2 1) var) (* (/ 1 1) var_3) (* (/ 1 1) var_4)) ) ) )
43
44  (assert (>= var (/ 0 1)))
45  (assert (>= var_1 (/ 0 1)))
46  (assert (>= var_2 (/ 0 1)))
47  (assert (>= var_3 (/ 0 1)))
48  (assert (>= var_4 (/ 0 1)))
49  (assert (>= var_5 (/ 0 1)))
```

The example uses the syntax of the SMT-LIB Standard Version 2.6 [BFT17]. First, the real variables for every edge are declared. The variable corresponding to edge $\{1,2\}$ is called var as shown by the comment on the right. Then follows the condition that T_{OPT} is a social optimum. This is expressed as: the social cost of any tree is at least the social cost of T_{OPT}. In Lines 9-12 the corresponding inequality for the tree consisting of the edges $\{\{1,2\},\{1,4\},\{2,3\}\}$ is shown. The standard uses prefix notation for operators. Note that $\{1,2\}$ is used by two players and hence the total cost of this edge evaluates to $2f_{0.5}(2)\gamma_{\{1,2\}} = \frac{3}{2}\text{var}$. Next, the condition that T_{NE} is an equilibrium is stated using (NECond). We show only the two inequalities for edge $\{2,3\}$ in Lines 16 and 17. In Lines 21 - 35, we see the constraint that if T is an equilibrium, then its cost must be at least the cost of T_{NE} for the tree T consisting of edges $\{\{1,2\},\{1,4\},\{2,3\}\}$. We see the implication `=>` at top level followed by the (NECond) inequalities for the edges not in T. There is a corresponding statement for every tree. The bound of the ratio of the social costs of T_{OPT} and T_{NE} is shown in Lines 39-42. Note that $r \approx 1.163$ is written with its exact rational representation. This value for r appears as a PoS of some instance in the enumeration process. Since they heavily depend on the satisfying assignments returned by the solver, these ratios are not necessarily "nice". Finally, all variables are constrained to be non-negative.

These constraint systems are generated by a Haskell [Mar10] program using the smtlib2-library [Gün17]. This library is a type-safe interface to communicate with SMT solvers,

realized as embedded domain specific language. The code producing the constraints for T_{NE} being one of the best equilibria looks as follows:

```
1  -- ne is best equilibrium
2  -- (for any tree st: st is equilibrium ⟹ C(ne) ≤ C(st))
3  forM_ sts $ \st -> do
4    if st /= toChildMap ne  -- st is not the tree ne
5      then assert $
6        -- st is equilibrium
7        and' (map
8              (\e -> do
9                ((vul, vur), (uvl, uvr)) <- cnstr l xs (toTree (root opt) st) e
10               -- (NECond) for edge e outside st
11               (vul .>=. vur) .&. (uvl .>=. uvr) )
12             (es L.\\ edges st) )
13       .=>.
14       -- C(ne) ≤ C(st)
15       soccost l xs ne .<=. soccost l xs (toTree (root opt) st)
16   else assert true  -- no constraint needed for ne
```

We see features of the embedded language: the `assert` statement in Lines 5 and 16, the implication `.=>.` in Line 13, inequalities `.>=.` and `.<=.` in Lines 11 and 15 and a conjunction `.&.` in Line 11. Further, there are features of the host language like `forM_`, `$`, `->`, `do`, `map`. The function `cnstr` constructs the left and right hand sides of the (NECond) inequalities given a cost sharing scheme (`l`), the variables for the edges (`xs`), a rooted tree, and an edge (`e`). The complete program can be found at https://github.com/44534/networkdesign/commit/b2b048536834d62c179b8f411ba81fd158c2b92a.

For every pair of trees $(T_{\mathrm{OPT}}, T_{\mathrm{NE}})$ and every step for r one constraint system is generated and solved. Each system has a real variable for every edge of the (complete) graph. The largest part of the formula are the constraints for T_{NE} being one of the best equilibria. For every tree in the (complete) graph there are inequalities for all edges outside of the tree in the premise of the implications. Hence, the size of the formula is roughly the number of trees times the number of edges. There are n^{n-2} labeled trees on n nodes and $\frac{n(n-1)}{2}$ edges in the complete graph, hence the size of the formula is roughly n^n. This evaluates to $1, 4, 27, 256, 3125, 46656$ for $n = 1, \ldots, 6$.

Software and Hardware Specification. Our program is compiled with the Glasgow Haskell Compiler (GHC) [MP12] (version 8.10.4[1]). The constraint systems are solved by the SMT solver OpenSMT2 [Hyv+16] (version 2.0.2[2]). The time to solve one system varies from seconds to several minutes. The computations were executed on two machines. The computations for $f_s \in \mathcal{F}_{\mathrm{lin}}$ were conducted on a Fedora 31 system on a computer from the LRZ Compute Cloud[3] with 10 Intel Xeon (Skylake, IBRS) processors with 2.4 GHz and 45 GiB RAM. The computations for $f_\alpha \in \mathcal{F}_{\mathrm{poly}}$ and for fair cost allocation

[1] https://www.haskell.org/ghc/download_ghc_8_10_4.html
[2] https://github.com/usi-verification-and-security/opensmt/commit/334e87cf0adbd60c9e04219bad1a04ea36d32a6c
[3] https://doku.lrz.de/display/PUBLIC/Compute+Cloud

were excecuted on a Fedora 34 system on a machine with 24 Intel Xeon (E5-2620 0) processors with 2.3 GHz and 64 GiB RAM.

	s	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
sat	nr	106	101	75	46	94	77	82	78	76
	max	20.29	23.40	28.38	17.41	22.86	25.29	35.24	40.60	41.47
	min	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.14	0.15
	avg	4.77	3.28	3.65	5.37	3.01	3.22	4.50	5.07	5.88
unsat	max	48.15	43.00	66.24	49.12	36.64	67.96	59.19	54.19	97.97
	min	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.15
	avg	3.47	3.28	4.05	3.46	2.70	4.07	4.88	4.85	6.48
total	hours	1.22	1.12	1.34	1.15	0.92	1.34	1.62	1.62	2.16

	α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
sat	nr	94	69	81	116	47	63	45	51	44
	max	342.74	319.11	438.46	773.67	328.94	251.49	341.55	295.52	910.08
	min	0.33	0.40	0.33	0.33	0.31	0.19	0.19	0.19	0.19
	avg	69.71	90.83	98.62	138.37	119.47	86.25	70.17	85.06	116.70
unsat	max	776.95	781.51	720.25	1418.25	520.93	560.37	659.28	707.50	794.33
	min	0.30	0.30	0.26	0.27	0.28	0.16	0.16	0.16	0.16
	avg	46.41	43.39	39.56	95.14	34.39	33.40	39.02	38.98	42.67
total	hours	16.32	15.30	14.58	34.19	12.30	11.94	13.07	13.38	14.76

Table 4.2: Computation statistics for the complete enumeration with $n = 5$ nodes (4 players), step size 0.001, and functions $f_s \in \mathcal{F}_{\text{lin}}$ and $f_\alpha \in \mathcal{F}_{\text{poly}}$. The first part shows statistics for satisfiable constraint systems. We give the number of satisfiable systems and the maximum (max), minimum (min) and average (avg) time in seconds to solve one system. The same times are shown for the systems which are not satisfiable. Finally, we give the total computation time in hours.

Execution Time We give an impression of the number of iterations taken and the computation time for the complete enumeration for $n = 5$ and a step size of 0.001. In Table 4.2 we show some metrics depending on parameter s for $f_s \in \mathcal{F}_{\text{lin}}$ and α for $f_\alpha \in \mathcal{F}_{\text{poly}}$.[4] We show the number (nr) of satisfiable constraint systems solved during the complete enumeration. Recall, that for every pair of tree we encounter exactly one system that is not satisfiable, hence there are 1125 unsat systems for every parameter. We further give the maximum (max), minimum (min), and average (avg) time (in seconds) the solver takes to solve one of the systems. Finally, the total computation time (total) for the complete enumeration is given in hours.

We observe that the solver times vary among the parameters and that the case of functions $f_s \in \mathcal{F}_{\text{lin}}$ is considerably faster than the case of functions $f_\alpha \in \mathcal{F}_{\text{poly}}$. This

[4] data generated by the following calls to the program: `networkdesign --pos --enum --all 5 -b opensmt -c 1 --start 1 --normalize -s 0.001 --sharing2 1.1 --sharingslope 0.1 --complete --debug "results/linear/times/complete-s01-5.log"` for $f_{0.1} \in \mathcal{F}_{\text{lin}}$ and replacing `--sharing2` and `--sharingslope` with `--power 0.1` for $f_{0.1} \in \mathcal{F}_{\text{poly}}$.

may be due to the complexity of the numbers appearing in the constraints. For f_s the coefficients of the scaling factors are of the form $n_\sigma(e)f_s(n_\sigma(e)) = 1+(s-1)n_\sigma(e)$ and for f_α we get coefficients of the form $(n_\sigma(e))^\alpha$. We further see that there are big differences in the maximal and minimal time needed to solve the constraint systems.

For the complete enumeration for $n = 6$ nodes, we observed a significant increase in the average solver time for one system. For $f_{0.1} \in \mathcal{F}_{\text{lin}}$ we have an average (over the first 263 instances) time of around 4 h. Recall that there are 25920 pairs of trees in the complete enumeration for 6 nodes. Using the average of 4 h the enumeration would take roughly 12 years. For $f_{0.1} \in \mathcal{F}_{\text{poly}}$ we see an average (over the first 53 instances) time of roughly 19 h. Hence, we were not able to completely enumerate instances with more than 5 nodes (4 players).

Results The complete enumerations for small values of $n = 3, 4, 5$ result in instances with highest bound on the PoS where only edges in the union of the two trees T_{OPT} and T_{NE} are used. We thus believe that the answer to Research Question 5 is No. Further, in these instances the star is one of the best equilibria (see top parts of Tables 4.3 and 4.4). This suggests that the answer to Research Question 7 is Yes, even for all (not only fan) instances. Hence we reduce the search space for $n \geq 6$ by fixing T_{NE} to be the star and considering only the union of the two trees (instead of the complete graph).

In Tables 4.3 and 4.4 we show instances giving the highest bound achieved by Algorithm 2[5]. Note that these are just representatives, there may be more instances giving the same value[6]. The step size is 0.001. Hence, the values given in the tables are lower bounds on the PoS in the instances and the PoS is at most the value plus 0.001. The corresponding scaling factors for the edges can be found at `https://github.com/44534/networkdesign/tree/master/results`.

How to read Table 4.3. Instances with n nodes can be extended to instances with $n+1$ nodes without chaging the PoS by adding a single edge with scaling factor 0, connecting the new node to the root R (and putting a high scaling factor on all other edges incident to the new node in complete graphs). Whenever this construction happens, we do not draw the graph for $n+1$. For example, the graph for $s = 0.2$ and $n = 7$ is the one resulting from the graph for $n = 6$ by adding an isolated edge as described above. With the above extension it is immediate that the PoS is increasing with the number of nodes (for a fixed value of s). In some columns of Table 4.3 the values are not monotone. This is a consequence of computing just a lower bound within an accuracy given by the step size. The values depend on the satisfying assignments found by the SMT solver. We

[5]Data generated by `networkdesign --pos --enum --all 5 -b opensmt -c 1 --start 1 --normalize -s 0.001 --sharing2 1.1 --sharingslope 0.1 --complete | tee results/linear/complete-s01-5.txt` for $f_{0.1} \in \mathcal{F}_{\text{lin}}$ and $n \leq 5$. For fixing T_{NE} to be the star and considering only the union of the two trees for $n \geq 6$, replace `--all 5 --complete` by `--star 6 --continue`. Data for $\mathcal{F}_{\text{poly}}$ can be produced by replacing `--sharing2` and `--sharingslope` by `--power`.

[6]Those can be found with the program (see `https://github.com/44534/networkdesign` for a more detailed documentation of the program and its options). Example call for $f_{0.5} \in \mathcal{F}_{\text{lin}}, n = 6$, and ratio 1.1638: `networkdesign --find --star 6 -b opensmt --start 1.1638 --pos --sharing2 1.5 --sharingslope 0.5 --normalize -s 0.001 -c 4 +RTS -N2`

chose to show this numbers to be consistent with the output of the program. Whenever the underlying graphs are the same we can take the maximum of the computed bounds. For example for $s = 0.9$, the underlying graph is the same for all $n \geq 4$. The highest bound found for this graph is 1.0279 for $n = 5$. This bound is valid for all $n \geq 4$. But the SMT solver found satisfying assignments for other values of n which are already close to the actual PoS and hence our algorithm may terminate with smaller bounds (as 1.0277 for $n = 4$). We show the maximal bounds in bold.

We observe from Table 4.3 that for $s \geq 0.5$ the small instance with $n = 3$ and $t = 2$ gives the highest bounds for all $n \geq 4$. For s between 0.3 and 0.4 the small instances with $n = 4$ and $t = 2$ achieves the highest bound and for $s = 0.2$ the instance with $n = 5$ and $t = 3$ gives the highest bound. For $s = 0.1$ the tree $T_{\lceil \frac{n}{2} \rceil}$ gives the highest bound. This is evidence that the fan instances we studied in Section 4.6.2 are optimal (among all instances for $n \leq 5$ and for $n \geq 6$ among instances where the star is fixed to be one of the best equilibria and considering only the union of the two trees) for functions in $\mathcal{F}_{\text{lin}}$.

In Table 4.4 we see that for small $\alpha \leq 0.2$ the path and $T_{\lceil \frac{n}{2} \rceil}$ give the highest bounds. However, for larger α and n we observe a new structure for T_{OPT} giving the highest bounds. Here we no longer have a tree of the form T_t, but rather a star with arms of different length. The simplified version where T_{OPT} is another star (with all arms of length 1) gives the bridge instances studied in Section 4.6.3.

For fair cost allocation the experiments show that the fan instances of Bilò et al. [Bil+13] are optimal. For $n = 3, 4, 5$ complete enumeration of all pairs of trees in the complete graph, results in the fan instances giving the highest bound. The same is true for $n \geq 6$ and our restricted search space. This suggest that the fan instance is optimal in the class of all broadcast games and not only in the class of fan instances where the star is fixed to be the unique equilibrium. Hence, this is evidence that the answer to Research Question 7 is Yes.

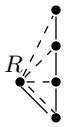
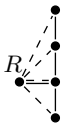
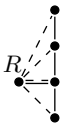
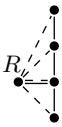
n \ s	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
3	1.2897 	1.2495 	1.2112 	1.1756 	1.1420 	1.1110 	1.0809 	1.0521 	1.0252 
4	1.3900 	1.3116 	1.2482 	1.2040 	**1.1638** 	1.1259 	1.0915 	1.0590 	1.0277 
5	1.4245 	1.3318 	1.2644 	**1.2068** 	1.1636	**1.1267**	1.0910	1.0583	**1.0279**
6	1.4396 	**1.3398** 	**1.2645**	1.2065	1.1637	1.1265	**1.0917**	**1.0591**	1.0277
7	1.4544 	1.3398	1.2645	1.2065	1.1637	1.1265	1.0917	1.0591	1.0277
8	1.4574 	1.3398	1.2645	1.2065	1.1637	1.1265	1.0917	1.0591	1.0277

Table 4.3: Lower bounds on $\mathrm{PoS}^f(n)$ for functions $f_s \in \mathcal{F}_{\text{lin}}$ returned by Algorithm 2 with step size 0.001 together with the trees achieving this bound. T_{OPT} is drawn solid and T_{NE} is drawn dashed. For $n = 3, 4, 5$ we completely enumerated all pairs of trees in the complete graph. For $n > 5$ the star rooted at R is fixed to be one of the best equilibria and we conisder only the union of the two trees. A missing graph means that the graph is the same as the one above in the same column. For example the instance giving a lower bound of 1.2077 for $s = 0.9$ and $n = 7$ is the same as the instance for $n = 4$. Bounds marked in bold are the highest bound for the same underlying graph (for fixed s).

n \ α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
3	1.3014	1.2701	1.2374	1.2043	1.1709	1.1373	1.1029	1.0690	1.0346
4	1.4319	1.3790	1.3269	1.2759	1.2323	1.1896	1.1445	1.0978	1.0490
5	1.5009	1.4307	1.3780	1.3297	1.2797	1.2269	1.1727	1.1161	1.0578
6	1.5410	1.4637	1.4108	1.3550	1.2992	1.2408	1.1836	1.1249	1.0638
7	1.5655	1.4921	1.4311	1.3699	1.3143	1.2554	1.1959	1.1337	1.0680
8	1.5819	1.5070	1.4456	1.3873	1.3274	1.2656	1.2039	1.1387	1.0710

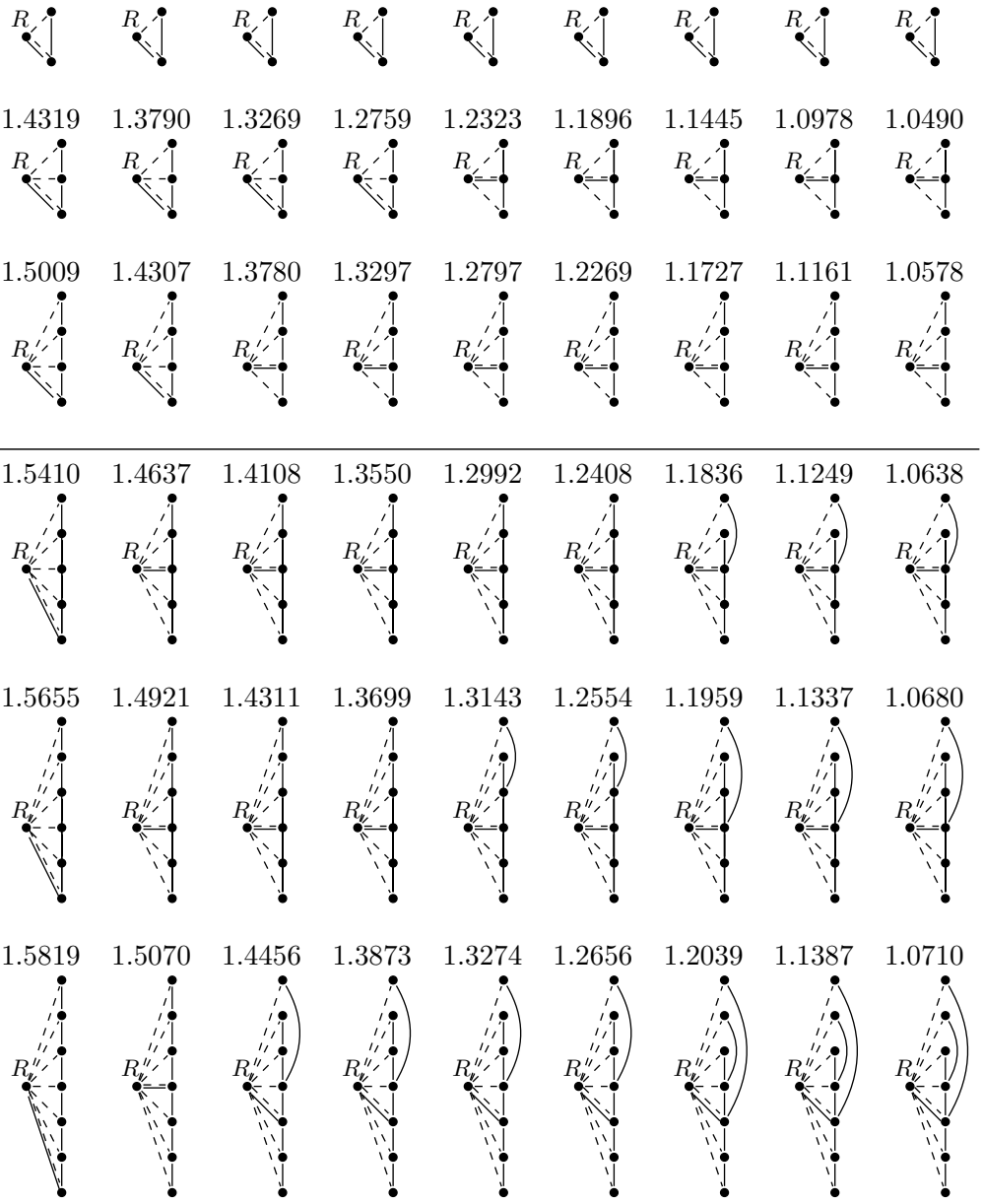


Table 4.4: Lower bounds on $\mathrm{PoS}^f(n)$ for functions $f_\alpha \in \mathcal{F}_{\mathrm{poly}}$ returned by Algorithm 2 with step size 0.001 together with the trees achieving this bound. T_{OPT} is drawn solid and T_{NE} is drawn dashed. For $n = 3, 4, 5$ we completely enumerated all pairs of trees in the complete graph. For $n > 5$ the star rooted at R is fixed to be one of the best equilibria and we conisder only the union of the two trees.

4.7 Computing Equilibria

In the previous sections we looked at the PoS and tried to understand how far away the cheapest equilibrium can be from a social optimum. For some classes we found good (constant) bounds. A system designer is now faced with the task of proposing such a good equilibrium to the users. Hence the question is:

Can we find a good equilibrium in network games efficiently?

We take two approaches to answer this question. First, we look at the players and ask whether they can reach an equilibrium fast. Note that here we are looking for *any* equilibrium, not necessarily the best. Since equilibria are the local minimizers of the potential (as discussed in Section 2.2), this can be posed as a local search problem. A natural thing is to look at the standard improving-dynamics and see how fast it converges. We give some weak evidence that this process can be slow for multicast games with fair cost allocation. Bilò et al. [Bil+21] show that finding an equilibrium in general network games with fair cost allocation is PLS-hard. We show how their reduction can be generalized to decreasing cost functions.

Secondly, we take the system designer's view and consider finding the best equilibrium in a centralized way. We show that this is NP-hard in multi- and broadcast games for sharing functions with economies of scale. Finally, we show the same hardness for finding the global potential minimizer in broadcast games.

4.7.1 Potential Decreasing Moves for Multicast Games

A natural approach to find an equilibrium is to look at the standard improving-dynamics. Starting from any strategy profile, let the players do improving moves iteratively until they reach an equilibrium. The progress of each step is measured by the improvement in the potential. If the potential decreases at least some fixed constant amount in each improving move, we get an upper bound on the running time of the improving-dynamics.

For network games the difference in the potential of two profiles σ and σ' which differ only in the strategy of a single player is of the form

$$\Phi(\sigma) - \Phi(\sigma') = \sum_{e \in \sigma_i \setminus \sigma_i'} (F_e(n_\sigma(e)) - F_e(n_\sigma(e) - 1)) - \sum_{e \in \sigma_i' \setminus \sigma_i} (F_e(n_\sigma(e) + 1) - F_e(n_\sigma(e))).$$

For fair cost allocation this evaluates to

$$\Phi(\sigma) - \Phi(\sigma') = \sum_{e \in \sigma_i \setminus \sigma_i'} \frac{1}{n_\sigma(e)} - \sum_{e \in \sigma_i' \setminus \sigma_i} \frac{1}{n_\sigma(e) + 1}.$$

This is the case we are now focusing on. We give an example of an improving move in a multicast game where the decrease in the potential is sub-polynomial in the number of players. This is some evidence that bounding the potential improvement by a constant is not possible. However, this does not show that improving-dynamics take many steps in general. It could be that there can not be too many improving moves with only a small decrease in the potential.

Research Question 8. Is there an example of a multicast (broadcast) game with a super-polynomial sequence of improving moves?

This question is closely related to the PLS-completeness of finding an equilibrium in a multicast game, which is studied later. In particular, the answer to this question is Yes for general network games where players have different source and sink nodes. Anshelevich et al. [Ans+08a] construct a directed network game where a sequence of improving moves emulates a binary counter. For undirected networks the result follows from the tight PLS-reduction of Syrgkanis [Syr10].

Hence, we consider multicast games. Our game has $2n-2$ players, where n is some integer power of 4, i.e., there is an even $k \in \mathbb{N}$ such that $n = 2^k$. The network is shown in Figure 4.31. The idea is that there are two paths connecting the source s_i of a player

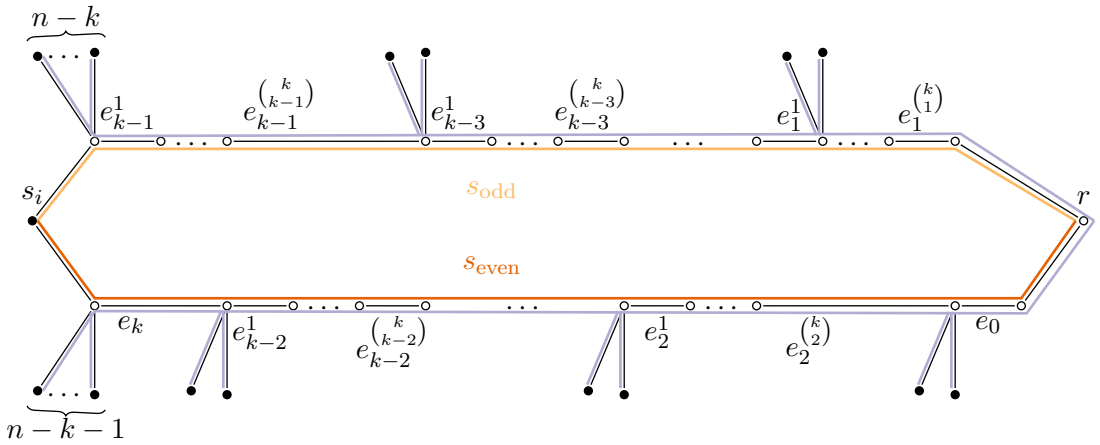


Figure 4.31: The multicast instance with a small decrease in the potential. The strategies σ of the path players are shown in light purple. Player i switches from s_{even} (orange) to s_{odd} (light orange).

i to the root. The bottom path for i is referred to as the *even path* and the top path as the *odd path*. There are edges $e^1_j, \ldots, e^{\binom{k}{j}}_j$ which form a path for every $j \in \{0, \ldots, k\}$. The total number of edges is $\sum_{j=0}^{k} \binom{k}{j} = 2^k = n$. The even path contains the paths related to even $j \in \{0, \ldots, k\}$, while the odd path contains the remaining paths related to odd $j \in \{0, \ldots, k\}$. At the beginning of each of the subpaths of some j there are 2 players joining. Except for $j = k$, where $n-k-1$ player join and for $j = k-1$ where $n-k$ players join. Hence, there are $n-1$ players associated to the even path and $n-2$ players associated to the odd path. These players are called *path players*. Source s_i and root r are connected to both paths by edges of cost zero, while all other edges have cost 1. Recall that we are considering the case of fair cost allocation.

Define the profile σ, where all path players use their canonical strategy of joining and then following the corresponding path to the root. We consider the two strategies s_{even}

and s_{odd} for player i and compute the change in the potential as

$$\Phi((s_{\text{even}}, \sigma)) - \Phi((s_{\text{odd}}, \sigma)) = \sum_{j=0}^{k} (-1)^j \binom{k}{j} \frac{1}{n-j} \overset{\text{Lemma 44}}{=} \frac{(-1)^k}{(n-k)\binom{n}{k}}.$$

From our choice of $n = 2^k$ and since $\log(n) \leq \sqrt{n}$ for large enough n, we obtain

$$\binom{n}{k} \geq \left(\frac{n}{k}\right)^k = \left(\frac{n}{\log(n)}\right)^{\log(n)} \geq n^{\frac{1}{2}\log(n)}.$$

Since the right hand side grows faster than any polynomial in n, the decrease in the potential can not be lower bounded by a polynomial in n.

For broadcast games we introduced the tree-dynamics where several (standard) improvement moves are subsumed in one tree-move such that every state in a tree-dynamics corresponds to a tree. It is not clear whether this tree-dynamics reaches an equilibrium fast. Observe that in broadcast games along a path in a tree the number of players using edges are all different. Thus, we can not use the example from above where we used the same number $\frac{1}{n-j}$ several times.

4.7.2 PLS-hardness for General Network Games

Since the potential tracks exactly the change in player costs, the task of finding an equilibrium can be formulated as a local search problem, where the neighborhood consists of all single player deviations. Recall the definition of the class PLS of *polynomial-time local search problems* as introduced by Johnson, Papadimitriou, and Yannakakis [JPY88] and explained in Section 2.4.

The task of finding an equilibrium in unweighted network games can be expressed as a problem in PLS. The feasible solutions are all strategy profiles in the game. The measure of a solution is the potential of the profile and the neighbors are all strategy profiles which differ in the strategy of a single player. We want to find a local minimum. Some feasible solution can be found by just choosing any of the strategies for every player. Once we have a strategy profile σ, we can compute the potential, since $n_\sigma(e)$ can be determined for every edge e. To decide if a profile σ is a local optimum (i.e., an equilibrium), we do a shortest-path computation for every source-sink pair of every player p in the graph with the following weights. For every edge $e \in \sigma_p$ we set the weight to $c_\sigma(e)$ and for any other edge $e' \in E \setminus \sigma_p$ the weight is $c_\sigma^{+1}(e')$. If the resulting shortest path has weight strictly smaller than $C_p(\sigma)$, then this path corresponds to an improving move. Otherwise, σ is an equilibrium. This shows that finding an equilibrium in a network game is a problem in PLS.

For directed networks, Syrgkanis [Syr10] shows that the problem is PLS-complete. His reduction was recently generalized by Bilò et al. [Bil+21] to work also for undirected graphs. We show that the same holds for uniform network games with sharing functions with economies of scale. Our reduction is a generalization of the reduction used in [Bil+21] which in turn is a generalization of the reduction of [Syr10]. We reduce from the PLS-complete problem Max Cut as introduced in Section 2.4.

The Intermediate Game Syrgkanis [Syr10] reduces Max Cut to an intermediate game. This game is the starting point of the reduction to undirected network games with fair cost allocation in [Bil+21]. This intermediate game is not a network game but a general cost sharing game. An instance of this game is given by an ordered set of n players $1, \dots, n$ and for every pair of players (i, j) where $i < j$ there are two resources $r^0_{i,j}$ and $r^1_{i,j}$. Further, we are given a weight $w_{i,j} \in \mathbb{R}_{\geq 0}$ and the cost functions of $r^0_{i,j}$ and $r^1_{i,j}$ are such that the difference of one and two player using the resource is exactly $w_{i,j}$, i.e., $c_{r^0_{i,j}}(1) - c_{r^0_{i,j}}(2) = w_{i,j}$ and the same holds for $r^1_{i,j}$. Every player $i \in [n]$ has two strategies

$$\sigma^0_i = \left\{r^0_{0,i}, \dots, r^0_{i-1,i}, r^1_{i,i+1}, \dots, r^1_{i,n}\right\} \text{ and } \sigma^1_i = \left\{r^1_{0,i}, \dots, r^1_{i-1,i}, r^0_{i,i+1}, \dots, r^0_{i,n}\right\}.$$

A player plays all resources associated to her, for the players before her she plays all resources of one of the sets r^0 or r^1, and for the players after her the resources of the other set. In this way a resource $r^*_{i,j}$ can only be used be the two players i and j, and they are both on the resource only if they play different strategies. That is for $i < j$ resource $r^0_{i,j}$ is used by both i and j, if and only if i plays σ^1_i and j plays σ^0_j. See Figure 4.32. Syrgkanis [Syr10] shows that finding an equilibrium in an intermediate game is PLS-complete, by a reduction from Max Cut.

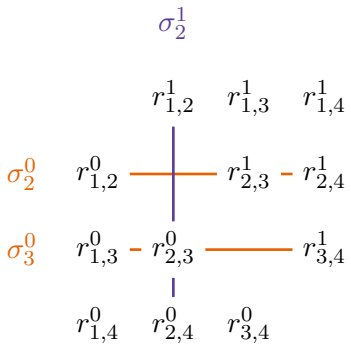


Figure 4.32: The strategies in an intermediate game with 4 players. Every player can either play its row (orange) or its column (purple). A resource is only used by two players, if and only if the corresponding players play different strategies. For example $r^0_{2,3}$ is used by two players, if and only if 2 plays σ^1_2 and 3 plays σ^0_3.

The Network Game We transform an instance of an intermediate game to a uniform undirected network game as shown in Figure 4.33. Note that this is the same graph as used in [Syr10] and [Bil+21].

The graph consists of (thin) edges for every resource $r^0_{i,j}$ and $r^1_{i,j}$ and some paths (thick edges) connecting the resource edges. For every player of the intermediate game there is a source and a sink node s_i and t_i. These nodes are connected by two *canonical paths* corresponding to the strategies σ^0_i and σ^1_i from the intermediate game. We chose the cost on the paths high enough, so that in an equilibrium every player is playing one of

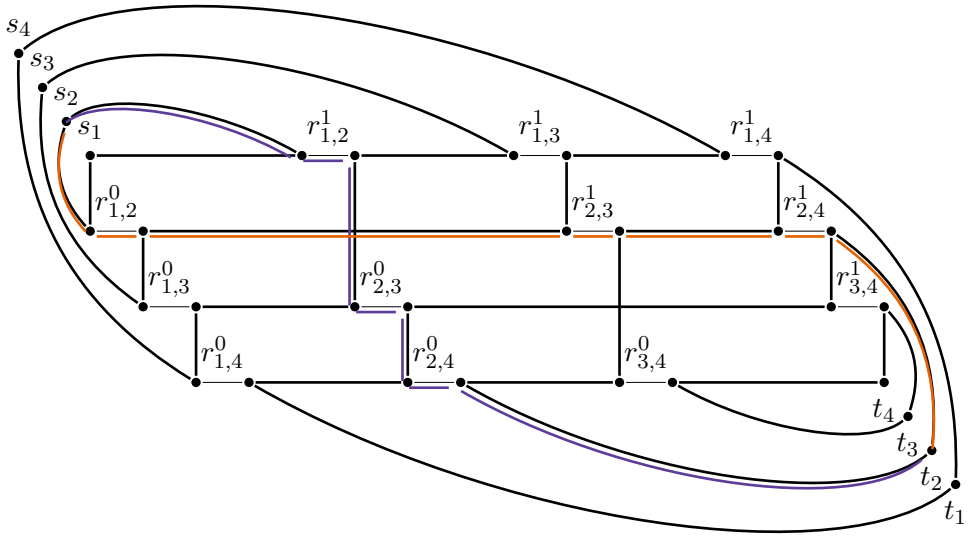


Figure 4.33: The network constructed from an instance of an intermediate game with 4 players. The canonical paths corresponding to the two strategies σ_2^0 (row) and σ_2^1 (column) are shown in orange and purple respectively. Thick edges represent paths of length L. The thin edges correspond to the resources of the intermediate game given as edge label.

its canonical paths. Observe from the graph that the canonical paths contain exactly n of the paths, while any other strategy contains strictly more. The main issue is to make the cost of the paths high regardless of how many players use it. For this we introduce some dummy players for each edge of a heavy path, who have their source and sink node at the two endpoints of the edge. Their canonical strategy is to just use the edge. If we have a lot of dummy players on an edge, then the cost for other players joining on this edge is almost constant. Think of fair cost allocation and consider the case where we have for example 100 players already using an edge. The cost incurred to the next few players is almost the same. Once we have that in an equilibrium every player plays their canonical strategy, the reduction is done since we immediately have the correspondence of strategies in the equilibrium and strategies in the intermediate game.

More formally, the graph to an instance of an intermediate game consists of *resource edges* for every resource $r^0_{i,j}$ and $r^1_{i,j}$. The scaling factor of a resource edge is set to $\frac{w_{i,j}}{f(1)-f(2)}$, where $w_{i,j}$ is the weight given in the intermediate game and f is the underlying function of our uniform network game. Note that for this choice the difference of one and two players using a resource edge is exactly $w_{i,j}$ just as in the intermediate game. Further, there are paths of length L in the network which will be called *heavy paths*. The structure of resource edges and heavy paths can be seen in Figure 4.33. We refer to an edge of a heavy path as a *heavy edge*. The scaling factor for a heavy edge is set to some $W \in \mathbb{R}_{\geq 0}$ which will be a lot larger than the other costs in the game. For every player in the intermediate game we have a player in the network game with the corresponding

source and sink nodes s_i and t_i as shown in Figure 4.33. Additionally, we have $D \in \mathbb{N}$ dummy players associated to every heavy edge. In total there are $n+2n^2DL$ players. We denote this game by $\mathcal{G}_f(L, D, W)$. The *canonical strategy* of a dummy player is playing the associated edge. For a player in the intermediate game, the canonical strategy is one of the paths corresponding to the respective row or column in the graph using exactly n heavy paths and $(n-1)$ resource edges (see Figure 4.33).

We will now show under which conditions on L, D and W, every player uses one of their canonical strategies in an equilibrium. That there are feasible values for the parameters will be discussed afterwards.

Lemma 34. *If $D \geq 1$ and $2D < L - 1$, then in any equilibrium in $\mathcal{G}_f(L, D, W)$ every dummy player plays her associated edge.*

Proof. Consider an equilibrium σ in $\mathcal{G}_f(L, D, W)$, where $D \geq 1$ and $2D < L-1$. Assume that there is some dummy player d associated to a heavy edge e but $\sigma_p \neq \{e\}$. Let P be the heavy path which contains e. First observe that in any equilibrium all dummy players associated to the same edge play the same strategy. If there are two players following different paths, then the cost of one of the paths upper bounds the cost of the other path. Since we are looking at decreasing per-player costs, switching to the cheaper (or same cost) strategy is an improving move for one of the players. Thus, we know that none of the D dummy players associated to edge e are using this edge. Note that these players use all other edges of the heavy path except edge e (see Figure 4.34). Let

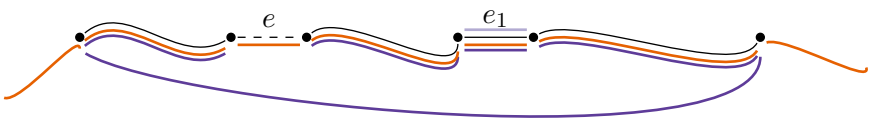


Figure 4.34: Drawing of a heavy path P where the dummy players associated to edge e do not follow their canonical strategy. They have to use all other edges on P as shown in purple. The players associated to e_1 play their canonical strategy (light purple). The orange path is a path of an external player to P. It has to use all edges on P.

$\bar{P}$ be those edges of P where the associated dummy players do not use their canonical strategy. From the above discussion this is well-defined as an edge is either used by all its associated players or by none of them. Denote by η the size of $\bar{P}$. Further let η_{ext} be the number of external players using path P. An external player is a player whose source and sink are not contained in P. Hence, if an external player uses any edge of P, she has to use the whole path to connect her source and sink node. With this notation we can bound the current cost of d by the cost incurred on path P. We consider the cost on the $\eta-1$ edges in $\bar{P}$ used by d and the remaining $L-\eta$ edges on P. On every edge in $\bar{P}$ there are the η_{ext} external players and all dummy players on P who do not play their canonical strategies except those associated with the edge. Thus we have a contribution of $(\eta-1)Wf(\eta_{\text{ext}} + D(\eta-1))$ to the cost of d from edges in $\bar{P}$. The remaining edges of P are used by all dummy players not playing their canonical strategy additionally to

the associated and the external players. Hence we can bound the current cost as

$$C_d(\sigma) \geq (L-\eta)Wf(\eta_{\text{ext}} + D(1+\eta)) + (\eta-1)Wf(\eta_{\text{ext}} + D(\eta-1)).$$

Now consider the strategy $s' = \{e\}$ for player d. The cost of this strategy is

$$C_d((s', \sigma_{-d})) = Wf(\eta_{\text{ext}} + D(\eta-1) + 1).$$

If $\eta \geq 2$, then $Wf(\eta_{\text{ext}} + D(\eta-1)+1)$ is smaller than $(\eta-1)Wf(\eta_{\text{ext}} + D(\eta-1))$ and since $L - \eta \geq 0$, s' would be an improving move for d. On the other hand, we bound the cost of s' for $\eta = 1$ as

$$\begin{aligned}
C_d((s', \sigma_{-d})) &= Wf(\eta_{\text{ext}} + 1) \\
&\leq W\frac{\eta_{\text{ext}} + 2D}{\eta_{\text{ext}} + 1} f(\eta_{\text{ext}} + 2D) && (kf(k) \text{ non-decr.}) \\
&\leq W2Df(\eta_{\text{ext}} + 2D) && (D \geq 1) \\
&< W(L-1)f(\eta_{\text{ext}} + 2D) && (\ 2D < L-1) \\
&\leq C_d(\sigma).
\end{aligned}$$

In both cases this is a contradiction to σ being an equilibrium. In total, we showed that every dummy player has to play her associated edge. □

Lemma 35. *If*

$$nf(D+1) < \left(n + \frac{1}{2}\right) f(D+n) \quad \textit{and} \quad W \geq n(n-1) \max_{1 \leq i < j \leq n} \frac{w_{i,j}}{f(1) - f(2)}$$

additionally to the conditions from Lemma 34, then in any equilibrium in $\mathcal{G}_f(L, D, W)$ every (non-dummy) player plays one of her canonical strategies.

Proof. Let σ be an equilibrium in $\mathcal{G}_f(L, D, W)$ where the conditions of the statement are satisfied. Assume that there is a (non-dummy) player i not playing one of the canonical strategies. From a previous discussion we know that she uses at least $n+1$ heavy paths and hence the current cost is at least

$$C_i(\sigma) \geq (n+1)LWf(D+n)$$

since from Lemma 34 we already have that every dummy player is on her associated edge in σ. On the other hand the cost of one of the canonical strategies for i is at most

$$nLWf(D+1) + (n-1)\max_{j \in [n]} \frac{w_{i,j}}{f(1) - f(2)} f(1),$$

where we set $w_{i,j} = w_{j,i}$ for $i > j$. With the first condition of this lemma we get the upper bound

$$(n+1)LWf(D+n) - \frac{1}{2}LWf(D+n) + (n-1)\max_{j \in [n]} \frac{w_{i,j}}{f(1) - f(2)} f(1).$$

From the conditions of Lemma 34 we have $L \geq 2D+2$ and since the total cost $kf(k)$ is non-decreasing we obtain

$$\frac{1}{2}LWf(D+n) \geq W\frac{L}{2(D+n)}f(1) \geq W\frac{2(D+1)}{2(D+n)}f(1) \geq W\frac{1}{n}f(1)$$

Together with the condition on W we get an upper bound on the cost of a canonical strategy of

$$(n+1)LWf(D+n) + f(1)\left((n-1)\max_{1\leq i<j\leq n}\frac{w_{i,j}}{f(1)-f(2)} - \frac{W}{n}\right) < C_i(\sigma).$$

This is a contradiction to σ being an equilibrium. □

We show that there are valid parameters L, W and D such that the conditions of Lemma 34 and Lemma 35 are satisfied. The main question is whether there exists D fulfilling $nf(D+1) < (n+\frac{1}{2})f(D+n)$. We show that for any sharing function f with economies of scale such a D exists and can be chosen to be polynomial in n.

Lemma 36. *For $n \geq 7$, there is a $D \in [2n^3]$ satisfying $nf(D+1) < \left(n+\frac{1}{2}\right)f(D+n)$ for any sharing function f with economies of scale.*

Proof. Assume the statement is false. Then for all $D \in [2n^3]$ we have $nf(D+1) \geq (n+\frac{1}{2})f(D+n)$. If we set $D = (\ell-1)(n-1)+1$ for some $\ell \in [n(2n+1)]$, then $D \leq 2n^3$ and

$$\begin{aligned} f(D+n) &= f((\ell-1)(n-1)+1+n) \\ &= f(\ell(n-1)+2) \leq \frac{n}{n+\frac{1}{2}}f((\ell-1)(n-1)+2) \end{aligned}$$

Chaining ℓ of these inequalities yields

$$f(\ell(n-1)+2) \leq \left(1-\frac{1}{2n+1}\right)^{\ell} f(2) \leq \left(1-\frac{1}{2n+1}\right)^{\ell} f(1).$$

Together with the non-decreasing total cost of f, we obtain

$$\begin{aligned} \frac{1}{2n^3}f(1) &\leq \frac{1}{n(2n+1)(n-1)+2}f(1) \\ &\leq f(n(2n+1)(n-1)+2) \leq \left(1-\frac{1}{2n+1}\right)^{n(2n+1)} f(1) \\ &\leq \mathrm{e}^{-n(2n+1)}f(1) \leq \mathrm{e}^{-n}f(1). \end{aligned}$$

Since the exponential function grows faster than any polynomial, this is a contradiction. In particular, for $n \geq 7$, we have $2n^3 < \mathrm{e}^n$. □

Since there are hard instances of Max Cut and hence instances of the intermediate game with many players, the assumption $n \geq 7$ is not a restriction for our reduction. Now that we have the existence of a feasible D, we choose the parameters as

$$D \in [2n^3] \quad \text{satisfying} \quad nf(D+1) < \left(n + \frac{1}{2}\right) f(D+n) \qquad \text{(from Lemma 36)}$$
$$L = 2(D+1) \quad \text{and} \quad W = n(n-1) \max_{1 \leq i < j \leq n} \frac{w_{i,j}}{f(1) - f(2)}. \tag{4.55}$$

These values satisfy the conditions of Lemma 34 ad Lemma 35. We are now ready to show the main theorem.

Theorem 11. *Computing an equilibrium in a uniform network game is PLS-complete.*

Proof. We reduce from an intermediate game of Syrgkanis [Syr10] with $n \geq 7$ players. Let $\mathcal{G}_f(L, D, W)$ be the game as described in the beginning of this section, where the parameters are chosen as in (from Lemma 36). Since the parameters and hence the graph are all polynomial in n, this specifies a feasible mapping $\varphi_{\text{instance}}$ of instances of an intermediate game to a network game.

Consider an equilibrium σ' in $\mathcal{G}_f(L, D, W)$. From Lemma 34 and Lemma 35 all players follow one of their canonical strategies in σ'. Define the profile σ in the intermediate game, where player i plays according to σ'_i. If σ'_i corresponds to the row-path, then $\sigma_i = \sigma_i^0$ and if σ'_i corresponds to the column-path we set $\sigma_i = \sigma_i^1$. From the previous lemmas this defines a mapping $\varphi_{\text{solution}}$ of local minima in $\mathcal{G}_f(L, D, W)$ to feasible solutions in the intermediate game. We show a relation of the player costs in $\mathcal{G}_f(L, D, W)$ and in the intermediate game. From this we will see that equilibria in $\mathcal{G}_f(L, D, W)$ are mapped to equilibria in the intermediate game. For a non-dummy player i we compute

$$\begin{aligned} C_i(\sigma') &= nLWf(D+1) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma'_i \cap \sigma'_j \neq \emptyset}} \frac{w_{i,j}}{f(1)-f(2)} f(2) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma'_i \cap \sigma'_j = \emptyset}} \frac{w_{i,j}}{f(1)-f(2)} f(1) \\ &= nLWf(D+1) + \sum_{j \in [n]\setminus\{i\}} \frac{w_{i,j}}{f(1)-f(2)} f(2) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma'_i \cap \sigma'_j = \emptyset}} \frac{w_{i,j}}{f(1)-f(2)} (f(1)-f(2)) \\ &= nLWf(D+1) + \sum_{j \in [n]\setminus\{i\}} \frac{w_{i,j}}{f(1)-f(2)} f(2) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma'_i \cap \sigma'_j = \emptyset}} w_{i,j}. \end{aligned}$$

The only term depending on the strategy chosen by i (and other players) is the last summand. For the profile σ in the intermediate game we have

$$\begin{aligned} C_i(\sigma) &= \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma_i \cap \sigma_j \neq \emptyset}} c_{r_{i,j}}(2) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma_i \cap \sigma_j = \emptyset}} c_{r_{i,j}}(1) \\ &= \sum_{j \in [n]\setminus\{i\}} c_{r_{i,j}}(2) + \sum_{\substack{j \in [n]\setminus\{i\} \\ \sigma_i \cap \sigma_j = \emptyset}} c_{r_{i,j}}(1) - c_{r_{i,j}}(2) \end{aligned}$$

$$= \sum_{j \in [n] \setminus \{i\}} c_{r_{i,j}}(2) + \sum_{\substack{j \in [n] \setminus \{i\} \\ \sigma_i \cap \sigma_j = \emptyset}} w_{i,j}.$$

Observe that the term depending on the strategy chosen by i is the same in both games. Hence, if σ' is an equilibrium in $\mathcal{G}_f(L, D, W)$, then σ is an equilibrium in the intermediate game, which completes the reduction. □

We remark here that our reduction follows closely the one from Bilò et al. [Bil+21]. We were able to keep the main property that every non-dummy player follows a canonical strategy in an equilibrium for sharing functions with economies of scale. These cost functions are already considered in [Syr10], but the main difference is that here we are looking at uniform games, whereas Syrgkanis uses different functions on the edges from a huge class of functions. So our reduction is a generalization of fair cost allocation from Bilò et al. [Bil+21] and a specialization of non-uniform games from Syrgkanis [Syr10].

The main open question is, whether the same result also holds for network games where all players have a common sink.

Research Question 9. Is the problem of finding an equilibrium in multi- or broadcast games PLS-complete?

This is still unsolved even for fair cost allocation. On the other hand, if we had allowed constant per-player cost functions finding an equilibrium in this case is easy. As there is no effect of sharing, every player just chooses a shortest path w.r.t. the scaling factors.

The above construction can not be adapted easily for multicast games. To make the cost on the heavy paths almost constant for the non-dummy players, we need to force the dummy players to use their edge. For this, we need that the source and sink node of a dummy player are exactly the endpoints of the associated edge. Further, we need to make sure that every player freely chooses one of her two canonical strategies. In multicast games any equilibrium forms a tree (Lemma 10). It seems rather difficult to model independent choices of paths in trees.

4.7.3 NP-Hardness Results for Multi- and Broadcast Games

We are now looking at finding equilibria in a centralized way. This question has been studied for fair cost allocation. Already Anshelevich et al. [Ans+08a] shows that computing the best equilibrium is NP-hard in directed multicast games. The same result for unidirected multicast games is implicitly given in [Syr10]. Syrgkanis shows that singleton cost sharing games are special cases of multicast games (directed and undirected) and remarks that the proof from Chekuri et al. [Che+06] showing that computing the global potential minimizer for singleton cost sharing games is NP-hard, can be used to show NP-hardness of computing the best equilibrium. Syrgkanis elaborates on a relation between minimizing the social cost in singleton cost sharing games and set cover.

We use the translation of a set cover instance to a multicast game as in [Che+06] and [Syr10]. We show how to use this for sharing functions with economies of scale. With a small adaption, this game can also be used to obtain results for broadcast games. Instead of reducing from (general) Set Cover we reduce from Exact Cover by 3-Sets

(X3C). Hence our reduction can also be viewed as a generalization of the reduction from 3D-Matching by Anshelevich et al. [Ans+08a].

Network Game from X3C Recall the Exact-Cover by 3-Sets (X3C) problem as introduced in Section 2.4. An instance is given by a ground set X and a family $\mathcal{S}$ of 3-element subsets of X. The question is, whether there exists an exact cover of X, i.e., is there a $\mathcal{C} \subseteq \mathcal{S}$ such that every element of X appears in exactly one of the sets in $\mathcal{C}$? We call a collection $\mathcal{C} \subseteq \mathcal{S}$ a *cover*, if $X \subseteq \bigcup_{C \in \mathcal{C}} C$ and it is called a *packing*, if the sets in $\mathcal{C}$ are pairwise disjoint. An exact cover is both a cover and a packing.

From an instance $(X, \mathcal{S})$ of X3C we build a graph as shown in Figure 4.35. There are nodes for every element of X and every element of $\mathcal{S}$. Whenever we have $x \in S$ there is an edge $\{x, S\}$. Additionally there is a root node r and every element of $\mathcal{S}$ is connected to r. We denote this graph by $G_{(X,\mathcal{S})}$. For the case of computing a best equilibrium in broadcast games we also add edges $\{x, r\}$ for every element of X (the dashed edges in Figure 4.35). This graph will be denoted by $\bar{G}_{(X,\mathcal{S})}$.

For multicast games we have players only in elements of X, whereas in broadcast games we also have players in elements of $\mathcal{S}$.

We consider uniform games, where the edges $\{S, r\}$ have scaling factor 1 and edges $\{x, S\}$ have factor Γ. This Γ is the same for all edges $\{x, S\}$ and is set to different values in the respective reductions. In $\bar{G}_{(X,\mathcal{S})}$ the edges $\{x, r\}$ have scaling factor $\bar{\Gamma}$.

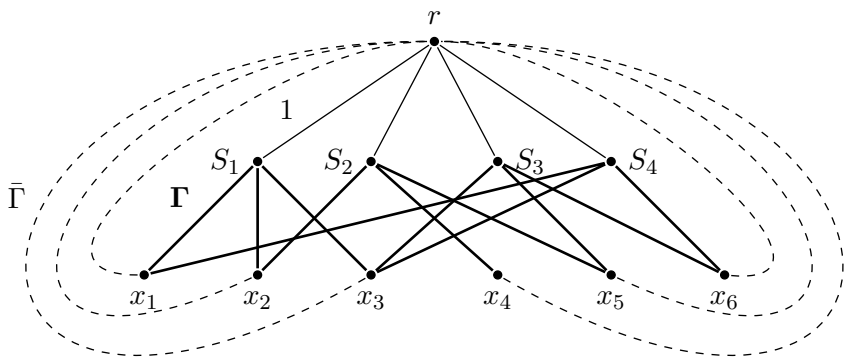


Figure 4.35: The graph $\bar{G}_{(X,\mathcal{S})}$ for an X3C instance $(X, \mathcal{S})$ where $X = \{x_1, \ldots, x_6\}$ and $\mathcal{S} = \{S_1, \ldots, S_4\}$. Without the dashed edges we get $G_{(X,\mathcal{S})}$. Thin edges going from S to r have $\gamma_{\{S,r\}} = 1$, bold edges have $\gamma_{\{u,S\}} = \Gamma$, and the dashed edges have $\gamma_{\{x,r\}} = \bar{\Gamma}$.

We define the games $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ and $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ where the underlying graphs are $G_{(X,\mathcal{S})}$ and $\bar{G}_{(X,\mathcal{S})}$ respectively. The parameter $\beta \in \{0, 1\}$ denotes whether we are in a multicast ($\beta = 0$) or in a broadcast ($\beta = 1$) game. Formally we have the broadcast game $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 1\right) = \left(\bar{G}_{(X,\mathcal{S})}, r, f, \gamma\right)$, where $\gamma_{\{S,r\}} = 1, \gamma_{\{x,S\}} = \Gamma$, and $\gamma_{\{x,r\}} = \bar{\Gamma}$. Whereas $\left(\mathcal{G}_{(X,\mathcal{S})}, 0\right)$ describes the multicast game $(G_{(X,\mathcal{S})}, r, X, f, \gamma)$, where γ is as above.

We show a relation between equilibria in $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ and $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$, and exact covers of $(X, \mathcal{S})$ depending on the choices of Γ and $\bar{\Gamma}$. Intuitively, we define for an exact cover $\mathcal{C}$, a profile, where every $S \in \mathcal{S}$ goes directly to the root and every $x \in X$ goes via the unique set in $\mathcal{C}$ to the root. On the other hand for any equilibrium, we consider a set $S \in \mathcal{S}$ to be in the cover, if some x uses S to connect to r.

For an exact cover $\mathcal{C}$ denote by $C(x)$ the unique element of $\mathcal{C}$ containing x. We define the profile $\sigma^{\mathcal{C}}$, where

$$\sigma^{\mathcal{C}}_S = \{S, r\} \quad \text{and} \quad \sigma^{\mathcal{C}}_x = \{\{x, C(x)\}, \{C(x), r\}\}$$

for players in $\mathcal{S}$ and X respectively. Under some conditions on Γ and $\bar{\Gamma}$ these profiles are equilibria.

Lemma 37. *For an exact cover $\mathcal{C}$ of $(X, \mathcal{S})$, the profile $\sigma^{\mathcal{C}}$ is*

- *an equilibrium in $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$, if $\Gamma \geq 1$*
- *an equilibrium in $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$, if $\Gamma \geq 1$ and $f(1)\bar{\Gamma} \geq f(1)\Gamma + f(3+\beta)$.*

Further, $\sigma^{\mathcal{C}}$ satisfies

$$n_{\sigma^{\mathcal{C}}}(\{C, r\}) = 3 + \beta \text{ for } C \in \mathcal{C} \quad \text{and} \quad n_{\sigma^{\mathcal{C}}}(\{S, r\}) = \beta \text{ for } S \in \mathcal{S} \setminus \mathcal{C}.$$

Proof. Let $\mathcal{C}$ be an exact cover of $(X, \mathcal{S})$. The numbers of players using edges $\{S, r\}$ follow immediately from the fact that the sets in $\mathcal{C}$ are disjoint and cover all elements of X.

Now let $\Gamma \geq 1$. Consider a player $S \in \mathcal{S}$ (if $\beta = 1$). The current cost of S is

$$C_S\left(\sigma^{\mathcal{C}}\right) = f(3+\beta).$$

Any other strategy begins with an edge $\{S, x\}$ followed by an edge $\{x, S'\}$. From $x \in S$ and the discussion above we get that $\{S, x\}$ is used by x and $\{x, S'\}$ is not used in $\sigma^{\mathcal{C}}$. Hence, the cost of any other strategy for player S is lower bounded by $\Gamma f(2) + \Gamma f(1)$. Since f is decreasing and $\Gamma \geq 1$, there are no improving moves for S.

For a player $x \in X$ we compute the current cost

$$C_x\left(\sigma^{\mathcal{C}}\right) = \Gamma f(1) + f(3+\beta).$$

If we are in $G_{(X,\mathcal{S})}$, then any other strategy begins with an edge of cost $\Gamma f(1)$ and the next edge is used by at most 2 players and may even have the scaling factor Γ. Hence, there is no improving move for x. If we are in $\bar{G}_{(X,\mathcal{S})}$, the direct edge to r is an available strategy of cost $f(1)\bar{\Gamma}$. From the condition on $\bar{\Gamma}$ together with the arguments above, there are no improving moves for x. □

For the other direction, consider an equilibrium σ in $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ or $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ and define the set

$$\mathcal{S}^{\sigma} = \{S \in \mathcal{S} : n_{\sigma}(\{S, r\}) > \beta\}.$$

We will show that in a game $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ the collection $\mathcal{S}^\sigma$ is a cover, and in $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ it is a packing of $(X, \mathcal{S})$. For this we need an auxiliary lemma showing that only the elements of a set S are using S on their path to the root.

Lemma 38.
An equilibrium τ of $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ where $\Gamma \geq 1$ and $f(1) \leq f(2+\beta)\Gamma$ satisfies

$$\forall S \in \mathcal{S} : \tau_S = \{S, r\} \quad \text{and}$$
$$\forall x \in X : \tau_x = \{\{x, S\}, \{S, r\}\} \text{ for some } S \in \mathcal{S} \text{ with } x \in S.$$

An equilibrium τ of $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ where $\Gamma \geq 1$ and $f(1)\bar{\Gamma} \leq f(1)\Gamma + f(2+\beta)\Gamma$ satisfies

$$\forall S \in \mathcal{S} : \tau_S = \{S, r\} \quad \text{and}$$
$$\forall x \in X : \tau_x = \{x, r\} \text{ or } \tau_x = \{\{x, S\}, \{S, r\}\} \text{ for some } S \in \mathcal{S} \text{ with } x \in S.$$

Proof. Let τ be an equilibrium of $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ or $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ and let $T = \text{supp}(\tau)$ be the corresponding tree from Lemma 9.

We show that any strategy consists of at most 2 edges if the corresponding properties of Γ and $\bar{\Gamma}$ are satisfied. From the structure of $G_{(X,\mathcal{S})}$ and $\bar{G}_{(X,\mathcal{S})}$ the given form of the strategies follows.

Let v be a player with a longest path in τ, i.e., v maximizes $|\tau_v|$. Then v is a leaf in T. Assume for a contradiction that $|\tau_v| > 2$.

If $v \in \mathcal{S}$ (if $\beta = 1$), then the current cost of v is strictly greater than $f(1)\Gamma$. But going directly to the root would incur a cost of at most $f(1)$. Since $\Gamma \geq 1$, this shows that v would have an improving move, contradicting τ being an equilibrium.

So we have $v \in X$. Since v uses more than 2 edges the strategy is of the form $\tau_v = \{\{v, S\}, \{S, x\}\} \cup \tau_x$. Since v is a leaf in T she uses the first edge on her own. The edge $\{S, x\}$ is used by all descendants of S in T. As v has a longest path, these can only be direct neighbors of S (excluding r). Hence this edge is used by at most $2 + \beta$ players: player v, the third element of S unequal to v and x, and possibly S. Since the path τ_x incurs some positive cost the current cost of v is lower bounded by

$$C_v(\tau) > f(1)\Gamma + f(2+\beta)\Gamma.$$

We are now looking at two alternative strategies for v. In $G_{(X,\mathcal{S})}$ consider the strategy $t' = \{\{v, S\}, \{S, r\}\}$. This strategy incurs a cost of at most $f(1)\Gamma + f(1)$. Since $f(1) \leq f(2+\beta)\Gamma$, this would be an improving move for v. In $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ consider $t' = \{v, r\}$ of cost $f(1)\bar{\Gamma}$. Again, we get a contradiction from the condition on $\bar{\Gamma}$. □

We are now able to show that equilibria correspond to covers or packings. For the game $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ this follows immediately from the previous lemma.

Corollary 5. *For an equilibrium σ in a game $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$ with the conditions of Lemma 38, the set $\mathcal{S}^\sigma$ is a cover of $(X, \mathcal{S})$. Further, $\mathcal{S}^\sigma$ is an exact cover, if and only if $n_\sigma(\{S, r\}) \in \{1, 3+\beta\}$ for every $S \in \mathcal{S}$.*

Lemma 39. *For an equilibrium σ in a game $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$ where $f(1)\bar{\Gamma} \leq f(1)\Gamma + f(3+\beta)$, the set $\mathcal{S}^\sigma$ is a packing of $(X, \mathcal{S})$.*

Proof. First, observe that the condition on $\bar{\Gamma}$ given here implies the condition of Lemma 38 since f is decreasing. Hence, we know that the structure of the strategies in σ is as given in Lemma 38. We show that for any set $S \in \mathcal{S}$ either all of the three elements of S go via S to r or none of them does. This yields that the elements of $\mathcal{S}^\sigma$ are disjoint. Consider a set $S \in \mathcal{S}$ and a player $x \in X$ visiting S on its path. From Lemma 38 we have $\sigma_x = \{\{x, S\}, \{S, r\}\}$. Further, x is a leaf in $\text{supp}(\sigma)$ and the edge $\{S, r\}$ can be used by at most $3 + \beta$ players. Hence, the current cost of x is lower bounded by

$$C_x(\sigma) \geq f(1)\Gamma + f(3 + \beta).$$

Going directly to the root is a feasible strategy of x, which incurs a cost of

$$C_x((\{x, r\}, \sigma_{-x})) = f(1)\bar{\Gamma} \leq f(1)\Gamma + f(3 + \beta).$$

Since σ is an equilibrium, switching to $\{x, r\}$ can not be an improving move.

If $f(1)\bar{\Gamma} < f(1)\Gamma + f(3+\beta)$, all players use their direct edge and $\mathcal{S}^\sigma$ is a packing since it is empty. Now let $f(1)\bar{\Gamma} = f(1)\Gamma + f(3 + \beta)$, then the current cost of x has to be exactly $f(1)\Gamma + f(3 + \beta)$. This can only be the case, if all of the $3 + \beta$ available players use edge $\{S, r\}$. Meaning that if some player x goes via S in an equilibrium, then the other 2 elements have to join. This shows that the sets in $\mathcal{S}^\sigma$ are disjoint. □

We use these relations of equilibria and covers and packings to show that computing best equilibria is NP-hard.

Best Equilibria in Multicast Games We consider the multicast game $\left(\mathcal{G}_{(X,\mathcal{S})}, 0\right)$, where we set $\Gamma = \frac{f(1)}{f(2)}$. For this choice we have $\Gamma \geq 1$ and $f(1) \leq f(2)\Gamma$. Hence we can use Lemma 37 and Corollary 5.

Theorem 12. *It is NP-hard to find a best equilibrium in a uniform multicast game.*

Proof. We reduce from X3C. Let $(X, \mathcal{S})$ be an instance of X3C and consider the game $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 0\right)$ as described before. First assume that there is no exact cover of X in $\mathcal{S}$. Let σ be any equilibrium of $\left(\mathcal{G}_{(X,\mathcal{S})}, 0\right)$ and $\mathcal{S}^\sigma$ be the corresponding cover (Corollary 5). We show a relation of the social cost of σ and the size of $\mathcal{S}^\sigma$. For $i \in \{1, 2, 3\}$ define the set $\mathcal{S}_i^\sigma$ containing the sets C from $\mathcal{S}^\sigma$ where i elements go via C, i.e., $n_\sigma(\{C, r\}) = i$. From Lemma 38 we have $n_\sigma(\{C, r\}) \leq 2$ for any $C \in \mathcal{S} \setminus \mathcal{S}_3^\sigma$. The social cost can be bounded as

$$\begin{aligned} C(\sigma) &= \sum_{\substack{e=\{u,S\} \\ \in \text{supp}(\sigma)}} \gamma_e n_\sigma(e) f(n_\sigma(e)) \quad + \sum_{\substack{e=\{S,r\} \\ \in \text{supp}(\sigma)}} \gamma_e n_\sigma(e) f(n_\sigma(e)) \\ &\geq |U| f(1) \frac{f(1)}{f(2)} \quad + \sum_{S \in \mathcal{S}_3^\sigma} 3f(3) + \sum_{S \in \mathcal{S} \setminus \mathcal{S}_3^\sigma} n_\sigma(\{S, r\}) f(2). \end{aligned} \tag{4.56}$$

Since every player has to connect to the root we have $|U| = \sum_{S \in \mathcal{S}} n_\sigma(\{S, r\})$. We can thus rewrite the right hand side of (4.56) to

$$|U|\frac{f(1)^2}{f(2)} + |\mathcal{S}_3^\sigma|3f(3) + (|U| - 3|\mathcal{S}_3^\sigma|)f(2) = |U|\left(\frac{f(1)^2}{f(2)} + f(2)\right) + 3|\mathcal{S}_3^\sigma|(f(3) - f(2))$$

The number of players $|U|$ can also be expressed as $|\mathcal{S}^\sigma| + |\mathcal{S}_2^\sigma| + 2|\mathcal{S}_3^\sigma|$ and hence we have

$$2|\mathcal{S}_3^\sigma| \leq |U| - |\mathcal{S}^\sigma|. \tag{4.57}$$

We continue (4.56) as

$$\begin{aligned} C(\sigma) &\geq |U|\left(\frac{f(1)^2}{f(2)} + f(2)\right) - \frac{3}{2}(|U| - |\mathcal{S}^\sigma|)(f(2) - f(3)) \\ &= |U|\left(\frac{f(1)^2}{f(2)} - \frac{1}{2}f(2) + \frac{3}{2}f(3)\right) + \frac{3}{2}|\mathcal{S}^\sigma|(f(2) - f(3)). \end{aligned} \tag{4.58}$$

Since there is no exact cover for $(X, \mathcal{S})$ the inequalities (4.57) and hence (4.58) are strict for $\mathcal{S}^\sigma$.

On the other hand, if $\mathcal{C}$ is an exact cover for $(X, \mathcal{S})$, there is an equilibrium $\sigma^{\mathcal{C}}$ from Lemma 37. In the computations above for the social cost of $\sigma^{\mathcal{C}}$ the inequalities (4.56), (4.57) and (4.58) are satisfied with equality, since $\mathcal{C} = \mathcal{S}^{\sigma^{\mathcal{C}}}$ is an exact cover of U.

Assume we can compute a best equilibrium in $\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right)$. Comparing the social cost of the equilibrium to the right hand side of (4.58), where $|\mathcal{S}^\sigma| = \frac{|U|}{3}$, decides X3C. □

Best Equilibria in Broadcast Games For broadcast games consider the game $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 1\right)$ with the dashed edges connecting every element x directly to the root. We set $\bar{\Gamma} = 1 + \frac{f(4)}{f(1)}$ and $\Gamma = 1$. The conditions of Lemma 37 and Lemma 39 are satisfied as $\Gamma \geq 1$ and $f(1)\bar{\Gamma} = f(1)\Gamma + f(4)$.

For multicast games any equilibrium corresponds to a cover and by minimizing the social cost we tried to make the elements of the cover disjoint and hence an exact cover. Here we go in the other direction. We know that any equilibrium corresponds to a packing and minimizing the social cost leads to a packing covering all elements and hence an exact cover. For this, we need the information whether some elements are covered by a packing or not. This is why we can not use the same graph as for multicast games. Consider the graph without the dashed edges in the case of fair cost allocation. Then every profile has the same social cost, since every S connects to the root and every x to some S. We thus loose the information which S is actually used to cover some x. By adding the outer strategies for players in x we regain the information that some x is not covered.

Using the correspondence of equilibria and packings we show the main statement.

Theorem 13. *It is NP-hard to find a best equilibrium in a uniform broadcast game.*

Proof. We reduce from X3C. Let $(X, \mathcal{S})$ be an instance of X3C and consider the game $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 1\right)$ as described above. Assume there is no exact cover in $(X, \mathcal{S})$. Let τ be any equilibrium of $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 1\right)$ and $\mathcal{S}^\tau$ be the corresponding packing (Lemma 39). From Lemma 38 we can compute the social cost of τ as

$$C(\tau) = f(1)|\mathcal{S} \setminus \mathcal{S}^\tau| + f(4)|\mathcal{S}^\tau| + (f(1) + f(4))|U|.$$

We have $|\mathcal{S} \setminus \mathcal{S}^\tau| = |\mathcal{S}| - |\mathcal{S}^\tau|$ and since $\mathcal{S}^\tau$ is a packing also $|\mathcal{S}^\tau| \leq \frac{|U|}{3}$. Thus

$$\begin{aligned} C(\tau) &= f(1)|\mathcal{S}| + (f(4) - f(1))|\mathcal{S}^\tau| + (f(1) + f(4))|U| \\ &= f(1)|\mathcal{S}| + (f(1) - f(4))(|U| - |\mathcal{S}^\tau|) + 2f(4)|U| \\ &\geq f(1)|\mathcal{S}| + (f(1) - f(4))\frac{2}{3}|U| + 2f(4)|U| \qquad (4.59) \\ &= f(1)|\mathcal{S}| + |U|\left(\frac{2}{3}f(1) + \frac{4}{3}f(4)\right) \end{aligned}$$

Since $\mathcal{S}^\tau$ is not an exact cover, (4.59) is strict.

On the other hand, if there is an exact cover $\mathcal{C}$ for $(X, \mathcal{S})$, there is an equilibrium $\sigma^{\mathcal{C}}$ from Lemma 37 with $\mathcal{C} = \mathcal{S}^{\sigma^{\mathcal{C}}}$. In the above computation for the social cost of $\sigma^{\mathcal{C}}$, inequality (4.59) is satisfied with equality.

Thus, as before, if we can compute a best equilibrium in $\left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, 1\right)$, we can decide X3C by comparing the social cost to the final quantity in (4.59). □

The Global Potential Minimizer in Broadcast Games Instead of trying to find the best equilibrium, one can also try to find another special equilibrium, the global potential minimizer. Computing the global potential minimizer in a multicast game corresponds to finding a minimum cost flow with concave edge costs. For strictly concave edge costs, Guisewite and Pardalos [GP91a] show that finding a minimum cost flow is NP-hard in the directed multicast case. Our potential function induces strictly concave edge costs and hence falls into their class. We show that the problem stays NP-hard for undirected broadcast games.

We consider the game $\left(\mathcal{G}_{(X,\mathcal{S})}, 1\right)$, where we set Γ to $\frac{f(1)}{f(3)}$. With this choice we have $\Gamma \geq 1$ and $f(1) \leq f(3)\Gamma$ and hence we can apply Lemma 37 and Corollary 5.

Note that for fair cost allocation, in contrast to the social cost, the number of players using an edge influences the potential and hence we do not need the outer dashed edges in this case.

Theorem 14. *It is NP-hard to find a global potential minimizer in a uniform broadcast game.*

Proof. We reduce from X3C. Let $(X, \mathcal{S})$ be an instance of X3C and consider the game $\left(\mathcal{G}_{(X,\mathcal{S})}, 1\right)$ as described above. Assume there is no exact cover of $(X, \mathcal{S})$. Let σ be any equilibrium of $\left(\mathcal{G}_{(X,\mathcal{S})}, 1\right)$. With the structure of σ from Lemma 38 we compute the

potential as

$$\Phi(\sigma) = \sum_{S \in \mathcal{S}} F(n_\sigma(\{S, r\})) + \Gamma F(1)|U|.$$

Since F is concave and $1 \leq n_\sigma(\{S, r\}) \leq 4$, we can bound the potential by

$$\begin{aligned} \Phi(\sigma) &\geq \sum_{S \in \mathcal{S}} \left(\frac{4 - n_\sigma(\{S, r\})}{3} F(1) + \frac{n_\sigma(\{S, r\}) - 1}{3} F(4) \right) + \Gamma f(1)|U| \qquad (4.60) \\ &= \sum_{S \in \mathcal{S}} \left(3F(1) + (n_\sigma(\{S, r\}) - 1) \frac{F(4) - F(1)}{3} \right) \quad + \Gamma f(1)|U| \end{aligned}$$

Since $\sum_{S \in \mathcal{S}} n_\sigma(\{S, r\}) = |U| + |\mathcal{S}|$ we can write the right hand side as

$$\begin{aligned} &3|\mathcal{S}|F(1) + \left(\sum_{S \in \mathcal{S}} n_\sigma(\{S, r\}) - |S| \right) \frac{F(4) - F(1)}{3} + \Gamma f(1)|U| \\ =&3|S|F(1) + |U| \left(\frac{F(4) - F(1)}{3} + \Gamma f(1) \right) \end{aligned}$$

As F is strictly concave, (4.60) is tight only if $n_\sigma(\{S, r\}) \in \{1, 4\}$ for every $S \in \mathcal{S}$. From Corollary 5, we have that (4.60) is strict since there is no exact cover.

On the other hand, if there is an exact cover $\mathcal{C}$ of $(X, \mathcal{S})$, there is an equilibrium $\sigma^{\mathcal{C}}$ from Lemma 37 where $n_\sigma(\{S, r\}) \in \{1, 4\}$ for every $S \in \mathcal{S}$. In the above computation of the potential of $\sigma^{\mathcal{C}}$ the inequality is satisfied with equality.

Thus, if we can compute a global potential minimizer, we can decide X3C by comparing the potential to the final quantity in the above computation. □

Note that the social cost of the global potential minimizer can be far away from the cost of a social optimum as studied in [KM13].

4.8 Conclusion

In this chapter, we give the first detailed study of the efficiency of equilibria in uniform network cost sharing games with general concave non-decreasing total costs. We determine the Price of Anarchy for uniform games and give bounds on the Price of Stability for general and broadcast games. Finally, we show hardness results for the computation of (good) equilibria.

Upper Bounds on the Price of Stability. For general games, we show a constant upper bound for linear and polynomial total cost functions. This constant depends on the parameter s or α, and as the parameter gets close to 0, the bound grows very large. On the other hand, the case where s, α are 0 is fair cost allocation for which a constant bound is already known. Thus, the main direction of future research is to find better upper bounds on the PoS for small parameters s and α, which are closer to the known bounds for fair cost allocation.

For broadcast games, we extend the homogenization-absorption framework from Bilò, Flammini, and Moscardelli [BFM20] for fair cost allocation to sharing functions with

economies of scale. This gives constant upper bounds for broadcast games where the cost function decreases quickly. In our analysis, we did not use any properties specific to a social optimum. We did not even use the exact social cost of an optimum but just a lower bound. Finding ways to incorporate the above aspects into the analysis is a promising direction to reduce the bound (Research Question 4).

Lower Bounds on the Price of Stability. We construct instances of broadcast games to obtain lower bounds for sharing functions with economies of scale. For small graphs, our computational experiments show that these instances give the highest values among graphs with the same number of nodes. However, we do not have any theoretical tools to prove this formally. There is a significant lack in the understanding of instances where the best equilibrium is far away from a social optimum. It is, for example, not clear what graphs we have to consider. Do we need to take the complete graph, or can we restrict the instances to smaller subclasses (Research Question 5)? Intuitively, due to economies of scale effects, the cost of an equilibrium is high if only few players share edges. Hence, another point of attack is to show that the PoS is defined by an instance where a star is one of the best equilibria (compare to Research Question 7). Developing compact characterizations of a profile being one of the best equilibria would also benefit the computational experiments. So far, all lower bounds are below 2. Hence, we ask

Research Question 10. Is the PoS of broadcast games at most 2?

Computing (good) Equilibria. We extend the PLS-hardness proof for finding an equilibrium in general network games of Bilò et al. [Bil+21] to sharing functions with economies of scale and show that computing the best equilibrium is NP-hard even in broadcast games for sharing functions with economies of scale. The main open question is whether the PLS-hardness also holds for multicast games (Research Question 9). On the one hand, existing constructions in PLS-hardness proofs do not seem to transfer to the single-sink case. The tree structure of equilibria is very restrictive in modeling independent decisions. Does this prevent PLS-reductions for the problem? On the other hand, while trying to find a local search algorithm, we came up with the example in Section 4.7.1 showing that there are local moves where the progress is very little. We were not able to extend this to a complete sequence of improving moves taking exponentially many steps. Apart from local search algorithms, there may exist centralized algorithms to find an equilibrium.

5 The Undirected Two Disjoint Shortest Paths Problem

5.1 Introduction

In the previous two chapters, we looked at two types of effects appearing when sharing common resources. In both cases, using the same resource is allowed but has a different impact on the players. In the first setting, sharing was disadvantageous, and in the second setting, sharing was beneficial for the players. In this chapter, we are interested in the case where using the same resource is disallowed.

Consider a factory for filled chocolates. In one of the buildings, the chocolates get their outer chocolate cover. The company produces two types of chocolates: One with cassis filling and white chocolate cover and one with orange filling covered with dark chocolate. There are some conveyor belts installed that can be run in both directions interconnected by junctions. The plain chocolates arrive at two places from another building, and the melted white and dark chocolate arrive at two other places. The producer has to find two ways of activating some of the conveyor belts to bring the coverings to their plain chocolates. He must make sure that the two assembly lines he chooses do not cross in any of the junctions. Otherwise, the white and dark chocolate get mixed and can not be used anymore. Further, the melted chocolate has to be used very quickly as otherwise, it gets too cold. Thus the assembly lines have to be as short as possible. Figure 5.1 shows an example of the situation.

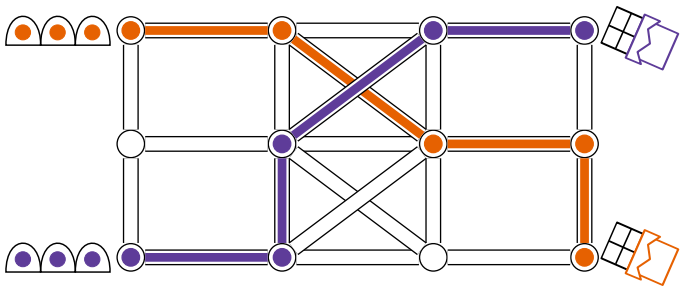


Figure 5.1: Example of conveyor belts and junctions in the cover building. The arrival points of the plain chocolates (left) and the melted cover chocolate (right) are given. All conveyor belts have length 1. Two feasible assembly lines are highlighted. They are as short as possible and do not meet at any junction.

In the above setting, the shared resources are the junctions, and the task is to find two *node disjoint shortest paths in an undirected graph*. An instance of the problem is given by an undirected graph with edge lengths and two source-sink node pairs. The question is whether there exist two paths connecting the sources to their corresponding sinks, such that each path is a shortest source-sink path in the graph, and they do not use any nodes in common.

This type of problem often arises in practice, and there are many versions. The natural generalization of the above problem to k source-sink pairs is called k disjoint shortest paths problem (k-DSPP). If we drop the condition that the paths have to be shortest paths, the problem is called the k disjoint paths problem (k-DPP). Both problems can be studied in undirected graphs and in directed graphs, where the task is to find directed paths. There are two versions of each problem. The paths can either considered to be node disjoint or edge disjoint. Finally, different types can be specified by restricting the allowed lengths of edges. If all lengths are 0, then every path is a shortest path, and hence k-DSPP and k-DPP are the same problem. In all other cases, these two problems are different. We mention two of the cases of interest. Instances, where every length is strictly positive, are distinguished from instances where the lengths are restricted to be non-negative, i.e., both edges of length 0 and positive length are allowed.

We study the node disjoint version of the undirected 2 disjoint shortest paths problem (2-DSPP) with non-negative edge lengths.

Note on Collaboration This chapter is based on joint work with Marinus Gottschau and Marcus Kaiser. The results are published in *Operations Research Letters*, Volume 47, Issue 1, 2019 [GKW19] and presented here in rewritten form.

Previous Work

Disjoint Paths. The disjoint paths problem is an old problem in graph theory and has received much attention. The first results show that the problem is NP-complete if the number k of disjoint paths is part of the input. On undirected graphs, it is one of Karp's first NP-complete problems [Kar72]. Even, Itai, and Shamir [EIS76] show that it is also NP-complete for directed graphs and even the restriction to planar graphs remains hard as shown by Lynch [Lyn75].

Hence, most papers study k-DPP for the case where k is *fixed* and not part of the input. Unfortunately, Fortune, Hopcroft, and Wyllie [FHW80] show that the problem remains NP-complete for any fixed $k \geq 2$ in general *directed* graphs. For special classes of directed graphs, polynomial-time algorithms are known. Shiloach and Perl [SP78] give the first efficient algorithm for the 2 disjoint paths problem (2-DPP) in directed *acyclic* graphs. Their algorithm was later improved by Tholey [Tho05] and generalized to k disjoint paths by Fortune, Hopcroft, and Wyllie [FHW80]. For *acyclic mixed* graphs, that contain undirected edges and directed arcs such that no orientation of the edges induces a directed cycle, Zhang and Nagamochi [ZN12] give a polynomial-time algorithm for the node disjoint version of k-DPP. Schrijver [Sch94] gives a polynomial-time algorithm for the same problem on *planar* directed graphs.

For *undirected* graphs, the first results are efficient algorithms for 2-DPP found independently by Seymour [Sey80], Shiloach [Shi80], Thomassen [Tho80], and Ohtsuki [Oht80]. Similarly to directed graphs, an efficient algorithm for planar graphs was developed by Reed, Robertson, Schrijver, and Seymour [Ree+91]. For the general k-DPP where k is fixed and not part of the input, Robertson and Seymour [RS85; RS95] give an $\mathcal{O}(n^3)$ algorithm (where n is the number of nodes) as part of their influential graph minors project. Kawarabayashi, Kobayashi, and Reed [KKR12] improved their algorithm to run in quadratic time. We summarize the known complexity results for k-DPP in Table 5.1.

	undirected	directed
k arbitrary	NP-complete [Kar72]	NP-complete [EIS76]
k fix	P [RS85]	NP-complete [FHW80]
$k = 2$	P [Sey80; Shi80; Tho80; Oht80]	NP-complete [FHW80]

Table 5.1: Complexity of k-DPP for $k \geq 2$.

Disjoint Shortest Paths. The problem we study in this chapter was introduced by Eilam-Tzoreff [Eil98]. She shows that k-DSPP is NP-complete if k is part of the input in both directed and undirected graphs, for the node and edge disjoint version, even for planar graphs where all edges have unit length. For *directed* graphs with non-negative edge lengths, the problem remains NP-complete for every fixed $k \geq 2$, as shown by Bérczi and Kobayashi [BK17]. On the positive side, Bérczi and Kobayashi give polynomial-time algorithms for 2-DSPP in directed graphs with positive edge lengths and for the node disjoint version of k-DSPP in planar directed graphs. They further give algorithms for k-DSPP in *undirected* planar graphs. As for k-DPP, the first results for k-DSPP are algorithms for the case $k = 2$. Already Eilam-Tzoreff [Eil98] gives an $\mathcal{O}(n^8)$ algorithm for the node and edge disjoint version of 2-DSPP on graphs with positive edge lengths. For the node disjoint case Akhmedov [Akh20] improves these algorithms to $\mathcal{O}(n^7)$ and $\mathcal{O}(n^6)$ for unit length edges. Apart from the algorithm for planar graphs by Bérczi and Kobayashi [BK17], there are only efficient algorithms for the unit length case by Lochet [Loc21] and Bentert, Nichterlein, Renken, and Zschoche [Ben+21] for k-DSPP when k is fixed. The complexity of k-DSPP for fixed $k \geq 3$ and general edge lengths is open for undirected graphs. We summarize the known complexity results for k-DSPP in Table 5.2.

Variations. There are several variations of k-DSPP. We mention only some of them without giving a complete list of known results. Eilam-Tzoreff [Eil98] studies 2-DSPP, where *only one* of the paths is constrained to be a shortest path. She shows that this problem is NP-complete in directed and undirected graphs for node and edge disjoint paths, even for unit length edges. Cai and Ye [CY16] develop FPT-algorithms for several bounds on the lengths of the two paths. Their parameters are different sums of the two bounds. So for the problem mentioned above studied by Eilam-Tzoreff, they give an algorithm that is polynomial in the length of a shortest $s_1 - t_1$ path.

	undirected			directed	
	$\ell = 1$	$\ell > 0$	$\ell \geq 0$	$\ell > 0$	$\ell \geq 0$
k arb.	NP-compl. [Eil98]	NP-compl. [Eil98]	NP-compl. [Kar72]	NP-compl. [Eil98]	NP-compl. [EIS76]
k fix	P [Loc21; Ben+21]				NP-compl. [FHW80]
$k = 2$	P [Eil98]	P [Eil98]	P * [KS19; GKW19]	P [BK17]	NP-compl. [FHW80]

Table 5.2: Complexity of k-DSPP for $k \geq 2$ and constraints on the edge lengths ℓ. An algorithm for the undirected 2 disjoint shortest paths problem with $\ell \geq 0$ (marked with *) is the topic of this chapter. This result has been obtained independently by Kobayashi and Sako [KS19].

Other versions include minimizing the *total length* of both paths (in contrast to individually minimizing their length). Suurballe [Suu74] is the first to study this version where all sources are the same and all sinks are the same. Later, Björklund and Husfeldt [BH14; BH19] give a randomized algorithm for two source-sink pairs and unit length edges. Instead of minimizing the total length, Li, McCormick, and Simchi-Levi [LMS90] consider minimizing the *maximum* length of both paths. They show that this problem is NP-complete in undirected and directed graphs for node and edge disjoint paths. Amiri and Wargalla [AW20] consider k-DSPP where the paths do not have to be completely node disjoint, but *up to c* paths can meet in a node. They give a polynomial-time algorithm for fixed k in directed acyclic graphs with unit length arcs.

On the other hand, the edge disjoint version of k-DPP is a special case of *integral multi-commodity flow* where all capacities are 1. Even, Itai, and Shamir [EIS76] show that the problem is NP-complete even for two commodities in directed and undirected graphs. For instances of k-DPP where all sources are the same and all sinks are the same, the problem reduces to the standard maximum flow problem using Menger's theorem [Men27].

Our Results We give a polynomial-time algorithm for solving 2-DSPP with non-negative edge lengths. The restriction to shortest paths is guaranteed by directing edges according to shortest paths networks. This removes the length information and results in a *mixed* graph that contains undirected edges and directed arcs. We introduce the class of *weakly acyclic* mixed graphs and give an algorithm to solve k-DPP in such graphs. This problem appears as a subproblem when solving 2-DSPP. Our algorithm follows the one from Bérczi and Kobayashi [BK17] for 2-DSPP on directed graphs. They reduce the problem to 2-DPP in directed acyclic graphs. We reduce 2-DSPP on undirected graphs to 2-DPP on weakly acyclic mixed graphs. Our algorithm for this subproblem is a generalization of the one from Fortune, Hopcroft, and Wyllie [FHW80] for k-DPP in directed acyclic graphs.

We remark here that Kobayashi and Sako [KS19] independently developed a polynomial-time algorithm for 2-DSPP.

5.2 Model and Notation

Model We study the node disjoint version of the undirected k *disjoint shortest paths problem (k-DSPP)*. An instance is given by an undirected graph $G = (V, E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and k source and sink nodes s_i and t_i in V, that are pairwise different. The task is to find a *shortest* $s_i - t_i$ path in G w.r.t. ℓ for each of the k source-sink pairs, such that all of the paths are pairwise *node disjoint*, or decide that no such paths exist. Two paths (as sequence of nodes) are node disjoint if they do not have a node in common. If the paths are not restricted to be shortest $s_i - t_i$ paths, the problem is called the k *disjoint paths problem (k-DPP)*. See Figure 5.2 for an example of 2-DSPP.

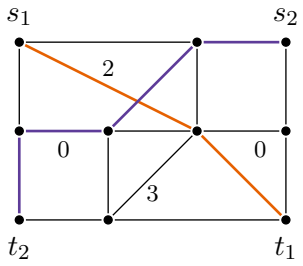


Figure 5.2: An instance of 2-DSPP. The edge labels give the length of the edge. Edges without label have length 1. Two node disjoint shortest $s_i - t_i$ paths are shown in orange and purple.

An instance of k-DSPP can be modeled as a congestion game with k players. The shared resources are the nodes of the graph $G = (V, E)$ with increasing cost functions, where the cost for one player is 0 and for more than one player ∞. The available strategies for a player i are all shortest $s_i - t_i$ paths in G w.r.t. ℓ. We are interested in the existence of a profile with finite cost.

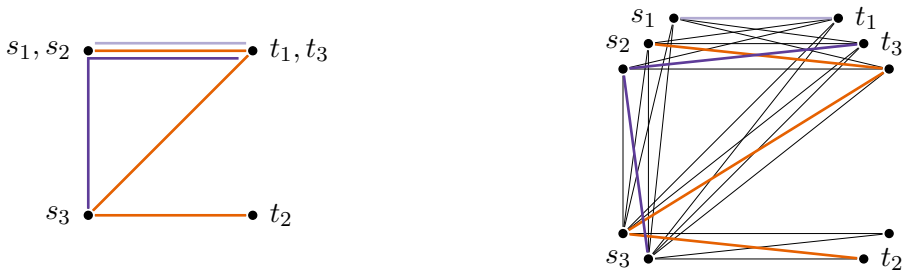
We will use the graph theoretic interpretation of the problem rather than the game theoretic one in this chapter.

Relation to Other Problems We are interested in *node* disjoint paths. There are two closely related versions of the disjoint paths problem. We show that both of them can be modeled in our setting.

Internally Node Disjoint Paths. In some settings, looking at node disjoint paths is too restrictive. It disallows for sources and sinks to lie on some of the shortest paths chosen by other players, or instances where sources and sinks may coincide. If we want to allow these scenarios, we are looking for *internally* node disjoint paths. Two $s_i - t_i$ paths $P_1 = (s_1, p_1, \ldots, p_l, t_1)$ and $P_1 = (s_2, q_1, \ldots, q_m, t_2)$ are internally node disjoint, if the sets $\{p_1, \ldots, p_l\}$ and $\{q_1, \ldots, q_m\}$ of internal nodes are disjoint.

However, we can transform an instance of internally node disjoint paths into an instance of node disjoint paths by copying nodes. We add a copy of every node that is a

source or a sink together with its incident edges. If a node is used as source or sink for several players, we add copies for each of them. Figure 5.3 shows an example.



(a) A graph G with three internally node disjoint $s_i - t_i$ paths.

(b) The corresponding node disjoint $s_i - t_i$ paths in $\overline{G}$.

Figure 5.3: Transforming an instance with internally node disjoint paths into an instance with node disjoint paths.

In this way, the sources and sinks end up being all different and there is a one-to-one correspondence between internally node disjoint $s_i - t_i$ paths in the original graph and node disjoint $s_i - t_i$ paths in the new graph.

Take a set of internally node disjoint $s_i - t_i$ paths in the original graph. Every path has a canonical image in the new graph, obtained by the same copy transformation. These images are node disjoint paths in the new graph. In the original graph the paths intersected only at source or sink nodes. These intersections are resolved by the transformation.

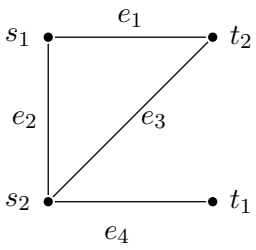
On the other hand, a set of node disjoint $s_i - t_i$ paths in the new graph, can be mapped to a set of paths in the original graph by merging the copies of each node into a single node. The resulting paths can only intersect in source or sink nodes, hence they are internally node disjoint.

The mappings between paths in the original graph and in the new graph maintain the lengths of the paths.

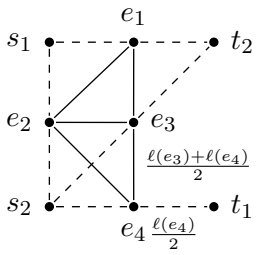
Edge Disjoint Paths. Instead of taking the resources to be nodes we can also consider edges. We are then looking for *edge* disjoint paths. An instance for edge disjoint paths can be transformed to node disjoint paths by taking the line graph.

The *line graph* of a graph is constructed by swapping the roles of edges and nodes. For every edge in the graph there is a node in the line graph. Two edges are adjacent in the line graph, if they are incident in the original graph. We add the sources and sinks as nodes to the line graph and connect them to their incident edges. The length of an edge in the original graph is distributed to the incident edges in the new graph. Each of those gets half of the length. If the edges e_1 and e_2 are adjacent in the line graph, then the edge $\{e_1, e_2\}$ has length $\frac{\ell(e_1)}{2} + \frac{\ell(e_2)}{2}$. Figure 5.4 shows an example of the construction.

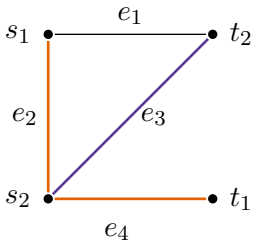
There is a canonical translation of paths in G to paths in $\overline{G}$ by using the edges of G as nodes on the path in $\overline{G}$. For paths in $\overline{G}$ there also is a canonical translation to edge sets in G, taking all edges used on the path. Notice that this translation does not necessarily result in paths in G. In the example of Figure 5.4 the path $s_2 - e_3 - e_1 - t_2$



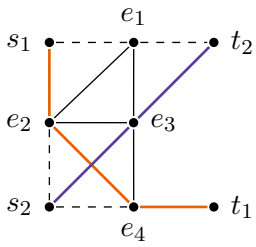
(a) A graph G with 4 nodes and edges, and 2 pairs of source and sink nodes



(b) The graph $\overline{G}$ consisting of the line graph of G (solid edges) and the source and sink nodes with their connections (dashed). We only show two edge labels illustrating the construction of the lengths on new edges $\{e_3, e_4\}$ and $\{e_4, t_1\}$.



(c) Two edge disjoint $s_i - t_i$ paths in G.



(d) The corresponding node disjoint $s_i - t_i$ paths in $\overline{G}$.

Figure 5.4: Transforming an instance with edge disjoint paths to an instance with node disjoint paths.

is a feasible $s_2 - t_2$ path in $\overline{G}$. Its image is the set $\{e_3, e_1\}$. We observe that this set is not an $s_2 - t_2$ path in G. However, every image of a path in $\overline{G}$ contains a valid path in G. In the above example, just taking edge e_3 is an $s_2 - t_2$ path.

Lemma 40. *Every canonical image in G of an $s_i - t_i$ path in $\overline{G}$ contains an $s_i - t_i$ path in G.*

Proof. Let $P = (s_i, e_1, \ldots, e_l, t_i)$ be an $s_i - t_i$ path in $\overline{G}$. By construction of $\overline{G}$, s_i and t_i are connected in the canonical image of P in G. Now let P' be an $s_i - t_i$ path in $\overline{G}$ using only nodes in $\{e_1, \ldots, e_l\}$ with minimal number of nodes. The canonical image of P' connects s_i and t_i in G with minimal number of edges from $\{e_1, \ldots, e_l\}$. Hence, it is an $s_i - t_i$ path in G. □

With the previous lemma we can show a one-to-one correspondence between edge disjoint shortest paths in the original graph G and node disjoint shortest paths in the new graph $\overline{G}$.

Take a collection of edge disjoint shortest $s_i - t_i$ paths in G. Every path has a canonical image in $\overline{G}$, where the edges on the paths are now nodes on the path (see Figure 5.4d). By construction of $\overline{G}$ these images are node disjoint $s_i - t_i$ paths. Further, the length of the paths are maintained, as every node in $\overline{G}$ corresponding to an edge in G is an

internal node with degree 2 in an image of a path. Thus the length of the edge in G is split to the two incident edges on the path in $\overline{G}$. The missing half of the length of the first and last edge on the path, is covered by the additional edges from the source and sink nodes in $\overline{G}$. With the same arguments, the length of any path in $\overline{G}$ is the sum of the lengths of the edges (nodes) on the path. With the previous lemma we have that any $s_i - t_i$ path in $\overline{G}$ is at least as long as an $s_i - t_i$ path in G. This shows that the images are indeed shortest paths in $\overline{G}$.

For the other direction take a set of node disjoint shortest $s_i - t_i$ paths in $\overline{G}$. From Lemma 40 the images of these paths contain $s_i - t_i$ paths in G. These paths are edge disjoint. Let $\overline{P}$ be an $s_i - t_i$ path in $\overline{G}$ and P be the $s_i - t_i$ path in G in the canonical image of $\overline{P}$. By construction we have $\ell(P) \leq \ell(\overline{P})$. Any $s_i - t_i$ path in G translates to an $s_i - t_i$ path in $\overline{G}$ with the same length. As $\overline{P}$ is a shortest $s_i - t_i$ path in $\overline{G}$, its length is at most the length of any other $s_i - t_i$ path. In particular, at most the length of the images of $s_i - t_i$ paths in G. We thus obtain that P is a shortest $s_i - t_i$ path in G.

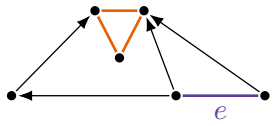
With these transformations we can restrict our attention to *node* disjoint paths. When we write just "disjoint" for paths in the remainder of this chapter, we mean node disjoint.

Mixed Graphs To solve the disjoint shortest path problem on undirected graphs, we use an auxiliary graph where some of the edges are directed.

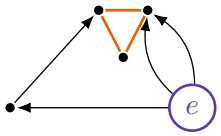
Graphs with (undirected) edges and (directed) arcs are called *mixed graphs*. We denote a mixed graph as $G = (V, A \dot\cup E)$, where A contains the arcs and E the edges. We write $L = A \dot\cup E$ for the set of all *links* in G. We assume that the link type between any pair of nodes is unique, that is, we don't have $\{u, v\} \in E$ and $(u, v) \in A$ or $(v, u) \in A$ at the same time. The notion of a $u-v$ path is extended in the natural way to mixed graphs: A *path* is a sequence of different nodes such that the link between two subsequent nodes is either an edge in E or an arc in A, i.e., $P = (v_0, \dots, v_k)$ is a path if and only if all nodes are different and for all $i \in \{1, \dots, k\}$ we have either $\{v_{i-1}, v_i\} \in E$ or $(v_{i-1}, v_i) \in A$. For a set of links L we define $V(L)$ to be the set of nodes incident to links in L. For example $V(\{\{1, 2\}, (2, 3)\}) = \{1, 2, 3\}$.

The auxiliary graphs will have a crucial property: they do not contain directed cycles. To state this formally, we introduce the notion of *contracting* an edge. We denote by G/e the multigraph resulting from G by identifying the two endpoints of edge e as a single node and removing all links between them. Figure 5.5b shows an example of a contraction of an edge. This operation can be extended to a set of edges E' where G/E' is the multigraph resulting from G by contracting all edges in E' in any order. If we contract all edges of a mixed graph we end up with a directed multigraph. We set $G^A = G/E$. If during the contractions no arc in A has been removed and G^A is a directed acyclic multigraph, then the mixed graph G is called *weakly acyclic*. The example graph in Figure 5.5 is weakly acyclic. The multigraph G^A (shown in Figure 5.5c) is acyclic. Notice that weakly acyclic graphs can contain undirected cycles.

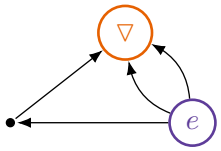
Instead of looking only at the arcs of a mixed graph, we also look at the graph $G^E = (V, E)$ containing only the edges of G. Note that in contrast to G^A we are not performing any contractions of arcs, we just remove all arcs from G.



(a) A mixed graph G with arcs (black) and edges (orange and purple).



(b) The contracted multigraph G/e.



(c) The multigraph G^A obtained from G by contracting all edges.

Figure 5.5: A weakly acyclic mixed graph G (a). The multigraph G^A (c). The graph G^E consists of the two components containing the purple and orange edges and the single node on the left.

A *connected component* in an undirected graph is an inclusion-wise maximal subset of the nodes that is connected. I.e., there is a path between any pair of nodes in the set and no node can be added while maintaining this property. In a weakly acyclic mixed graph the connected components of G^E can be ordered according to a topological ordering of G^A. In Figure 5.5c the topological ordering of the three components of G^E is: $e, \bullet, \triangledown$.

Path Relations Our algorithm to solve 2-DSPP does not just compute the two disjoint $s_i - t_i$ paths if they exist. Instead we compute all pairs of pairs of nodes where two disjoint shortest paths exist.

Since we consider node disjoint paths, the sources and sinks have to have different entries. We thus define the notation $V^k_{\neq}$ for the set of k tuples of pairwise different elements of V.

For an undirected graph $G = (V, E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and a subset of edges $E' \subseteq E$, we define the relation $\overset{\ell}{=\!=}_{E'}$ on $V^2_{\neq}$, where

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \overset{\ell}{=\!=}_{E'} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

if there exists a shortest $u_1 - v_1$ path and a shortest $u_2 - v_2$ path w.r.t. ℓ in the edge set E' which are disjoint.

By building an auxiliary mixed graph we go from shortest paths to directed paths. The corresponding version of the relation is extended to k tuples of nodes. For a mixed graph $G = (V, L)$ and a subset of links $L' \subseteq L$ define the relation $\overset{\rightarrow}{\rightarrow}_{L'}$ on $V^k_{\neq}$ by

$$u \overset{\rightarrow}{\rightarrow}_{L'} v$$

if there are disjoint $u_i \to v_i$ paths in L' for all $i \in \{1, \ldots, k\}$.

For the special case where $k = 2$ we use the symbol $\rightrightarrows$. We further allow to restrict the set of links for both paths individually. For two subsets of links $L_1, L_2 \subseteq L$, define the relation $\rightrightarrows^{L_1}_{L_2}$ on $V^2_{\neq}$ by

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \rightrightarrows^{L_1}_{L_2} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

if there is a $u_1 \to v_1$ path in L_1 and a $u_2 \to v_2$ path in L_2 which are disjoint.

For technical reasons we define an asymmetric version of the previous relation for $k = 2$. For two subsets of links $L_1, L_2 \subseteq L$, define the relation $\rightleftarrows_{L_2}^{L_1}$ on $V_{\neq}^2$ by

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \rightleftarrows_{L_2}^{L_1} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$$

if there is a $u_1 \to v_1$ path in L_1 and a $v_2 \to u_2$ path in L_2 which are disjoint. Note that obviously

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \rightleftarrows_{L_2}^{L_1} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \iff \begin{pmatrix} u_1 \\ v_2 \end{pmatrix} \rightrightarrows_{L_2}^{L_1} \begin{pmatrix} v_1 \\ u_2 \end{pmatrix}.$$

Thus, if we have one of the relations, we obtain the other by swapping the second components. However, composing two of those relations behaves differently. The composition $\rightrightarrows' \circ \rightrightarrows$ builds both paths in the same direction. That is $u_1 \to v_1 \to w_1$ and $u_2 \to v_2 \to w_2$. On the other hand $\rightleftarrows' \circ \rightleftarrows$ builds one of the paths backwards. I.e., $u_1 \to v_1 \to w_1$ but $u_2 \leftarrow v_2 \leftarrow w_2$. Constructing paths in opposite directions is one of the crucial ideas for our algorithm.

Since we consider only tuples of nodes with different entries, all of the above relations are reflexive. We define the relation

$$\mathrm{Id}(V) = \{(v, v) : v \in V\}$$

consisting of all pairs of elements of V with the same entries. Observe that $\mathrm{Id}\left(V_{\neq}^k\right)$ is contained in any of the above path relations.

For the relations on mixed graphs we show conditions for which transitivity holds.

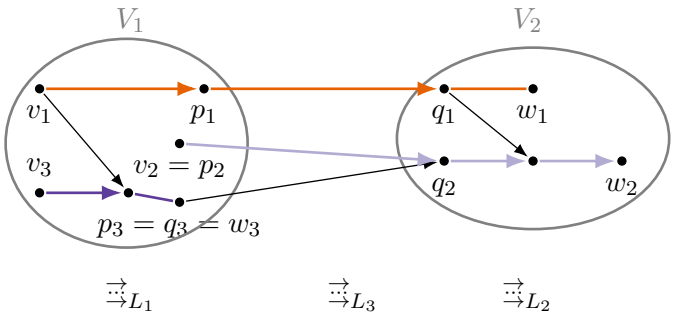


Figure 5.6: The situation in Lemma 41. The three node disjoint paths showing $v \rightrightarrows_{L_1} p \rightrightarrows_{L_3} q \rightrightarrows_{L_2} w$ are highlighted with colors.

Lemma 41. *Let $G = (V, L)$ be a mixed graph and $V_1, V_2 \subseteq V$ two subsets of nodes that are disjoint. Let $L_1, L_2 \subseteq L$ be two sets of links with the property that $V(L_1) \subseteq V_1$ and $V(L_2) \subseteq V_2$. Further let $L_3 \subseteq V_1 \times V_2$ be a set of arcs in L from V_1 to V_2. Then*

$$\forall v, w \in V_{\neq}^k : \left(\exists p, q \in V_{\neq}^k : v \rightrightarrows_{L_1} p \rightrightarrows_{L_3} q \rightrightarrows_{L_2} w\right) \iff v \rightrightarrows_{L_1 \dot\cup L_2 \dot\cup L_3} w$$

Proof. First observe that the three sets of links L_1, L_2 and L_3 are pairwise disjoint as their underlying node sets are different.
"$\Rightarrow$": Let $v, w, p, q \in V_{\neq}^k$ such that $v \overset{\rightarrow}{\rightarrow}_{L_1} p \overset{\rightarrow}{\rightarrow}_{L_3} q \overset{\rightarrow}{\rightarrow}_{L_2} w$ and consider the paths $v_i \to p_i$ in L_1, $p_i \to q_i$ in L_3 and $q_i \to w_i$ in L_2 for all $i \in \{1, \dots, k\}$ (see Figure 5.6).

As $V(L_1) \subseteq V_1$ and $V(L_2) \subseteq V_2$, and $V_1 \cap V_2 = \emptyset$, the paths $v_i \to p_i$ and $q_i \to w_i$ are disjoint. Since L_3 consists only of arcs going from V_1 to V_2 the path $p_i \to q_i$ is a single arc. Thus the concatenation of all three paths is a valid simple $v_i \to w_i$ path in the union of the three link sets.

Further, all three parts of all $v_i \to w_i$ paths are node disjoint in L_1, L_2 and L_3. Since the entries of p and q are all different, also the concatenations are pairwise node disjoint, showing $v \overset{\rightarrow}{\rightarrow}_{L_1 \dot\cup L_2 \dot\cup L_3} w$.
"$\Leftarrow$": Take $v, w \in V_{\neq}^k$ with $v \overset{\rightarrow}{\rightarrow}_{L_1 \dot\cup L_2 \dot\cup L_3} w$ and let P_i be the corresponding $v_i \to w_i$ path for each $i \in \{1, \dots, k\}$. Similar to the previous direction we get from the disjointness of the node sets of L_1 and L_2 and the direction of arcs in L_3, that every path P_i can be split in three parts: a subpath in L_1, an arc in L_3, and finally a subpath in L_2. Each of the parts can be empty. Let p_i be the last node on P_i in $V(L_1)$. Set $p_i = v_i$ if P_i does not visit $V(L_1)$. Further let q_i be the first node on P_i in $V(L_2)$. Set $q_i = w_i$ if P_i does not visit $V(L_2)$. Figure 5.6 shows example paths and the corresponding p_i and q_i. Since the whole paths P_i are node disjoint also the subpaths are node disjoint in the respective link sets and $p, q \in V_{\neq}^k$ shows the claim. □

For $k = 2$ this transitivity holds also for restricted sets of links for both paths. We phrase the statement using the asymmetric path relation, as this is how we are going to apply it later.

Corollary 6. *Let $G = (V, E)$ be a mixed graph and $V_1, V_2 \subseteq V$ two subsets of nodes that are disjoint. Let $L_1, L_1', L_2, L_2' \subseteq L$ be sets of links with the properties $V(L_1), V(L_1') \subseteq V_1$ and $V(L_2), V(L_2') \subseteq V_2$. Further let $L_3 \subseteq V_1 \times V_2$ and $L_3' \subseteq V_2 \times V_1$ be sets of arcs in L from V_1 to V_2 and vice versa. Then*

$$\forall v, w \in V_{\neq}^2 : \left(\exists p, q \in V_{\neq}^2 : v \overset{L_1}{\underset{L_1'}{\rightleftarrows}} p \overset{L_3}{\underset{L_3'}{\rightleftarrows}} q \overset{L_2}{\underset{L_2'}{\rightleftarrows}} w\right) \iff v \overset{L_1 \dot\cup L_2 \dot\cup L_3}{\underset{L_1' \dot\cup L_2' \dot\cup L_3'}{\rightleftarrows}} w$$

These compositions of relations on separate node sets is the key building block of our algorithms.

As subroutines we compute relations on parts of the graph. To compose these relations we have to extend them to the whole graph. We introduce the process of *extending* a relation R on $U_{\neq}^k$ to $V_{\neq}^k$, where $U \subseteq V$. The extension contains $\mathrm{Id}\left(V_{\neq}^k\right)$ and for every element $((u_1, \dots, u_k), (v_1, \dots, v_k))$ of R we add elements of the form

$$\begin{aligned}&((u_1, \dots, u_{i-1}, x_i, u_{i+1}, \dots, u_{j-i}, x_j, \dots, x_{j+3}, u_{j+4}, \dots, u_k),\\ &\ (v_1, \dots, v_{i-1}, x_i, v_{i+1}, \dots, v_{j-i}, x_j, \dots, x_{j+3}, v_{j+4}, \dots, v_k)),\end{aligned}$$

where we replaced some of the entries with pairwise different elements x_i from $V \setminus U$ on both sides. This replacement can be done for any position and any number of positions in the vectors. Note that if the relation R is empty, its extension is equal to $\mathrm{Id}\left(V_{\neq}^k\right)$.

The size of the extension is upper bounded by $\binom{|V|}{k}$ for $\mathrm{Id}\left(V_{\neq}^{k}\right)$ plus $|R| \cdot |V \setminus U| \cdot 2^k$ for the new elements resulting from the elements of R. Hence the size is bounded by a polynomial in the size of R.

Common Idea of the Algorithms The two problems "Find two disjoint shortest paths in an undirected graph" and "Find k disjoint paths in a weakly acyclic mixed graph" are solved by the same high-level idea. Both problems are transformed to finding disjoint paths in a mixed graph. We do not only compute the $s_i - t_i$ paths, but rather a large part of the corresponding disjoint path relation. We identify two layers in the mixed graph and compute the relation separately on those layers. The *inner* layer consists of disjoint components where computing disjoint paths can be handled by some subroutine. The paths computed in these layers are then extended to paths in the whole graph. The inner components are connected by the *outer* layer. For both problems the outer layer has an *acyclic* structure. This is helpful as we then know the order in which paths in the whole graph pass through the inner components. We can thus use dynamic programming to extend the paths in the inner components to larger parts of the whole graph.

The three key properties for this approach are

- finding disjoint paths in the *inner components* can be done efficiently
- the outer layer is *acyclic*
- the disjoint paths relation on the mixed graph can be *decomposed* to the inner components as in Lemma 41 and Corollary 6.

We begin with an algorithm for k-DPP in weakly acyclic mixed graphs. This algorithm is then used as subroutine for the inner components for 2-DSPP.

5.3 Disjoint Paths in Weakly Acyclic Mixed Graphs

In this section, we give an algorithm to solve the problem of finding disjoint paths in weakly acyclic mixed graphs. This algorithm is used as a subroutine in the algorithm for solving 2-DSPP. In the special case where the mixed graph does not contain any edges, the graph is a directed acyclic graph and our algorithm resembles the one of Fortune, Hopcroft, and Wyllie [FHW80].

Formally, we consider the following problem. Given a mixed graph $G = (V, L)$ and two vectors of k sources $s \in V_{\neq}^{k}$ and k sinks $t \in V_{\neq}^{k}$, decide whether there are k pairwise node disjoint $s_i \to t_i$ paths in G.

Following the high-level idea of the algorithm as explained above in Section 5.2, we solve this problem by computing the relation $\overset{\rightarrow}{\rightarrow}_L$ on $V_{\neq}^{k}$ as shown in Algorithm 3. That is, we compute all pairs of source and sink vectors having disjoint paths between them.

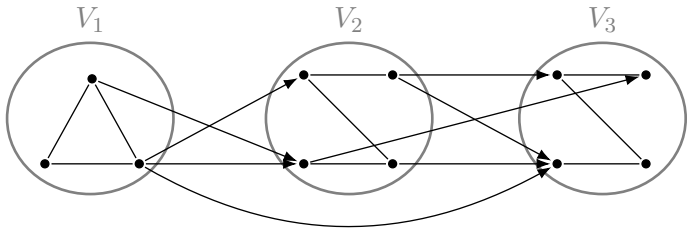
The two layers of the mixed graph are specified by the arcs and edges. The outer layer is the *arc* layer G^A containing only the directed arcs, and the inner layer is the *edge* layer G^E containing only the undirected edges. The connected components of the edge layer are the inner components. Finding disjoint paths in these components corresponds to finding disjoint paths in an undirected graph. Hence it can be solved efficiently by any algorithm for k-DPP, for example the one from Robertson and Seymour [RS95]. As the

whole graph is weakly acyclic, the outer layer is acyclic by definition. Thus, we can order the inner components according to a topological ordering of the arc layer. Hence, paths in G have to traverse the connected components of the edge layer in the order given by the arc layer. We will thus use Lemma 41 for the connected components in the edge layer and the arcs between this components from the arc layer. Figure 5.7 shows one update iteration of Algorithm 3 in a mixed graph.

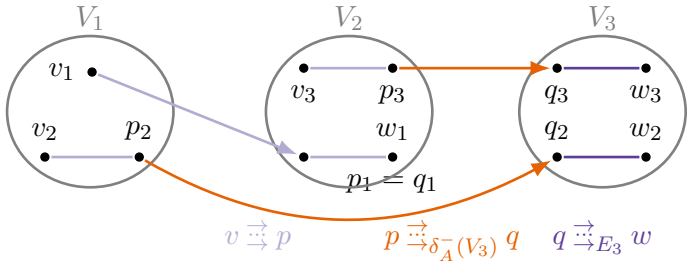
Input: weakly acyclic mixed graph $G = (V, A \dot\cup E)$, $k \in \mathbb{N}_{\geq 2}$
Output: $\overset{\rightarrow}{\rightarrow}_{A \cup E}$ on $V_{\neq}^k$

1 **let** $(V_1, E_1), \dots, (V_l, E_l)$ *be the connected components of* G^E *sorted according to a topological ordering of* G^A
2 **for** $j = 1, \dots, l$ **do**
3 Compute $\overset{\rightarrow}{\rightarrow}_{E_j}$ using an algorithm for k-DPP and extend it to $V_{\neq}^k$
4 Compute $\overset{\rightarrow}{\rightarrow}_{\delta_A^-(V_j)}$ on $V_{\neq}^k$
5 **let** $\overset{\rightarrow}{\rightarrow} = \mathrm{Id}\left(V_{\neq}^k\right)$
6 **for** $j = 1, \dots, l$ **do**
7 Update $\overset{\rightarrow}{\rightarrow}$ to $\overset{\rightarrow}{\rightarrow}_{E_j} \circ \overset{\rightarrow}{\rightarrow}_{\delta_A^-(V_j)} \circ \overset{\rightarrow}{\rightarrow}$
8 **return** $\overset{\rightarrow}{\rightarrow}$

Algorithm 3: Solving k-DSPP for fixed k in weakly acyclic mixed graphs



(a) A mixed graph with the connected components V_1, V_2, V_3 of G^E in a topological ordering of G^A.



(b) One iteration of extending the disjoint paths in the components by arcs of the outer layer. The relation $\overset{\rightarrow}{\rightarrow}$ has already been computed on the first two components. It is extended to V_3. Three disjoint $v_i \to w_i$ paths are shown, illustrating the composition of the relations.

Figure 5.7: Example of an iteration of Algorithm 3.

Theorem 15. *For a fixed $k \in \mathbb{N}_{\geq 2}$, Algorithm 3 computes $\overset{\rightarrow}{\rightarrow}_L$ on $V_{\neq}^k$ for a weakly acyclic mixed graph $G = (V, L)$ in polynomial time.*

Proof.

Correctness. Let $(V_1, E_1), \ldots, (V_l, E_l)$ be the connected components of G^E sorted by a topological ordering of G^A, as computed in Line 1. For every $j \in \{0, \ldots, l\}$ we define L_j as the set of arcs and edges in the subgraph of G induced by the first j components $V_1, \ldots, V_j$. Further let $\overset{\rightarrow}{\rightarrow}^j$ be the relation $\overset{\rightarrow}{\rightarrow}$ after the j-th iteration of Line 7. With these definitions we have in particular $L_0 = \emptyset$ and $\overset{\rightarrow}{\rightarrow}^0 = \mathrm{Id}\left(V_{\neq}^k\right)$ from Line 5.

We prove the invariant

$$\overset{\rightarrow}{\rightarrow}^j = \overset{\rightarrow}{\rightarrow}_{L_j} \text{ on } V_{\neq}^k \tag{5.1}$$

by induction on j. The base case for $j = 0$ follows immediately from the definitions. Now assume (5.1) is true for j and take $\overset{\rightarrow}{\rightarrow}^{j+1} = \overset{\rightarrow}{\rightarrow}_{E_{j+1}} \circ \overset{\rightarrow}{\rightarrow}_{\delta_A^-(V_{j+1})} \circ \overset{\rightarrow}{\rightarrow}^j$.

Observe that $V(L_j) \subseteq V_1 \cup \ldots \cup V_j$ by the definition of L_j, and $V(E_{j+1}) \subseteq V_{j+1}$. Thus these two sets of links are separated as in Lemma 41. From G being weakly acyclic we further have $\delta_A^-(V_{j+1}) \subseteq (V_1 \cup \ldots \cup V_j) \times V_{j+1}$. Thus all conditions of Lemma 41 are satisfied. As $L_j \dot\cup \delta_A^-(V_{j+1}) \dot\cup E_{j+1} = L_{j+1}$, (5.1) is shown for $j + 1$.

Running time. The graphs G^E and G^A can be constructed from G in polynomial time. Further finding the connected components $(V_1, E_1), \ldots, (V_l, E_l)$ can be done with a breadth first search and a topological ordering of G^A can be found in time polynomial in the size of G by successively removing vertices with in-degree zero.

As there are efficient algorithms for k-DPP (e.g., [RS95]), $\overset{\rightarrow}{\rightarrow}_{E_j}$ can be computed in polynomial time. Its extension to $V_{\neq}^k$ can be computed in time polynomial in the size of $\overset{\rightarrow}{\rightarrow}_{E_j}$ which is upper bounded by $|V|^{2k}$.

The relations $\overset{\rightarrow}{\rightarrow}_{\delta_A^-(V_j)}$ consist of all matchings between nodes in V_j and nodes in the preceding components $V_1 \cup \ldots \cup V_{j-1}$. As relations on $V_{\neq}^k$ they contain at most $|V|^{2k}$ elements. Hence they can be computed in polynomial time, even by complete enumeration. Similarly, the composition of relations on $V_{\neq}^k$ can be computed in polynomial time. As the number of update iterations is at most $|V|$, the algorithm runs in time polynomial in $|V|$. □

5.4 Disjoint Shortest Paths in Undirected Graphs

We are now giving the algorithm to solve the two disjoint shortest path problem for non-negative edge lengths. From the undirected graph we construct a weakly acyclic mixed graph, such that the shortest paths in the original graph correspond to directed paths in the mixed graph. To solve the problem we use the algorithm from the previous section as subroutine. This approach is based on the algorithm of Bérczi and Kobayashi [BK17] for 2-DSPP in directed graphs.

5.4.1 From Shortest Paths to Directed Paths

To restrict the set of available paths to the shortest $s_i - t_i$ paths in the undirected graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$, we consider the *shortest paths networks* rooted at the two sources s_1 and s_2. The shortest paths network consists of all edges appearing on shortest paths from the source to the other nodes. The edges of the graph get directed by increasing distance to the source. The formal definition is given in the following.

For a source s define the *distance function* $d^s : V \to \mathbb{R}_{\geq 0}$ by

$$d^s(v) = \min_{s-v \text{ path } P} \sum_{e \in P} \ell(e).$$

The distance of node v is the length of a shortest $s - v$ path in G w.r.t. ℓ.

The shortest paths network consists of all edges appearing on shortest $s - v$ paths. These are edges where the length of the edge is exactly the difference of the distances of its endpoints. We set

$$E^s = \{\{u, v\} \in E : \ell(\{u, v\}) = |d^s(u) - d^s(v)|\}.$$

For edges in E^s with strictly positive length the distance increases from u to v or vice versa. We orient the edges in E^s by increasing distance. The shortest paths network of G rooted at s is the mixed graph $\overrightarrow{G}_s = (V, E_0 \cup A^s)$, where E_0 are the edges of length 0 in G and

$$A^s = \{(u, v) : \{u, v\} \in E^s \wedge d^s(u) < d^s(v)\}.$$

To solve 2-DSPP, we take the union of both shortest paths networks rooted at s_1 and s_2. Define the *shortest path orientation* of G as the mixed graph

$$\overrightarrow{G} = (V, E_0 \cup A^{s_1} \cup A^{s_2}).$$

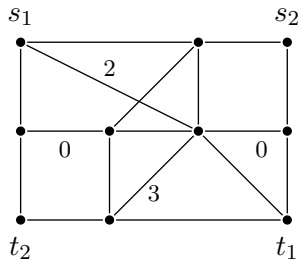
Figure 5.8 shows an example of a shortest path orientation. Notice that the arc sets A^{s_1} and A^{s_2} may have arcs in common, and it is possible that $(u, v) \in A^{s_1}$ and $(v, u) \in A^{s_2}$.

With this construction, shortest $s_i - t_i$ paths in G (as sequence of nodes) are exactly the mixed $s_i \to t_i$ paths in $E_0 \cup A^{s_i}$ in the shortest path orientation of G.

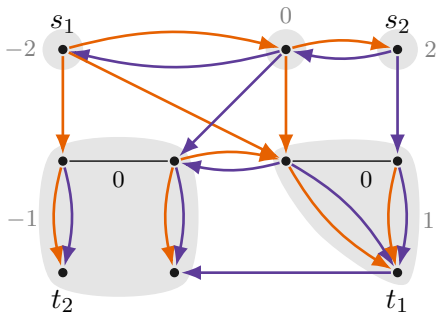
Observation 9. For an undirected graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and a pair of sources $(s_1, s_2) \in V^2_{\neq}$ the following holds for all $(t_1, t_2) \in V^2_{\neq}$

$$\begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \overset{\ell}{\underset{E}{\rightleftharpoons}} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \text{ in } G \iff \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \overset{E_0 \cup A^{s_1}}{\underset{E_0 \cup A^{s_2}}{\rightrightarrows}} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \text{ in } \overrightarrow{G}.$$

This shortest path orientation is the mixed graph used in the algorithm for 2-DSPP. We show that the shortest path orientation has the properties mentioned in the description of the high-level idea in Section 5.2.



(a) An undirected graph G. Edge labels give the length ℓ of the edge. Edges without label have length 1.



(b) The shortest path orientation $\overrightarrow{G}$ of G. Arcs in the shortest path network rooted at s_1 are drawn orange and arcs in A^{s_2} are drawn in purple. The gray highlighted node sets are the weakly connected components of $\overrightarrow{G}^{\text{uni}}$. Their labels give the value of $d^{s_1} - d^{s_2}$.

Figure 5.8: Construction of the shortest path orientation of an undirected graph.

The inner layer is the *unidirectional* layer consisting of the arcs in $A^{s_1} \cap A^{s_2}$ which are used in the same direction in both shortest paths networks together with the zero edges. Set

$$\overrightarrow{G}^{\text{uni}} = (V, E_0 \cup (A^{s_1} \cap A^{s_2})).$$

The outer layer is the *bidirectional* layer. It is the multigraph resulting from $\overrightarrow{G}$ when contracting all weakly connected components of $\overrightarrow{G}^{\text{uni}}$. A *weakly connected component* is a connected component in the multigraph resulting from ignoring the directions of arcs. The bidirectional layer consists of arcs in $A^{s_1} \triangle A^{s_2}$. We thus split this layer into two layers corresponding to the two directed multigraphs

$$\overrightarrow{G}^{\text{bi}}_{s_1} = \overrightarrow{G}_{s_1}/L\left(\overrightarrow{G}^{\text{uni}}\right) \qquad \text{and} \qquad \overrightarrow{G}^{\text{bi}}_{s_2} = \overrightarrow{G}_{s_2}/L\left(\overrightarrow{G}^{\text{uni}}\right)$$

resulting from the shortest paths networks by contracting the links that are used in the same direction.

We show that the weakly connected components of $\overrightarrow{G}^{\text{uni}}$ can be used as inner components. Refer to Figure 5.8 to follow the statements and the proof of the next lemma.

Lemma 42. *Let $G = (V, E)$ be an undirected graph with non-negative edge lengths and $(s_1, s_2) \in V^2_{\neq}$ a pair of sources. Further let $W_1, \dots, W_l$ be the weakly connected components of $\overrightarrow{G}^{uni}$. Then all of the following hold for all $j \in \{1, \dots, l\}$*

(i) $A\left(\overrightarrow{G}[W_j]\right) \subseteq A^{s_1} \cap A^{s_2}$ *and* $E\left(\overrightarrow{G}[W_j]\right) \subseteq E_0$.

(ii) $\overrightarrow{G}[W_j]$ *is weakly acyclic.*

(iii) Sorting the components non-decreasingly w.r.t. $d^{s_1} - d^{s_2}$ is a topological ordering of $\overrightarrow{G}^{bi}_{s_1}$ and a reverse topological ordering of $\overrightarrow{G}^{bi}_{s_2}$.

Proof. First note that d^{s_1} strictly increases along arcs in A^{s_1} and is constant on edges in E_0. The same holds for d^{s_2} on arcs in A^{s_2}. Now consider $d^{s_1} - d^{s_2}$ on an arc $(v, w) \in$

$A^{s_1} \setminus A^{s_2}$. Since $(v,w) \in A^{s_1}$ we have $d^{s_1}(w) = d^{s_1}(v) + \ell(\{v,w\})$ and in particular $d^{s_1}(v) < d^{s_1}(w)$. On the other hand $(v,w) \notin A^{s_2}$. If $(w,v) \in A^{s_2}$, then $d^{s_2}(w) < d^{s_2}(v)$ and hence $d^{s_1} - d^{s_2}$ increases along (v,w). If $(w,v) \notin A^{s_2}$, then from the definition of the distance we have $d^{s_2}(w) < d^{s_2}(v) + \ell(\{v,w\})$. Together with $d^{s_1}(w) = d^{s_1}(v) + \ell(\{v,w\})$, also in this case $d^{s_1} - d^{s_2}$ increases along (v,w). By symmetry we have that $d^{s_1} - d^{s_2}$ is strictly decreasing along any arc in $A^{s_2} \setminus A^{s_1}$. On arcs in $A^{s_1} \cap A^{s_2}$ and edges in E_0, both distances change in the same way. Hence $d^{s_2} - d^{s_2}$ is constant on those links.

This shows that for all $j \in \{1, \dots, l\}$ the function is constant on all nodes in $\overrightarrow{G}^{\text{uni}}[W_j]$. Thus there can not be a link in $\overrightarrow{G}[W_j]$ from $A^{s_1} \triangle A^{s_2}$, since then the value changes. This proves (i).

Similarly, for (iii) ((ii) will be shown later) consider two components W_i and W_j such that $d^{s_1} - d^{s_2}$ is smaller on W_i than on W_j. Since $d^{s_1} - d^{s_2}$ is strictly increasing on arcs in A^{s_1}, there can not be an arc going from W_j to W_i in A^{s_1}. On the other hand, as $d^{s_1} - d^{s_2}$ is strictly decreasing on arcs in A^{s_2}, there can only be arcs from W_j to W_i in A^{s_2}. Note that with the same arguments, there can not be arcs between two components where the value of $d^{s_1} - d^{s_2}$ is the same. Hence, sorting the components non-decreasingly w.r.t. $d^{s_1} - d^{s_2}$ and breaking ties arbitrarily is a topological ordering of $\overrightarrow{G}^{\text{bi}}_{s_1}$ and a reverse topological ordering of $\overrightarrow{G}^{\text{bi}}_{s_2}$.

For (ii) take any path in $\overrightarrow{G}[W_j]$. From (i) we know that d^{s_1} increases along the path. Hence, any cycle in $\overrightarrow{G}[W_j]$ can only consist of edges in E_0. Further, there are no arcs from A^{s_1} or A^{s_2} between the nodes $V(E_0)$, as along such arcs d^{s_1} or d^{s_2} changes but on the other hand along edges in E_0 the distances do not change. Thus, no arcs are removed in the contractions and $\overrightarrow{G}[W_j]$ is indeed weakly acyclic. □

5.4.2 Solving the Two Disjoint Shortest Path Problem

We are ready to give an algorithm to solve the 2-DSPP in undirected graphs. We reduce the problem to finding disjoint paths in weakly acyclic mixed graphs.

Following the high-level idea as explained in the beginning (Section 5.2), we solve the problem by computing a part of the relation $\overset{\ell}{=}_E$ on $V^2_{\neq}$ as shown in Algorithm 4.

The underlying mixed graph is the shortest path orientation of G with the inner ($\overrightarrow{G}^{\text{uni}}$) and outer ($\overrightarrow{G}^{\text{bi}}_{s_1}$ and $\overrightarrow{G}^{\text{bi}}_{s_1}$) layers as explained above. Using the weakly connected components of $\overrightarrow{G}^{\text{uni}}$ as inner components satisfies all three properties of the high-level idea. The disjoint paths relation on the inner components can be computed efficiently. Lemma 42 (ii) shows that they are weakly acyclic and hence we can use Algorithm 3 from the previous section. Further, Lemma 42 (iii) shows that the outer layers $\overrightarrow{G}^{\text{bi}}_{s_1}$ and $\overrightarrow{G}^{\text{bi}}_{s_1}$ are each acyclic. In contrast to the connected components in weakly acyclic mixed graphs, the two $s_i - t_i$ paths in the shortest path orientation pass through the weakly connected components in opposite directions (Lemma 42 (iii)). We thus build one of the paths backwards. In this way the path relation can be decomposed to the inner components with Corollary 6. Figure 5.9 shows an update iteration of Algorithm 4 in the shortest path orientation.

Input: undirected graph $G = (V, E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and a pair of sources $s \in V_{\neq}^2$

Output: the set of successors of s w.r.t. $\overset{\ell}{=}_E$

1 **let** $\overrightarrow{G} = (V, E_0 \cup A^{s_1} \cup A^{s_2})$ *be the shortest path orientation of* G

2 **let** $(W_1, L_1), \ldots, (W_l, L_l)$ *be the weakly connected components of* $\overrightarrow{G}^{uni}$ *sorted non-decreasingly w.r.t.* $d^{s_1} - d^{s_2}$

3 **for** $j = 1, \ldots, l$ **do**

4 Compute $\rightleftarrows_{L_j}^{L_j}$ in W_j using Algorithm 3 and extend it to $V_{\neq}^2$

5 Compute $\rightleftarrows_{\delta^+_{A^{s_2}}(W_j)}^{\delta^-_{A^{s_1}}(W_j)}$ on $V_{\neq}^2$

6 **let** $\rightleftarrows = \mathrm{Id}\left(V_{\neq}^2\right)$

7 **for** $j = 1, \ldots, l$ **do**

8 Update $\rightleftarrows$ to $\rightleftarrows_{L_j}^{L_j} \circ \rightleftarrows_{\delta^+_{A^{s_2}}(W_j)}^{\delta^-_{A^{s_1}}(W_j)} \circ \rightleftarrows$

9 **return** $\left\{ t \in V_{\neq}^2 : \begin{pmatrix} s_1 \\ t_2 \end{pmatrix} \rightleftarrows \begin{pmatrix} t_1 \\ s_2 \end{pmatrix} \right\}$

Algorithm 4: Solving 2-DSPP in undirected graphs with non-negative edge lengths

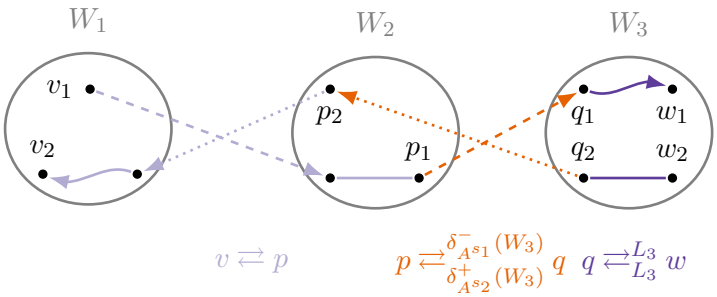


Figure 5.9: One iteration of extending the two disjoint paths in the weakly connected components W_1, W_2, W_3 of the shortest path orientation by arcs in the outer layers as in Algorithm 4. The relation $\rightleftarrows$ has already been computed on the first two components. It is extended to W_3. Two disjoint paths, a $v_1 \to w_1$ path and a $w_2 \to v_2$ path, are shown to illustrate the composition of the relations. Note that the second path is built backwards. Arcs in $A^{s_1} \setminus A^{s_2}$ are drawn dashed and arcs in $A^{s_2} \setminus A^{s_1}$ dotted. Links within the components are in $E_0 \cup (A^{s_1} \cap A^{s_2})$.

Theorem 16. *For an undirected graph* $G = (V, E)$ *with non-negative edge lengths* $\ell : E \to \mathbb{R}_{\geq 0}$ *and a pair of sources* $s \in V_{\neq}^2$, *Algorithm 4 computes the successors of* s *w.r.t.* $\overset{\ell}{=}_E$ *in polynomial time.*

Proof. Let $(W_1, L_1'), \ldots, (W_l, L_l')$ be the weakly connected components of $\overrightarrow{G}^{\text{uni}}$ sorted non-decreasingly w.r.t. $d^{s_1} - d^{s_2}$.

First observe that all steps of the algorithm can be executed in the given order. From Lemma 42 (ii) we know that the components (W_j, L'_j) are weakly acyclic and hence we can use Algorithm 3 to compute $\rightleftarrows^{L'_j}_{L'_j}$ in Line 4.

Correctness: For every $j \in \{0, \dots, l\}$ define L^1_j and L^2_j as the set of links in the subgraphs of $\overrightarrow{G}_{s_1}$ and $\overrightarrow{G}_{s_2}$ induced by the first j weakly connected components $W_1, \dots, W_j$. Further let $\rightleftarrows^j$ be the relation $\rightleftarrows$ after the j-th iteration of Line 8.

With these definitions we have in particular $L^1_0 = L^2_0 = \emptyset$ and $\rightleftarrows^0 = \mathrm{Id}\left(V^2_{\neq}\right)$. We prove the invariant

$$\rightleftarrows^j = \rightleftarrows^{L^1_j}_{L^2_j} \text{ on } V^2_{\neq} \tag{5.2}$$

by induction on j. The base case for $j = 0$ follows from the definitions. Now assume (5.2) holds for j and consider $\rightleftarrows^{j+1} = \rightleftarrows^{L'_{j+1}}_{L'_{j+1}} \circ \rightleftarrows^{\delta^-_{A^{s_1}}(V_{j+1})}_{\delta^+_{A^{s_2}}(V_{j+1})} \circ \rightleftarrows^j$.

Set $V_1 = W_1 \cup \ldots \cup W_j$ and $V_2 = W_{j+1}$. Then $V_1 \cap V_2 = \emptyset$ since W_i are the weakly connected components of $\overrightarrow{G}^{\text{uni}}$ and thus pairwise disjoint.

Further $V(L^1_j)$ and $V(L^2_j)$ are contained in V_1 since links between the j components $W_1, \dots, W_j$ stay in those components. Similarly $V(L'_{j+1}) \subseteq V_2$ as these are the links in component W_{j+1}.

Finally, since the components are sorted by a topological ordering in $\overrightarrow{G}^{\text{bi}}_{s_1}$ (Lemma 42 (iii)), $\delta^-_{A^{s_1}}(W_{j+1})$ is contained in $V_1 \times V_2$. For $\overrightarrow{G}^{\text{bi}}_{s_2}$, the components are sorted by a reverse topological ordering and hence $\delta^+_{A^{s_2}}(W_{j+1})$ is contained in $V_2 \times V_1$.

Thus all conditions of Corollary 6 are fulfilled.

$L^1_{j+1} = L^1_j \cup \delta^-_{A^{s_1}}(W_{j+1}) \cup L'_{j+1}$ and the corresponding equation for L^2_{j+1}, shows the invariant (5.2) for $j + 1$.

Since $L^1_l = E_0 \cup A^{s_1}$ and $L^2_l = E_0 \cup A^{s_2}$, we get by Observation 9 that the algorithm computes the successors of s w.r.t. $\overset{\ell}{=}_E$.

Running time: Computing the distances d^{s_1} and d^{s_2} for every node, the shortest path networks and the shortest path orientation $\overrightarrow{G}$ can be done efficiently, for example using Dijkstra's algorithm. Finding the weakly connected components $(W_1, L_1), \dots, (W_l, L_l)$ and sorting them w.r.t. $d^{s_1} - d^{s_2}$ can also be done in time polynomial in $|V|$.

Theorem 15 shows that $\rightleftarrows^{L'_j}_{L'_j}$ can be computed in polynomial time for every component. As in the proof of Theorem 15, the extension to $V^2_{\neq}$, the relations $\rightleftarrows^{\delta^-_{A^{s_1}}(W_j)}_{\delta^+_{A^{s_2}}(W_j)}$, and the composition in Line 8 can all be computed in polynomial time. □

To solve 2-DSPP, we execute Algorithm 4 and check whether (t_1, t_2) is contained in the returned set of pairs of nodes.

5.5 Conclusion

In this chapter, we give the first polynomial-time algorithm to solve 2-DSPP in graphs with non-negative edge lengths. We introduce a new class of acyclic mixed graphs and show that k-DPP can be solved efficiently in these graphs.

Future research directions are filling the gaps in Table 5.2. That is determining the complexity of k-DSPP for fixed k in undirected graphs and in directed graphs with strictly positive edge lengths. The ideas we used here of directing edges according to shortest paths networks and constructing one of the two paths backwards do not seem to be applicable to $k \geq 3$. The paths may visit the weakly connected components in any order.

In contrast to our approach of globally directing the arcs of the graph once and then searching for disjoint directed paths, the recent geometric view on the problem by Bentert et al. [Ben+21] tries different possibilities of directing parts of the graph, using the structure of parts where paths may intersect. Bentert et al. state that their approach also works for positive edge lengths. A promising direction for future research would be to see whether edges of length zero can be embedded into their framework. We have the feeling that this may result in weakly acyclic mixed graphs (as opposed to directed acyclic graphs in their original setting). Hence, our algorithm to find disjoint paths in weakly acyclic mixed graphs could be used as a subroutine.

Bibliography

[AA97] Baruch Awerbuch and Yossi Azar. "Buy-at-Bulk Network Design". In: *38th Annual Symposium on Foundations of Computer Science (FOCS 1997), 19-22 October, 1997, Miami Beach, USA*. IEEE Computer Society, 1997, pp. 542–547. DOI: 10.1109/SFCS.1997.646143.

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. ISBN: 978-0-521-42426-4.

[AFM07] Nir Andelman, Michal Feldman, and Yishay Mansour. "Strong price of anarchy". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), 7-9 January, 2007, New Orleans, USA*. Ed. by Nikhil Bansal, Kirk Pruhs, and Clifford Stein. SIAM, 2007, pp. 189–198. URL: http://dl.acm.org/citation.cfm?id=1283383.1283404.

[AFM09] Nir Andelman, Michal Feldman, and Yishay Mansour. "Strong price of anarchy". In: *Games and Economic Behavior* 65.2 (2009), pp. 289–317. DOI: 10.1016/j.geb.2008.03.005.

[Akh20] Maxim Akhmedov. "Faster 2-Disjoint-Shortest-Paths Algorithm". In: *Computer Science - Theory and Applications - 15th International Computer Science Symposium in Russia, CSR 2020, Yekaterinburg, Russia, June 29 - July 3, 2020, Proceedings*. Ed. by Henning Fernau. Vol. 12159. Lecture Notes in Computer Science. Springer, 2020, pp. 103–116. DOI: 10.1007/978-3-030-50026-9_7.

[AL10] Susanne Albers and Pascal Lenzner. "On Approximate Nash Equilibria in Network Design". In: *Internet and Network Economics - 6th International Workshop, WINE 2010, Stanford, CA, USA, December 13-17, 2010. Proceedings*. Ed. by Amin Saberi. Vol. 6484. Lecture Notes in Computer Science. Springer, 2010, pp. 14–25. DOI: 10.1007/978-3-642-17572-5_2.

[AL13] Susanne Albers and Pascal Lenzner. "On Approximate Nash Equilibria in Network Design". In: *Internet Mathematics* 9.4 (2013), pp. 384–405. DOI: 10.1080/15427951.2012.754800.

[Alb08] Susanne Albers. "On the value of coordination in network design". In: *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2008), 20-22 January, 2008, San Francisco, USA*. Ed. by Shang-Hua Teng. SIAM, 2008, pp. 294–303. URL: http://dl.acm.org/citation.cfm?id=1347082.1347115.

[Alb09] Susanne Albers. "On the Value of Coordination in Network Design". In: *SIAM Journal on Computing* 38.6 (2009), pp. 2273–2302. DOI: 10.1137/070701376.

[And04] Matthew Andrews. "Hardness of Buy-at-Bulk Network Design". In: *45th Annual Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October, 2004, Rome, Italy, Proceedings*. IEEE Computer Society, 2004, pp. 115–124. DOI: 10.1109/FOCS.2004.32.

[Ans+03] Elliot Anshelevich, Anirban Dasgupta, Éva Tardos, and Tom Wexler. "Near-optimal network design with selfish agents". In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC 2003), June 9-11, 2003, San Diego, CA, USA*. Ed. by Lawrence L. Larmore and Michel X. Goemans. ACM, 2003, pp. 511–520. DOI: 10.1145/780542.780617.

[Ans+04] Elliot Anshelevich, Anirban Dasgupta, Jon M. Kleinberg, Éva Tardos, Tom Wexler, and Tim Roughgarden. "The Price of Stability for Network Design with Fair Cost Allocation". In: *45th Annual Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October, 2004, Rome, Italy, Proceedings*. IEEE Computer Society, 2004, pp. 295–304. DOI: 10.1109/FOCS.2004.68.

[Ans+08a] Elliot Anshelevich, Anirban Dasgupta, Jon Kleinberg, Éva Tardos, Tom Wexler, and Tim Roughgarden. "The Price of Stability for Network Design with Fair Cost Allocation". In: *SIAM Journal on Computing* 38.4 (2008), pp. 1602–1623. DOI: 10.1137/070680096.

[Ans+08b] Elliot Anshelevich, Anirban Dasgupta, Éva Tardos, and Tom Wexler. "Near-Optimal Network Design with Selfish Agents". In: *Theory of Computing* 4.1 (2008), pp. 77–109. DOI: 10.4086/toc.2008.v004a004.

[ARV06] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. "On the Impact of Combinatorial Structure on Congestion Games". In: *47th Annual Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October, 2006, Berkeley, California, USA, Proceedings*. IEEE Computer Society, 2006, pp. 613–622. DOI: 10.1109/FOCS.2006.55.

[ARV08] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. "On the impact of combinatorial structure on congestion games". In: *Journal of the ACM* 55.6 (2008), 25:1–25:22. DOI: 10.1145/1455248.1455249.

[AW20] Saeed A. Amiri and Julian Wargalla. "Disjoint Shortest Paths with Congestion on DAGs". In: *CoRR* abs/2008.08368 (2020). arXiv: 2008.08368.

[BB11] Vittorio Bilò and Roberta Bove. "Bounds on the Price of stability of Undirected Network Design Games with Three Players". In: *Journal of Interconnection Networks* 12.1-2 (2011), pp. 1–17. DOI: 10.1142/S0219265911002824.

[BB95] David Borwein and Jonathan M. Borwein. "On an Intriguing Integral and Some Series Related to $\zeta(4)$". In: *Proceedings of the American Mathematical Society* 123.4 (1995), pp. 1191–1198. DOI: 10.2307/2160718.

[Ben+21] Matthias Bentert, André Nichterlein, Malte Renken, and Philipp Zschoche. "Using a Geometric Lens to Find k Disjoint Shortest Paths". In: *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*. Ed. by Nikhil Bansal, Emanuela Merelli, and James Worrell. Vol. 198. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 26:1–26:14. DOI: 10.4230/LIPIcs.ICALP.2021.26.

[BFM13] Vittorio Bilò, Michele Flammini, and Luca Moscardelli. "The Price of Stability for Undirected Broadcast Network Design with Fair Cost Allocation Is Constant". In: *54th Annual Symposium on Foundations of Computer Science (FOCS 2013) 2013, 26-29 October, 2013, Berkeley, USA*. IEEE Computer Society, 2013, pp. 638–647. DOI: 10.1109/FOCS.2013.74.

[BFM20] Vittorio Bilò, Michele Flammini, and Luca Moscardelli. "The price of stability for undirected broadcast network design with fair cost allocation is constant". In: *Games and Economic Behavior* 123 (2020), pp. 359–376. DOI: 10.1016/j.geb.2014.09.010.

[BFT17] Clark Barrett, Pascal Fontaine, and Cesare Tinelli. *The SMT-LIB Standard: Version 2.6*. Tech. rep. Available at www.SMT-LIB.org. Department of Computer Science, The University of Iowa, 2017.

[BG03] Venkatesh Bala and Sanjeev Goyal. "A Noncooperative Model of Network Formation". In: *Networks and Groups: Models of Strategic Formation*. Ed. by Bhaskar Dutta and Matthew O. Jackson. Springer Berlin Heidelberg, 2003, pp. 141–184. DOI: 10.1007/978-3-540-24790-6_7.

[BGR10] Kshipra Bhawalkar, Martin Gairing, and Tim Roughgarden. "Weighted Congestion Games: Price of Anarchy, Universal Worst-Case Examples, and Tightness". In: *Algorithms - ESA 2010, 18th Annual European Symposium, Liverpool, UK, September 6-8, 2010. Proceedings, Part II*. Ed. by Mark de Berg and Ulrich Meyer. Vol. 6347. Lecture Notes in Computer Science. Springer, 2010, pp. 17–28. DOI: 10.1007/978-3-642-15781-3_2.

[BGR14] Kshipra Bhawalkar, Martin Gairing, and Tim Roughgarden. "Weighted Congestion Games: The Price of Anarchy, Universal Worst-Case Examples, and Tightness". In: *ACM Transactions of Economics and Computation* 2.4 (Oct. 2014), 14:1–14:23. DOI: 10.1145/2629666.

[BH14] Andreas Björklund and Thore Husfeldt. "Shortest Two Disjoint Paths in Polynomial Time". In: *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*. Ed. by Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias. Vol. 8572. Lecture Notes in Computer Science. Springer, 2014, pp. 211–222. DOI: 10.1007/978-3-662-43948-7_18.

[BH19] Andreas Björklund and Thore Husfeldt. "Shortest Two Disjoint Paths in Polynomial Time". In: *SIAM Journal on Computing* 48.6 (2019), pp. 1698–1710. DOI: 10.1137/18M1223034.

[Bil+10] Vittorio Bilò, Ioannis Caragiannis, Angelo Fanelli, and Gianpiero Monaco. "Improved Lower Bounds on the Price of Stability of Undirected Network Design Games". In: *Algorithmic Game Theory - Third International Symposium, SAGT 2010, Athens, Greece, October 18-20, 2010. Proceedings*. Ed. by Spyros C. Kontogiannis, Elias Koutsoupias, and Paul G. Spirakis. Vol. 6386. Lecture Notes in Computer Science. Springer, 2010, pp. 90–101. DOI: 10.1007/978-3-642-16170-4_9.

[Bil+13] Vittorio Bilò, Ioannis Caragiannis, Angelo Fanelli, and Gianpiero Monaco. "Improved Lower Bounds on the Price of Stability of Undirected Network Design Games". In: *Theory of Computing Systems* 52.4 (2013), pp. 668–686. DOI: 10.1007/s00224-012-9411-6.

[Bil+15] Vittorio Bilò, Michele Flammini, Gianpiero Monaco, and Luca Moscardelli. "Computing Approximate Nash Equilibria in Network Congestion Games with Polynomially Decreasing Cost Functions". In: *Web and Internet Economics - 11th International Conference, WINE 2015, Amsterdam, The Netherlands, December 9-12, 2015, Proceedings*. Ed. by Evangelos Markakis and Guido Schäfer. Vol. 9470. Lecture Notes in Computer Science. Springer, 2015, pp. 118–131. DOI: 10.1007/978-3-662-48995-6_9.

[Bil+21] Vittorio Bilò, Michele Flammini, Gianpiero Monaco, and Luca Moscardelli. "Computing approximate Nash equilibria in network congestion games with polynomially decreasing cost functions". In: *Distributed Computing* 34.1 (2021), pp. 1–14. DOI: 10.1007/s00446-020-00381-4.

[BK17] Kristóf Bérczi and Yusuke Kobayashi. "The Directed Disjoint Shortest Paths Problem". In: *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*. Ed. by Kirk Pruhs and Christian Sohler. Vol. 87. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 13:1–13:13. DOI: 10.4230/LIPIcs.ESA.2017.13.

[Bra68] Dietrich Braess. "Über ein Paradoxon aus der Verkehrsplanung". In: *Unternehmensforschung* 12.1 (Dec. 1968), pp. 258–268. DOI: 10.1007/BF01918335.

[BS07] Cüneyt F. Bazlamaçci and Fatih Say. "Minimum concave cost multicommodity network design". In: *Telecommunication Systems* 36.4 (2007), pp. 181–203. DOI: 10.1007/s11235-008-9068-2.

[Car+11] Ioannis Caragiannis, Angelo Fanelli, Nick Gravin, and Alexander Skopalik. "Efficient Computation of Approximate Pure Nash Equilibria in Congestion Games". In: *Proceedings of the 52nd Annual Symposium on Foundations of Computer Science (FOCS 2011), 23-25 October, 2011, Palm Springs, USA*. 2011, pp. 532–541. DOI: 10.1109/focs.2011.50.

[Car+15] Ioannis Caragiannis, Angelo Fanelli, Nick Gravin, and Alexander Skopalik. "Approximate Pure Nash Equilibria in Weighted Congestion Games: Existence, Efficient Computation, and Structure". In: *ACM Transactions of Economics and Computation* 3.1 (Mar. 2015), 2:1–2:32. DOI: 10.1145/2614687.

[CF19] Ioannis Caragiannis and Angelo Fanelli. "On Approximate Pure Nash Equilibria in Weighted Congestion Games with Polynomial Latencies". In: *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 133:1–133:12. DOI: 10.4230/LIPIcs.ICALP.2019.133.

[CF21] Ioannis Caragiannis and Angelo Fanelli. "On approximate pure Nash equilibria in weighted congestion games with polynomial latencies". In: *Journal of Computer and System Sciences* 117 (2021), pp. 40–48. DOI: 10.1016/j.jcss.2020.10.007.

[CG13] George Christodoulou and Martin Gairing. "Price of Stability in Polynomial Congestion Games". In: *40th International Colloquium on Automata, Languages, and Programming, ICALP 2013, July 8-12, 2013, Riga, Latvia, Proceedings, Part II*. Ed. by Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg. Vol. 7966. Lecture Notes in Computer Science. Springer, 2013, pp. 496–507. DOI: 10.1007/978-3-642-39212-2_44.

[CG16] George Christodoulou and Martin Gairing. "Price of Stability in Polynomial Congestion Games". In: *ACM Transactions of Economics and Computation* 4.2 (Dec. 2016), 10:1–10:17. DOI: 10.1145/2841229.

[Che+06] Chandra Chekuri, Julia Chuzhoy, Liane Lewin-Eytan, Joseph Naor, and Ariel Orda. "Non-cooperative multicast and facility location games". In: *Proceedings of the 7th ACM Conference on Electronic Commerce, EC 2006, Ann Arbor, Michigan, USA, June 11-15, 2006*. Ed. by Joan Feigenbaum, John C.-I. Chuang, and David M. Pennock. ACM, 2006, pp. 72–81. DOI: 10.1145/1134707.1134716.

[Che+07] Chandra Chekuri, Julia Chuzhoy, Liane Lewin-Eytan, Joseph Naor, and Ariel Orda. "Non-Cooperative Multicast and Facility Location Games". In: *IEEE Journal on Selected Areas in Communications* 25.6 (2007), pp. 1193–1206. DOI: 10.1109/JSAC.2007.070813.

[Che+10] Chandra Chekuri, Mohammad Taghi Hajiaghayi, Guy Kortsarz, and Mohammad R. Salavatipour. "Approximation Algorithms for Nonuniform Buy-at-Bulk Network Design". In: *SIAM Journal on Computing* 39.5 (2010), pp. 1772–1798. DOI: 10.1137/090750317.

[Chr+09] George Christodoulou, Christine Chung, Katrina Ligett, Evangelia Pyrga, and Rob van Stee. "On the Price of Stability for Undirected Network Design". In: *Approximation and Online Algorithms, 7th International Workshop, WAOA 2009, Copenhagen, Denmark, September 10-11, 2009. Revised Papers*. Ed. by Evripidis Bampis and Klaus Jansen. Vol. 5893. Lecture Notes in Computer Science. Springer, 2009, pp. 86–97. DOI: 10.1007/978-3-642-12450-1_8.

[Chr+18] George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, and Paul G. Spirakis. "The Price of Stability of Weighted Congestion Games". In: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. Ed. by Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella. Vol. 107. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 150:1–150:16. DOI: 10.4230/LIPIcs.ICALP.2018.150.

[Chr+19] George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, and Paul G. Spirakis. "The Price of Stability of Weighted Congestion Games". In: *SIAM Journal on Computing* 48.5 (2019), pp. 1544–1582. DOI: 10.1137/18M1207880.

[Chr+20] George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, Diogo Poças, and Clara Waldmann. "Existence and Complexity of Approximate Equilibria in Weighted Congestion Games". In: *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*. Ed. by Artur Czumaj, Anuj Dawar, and Emanuela Merelli. Vol. 168. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 32:1–32:18. DOI: 10.4230/LIPIcs.ICALP.2020.32.

[CKN01] Chandra Chekuri, Sanjeev Khanna, and Joseph Naor. "A deterministic algorithm for the cost-distance problem". In: *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms (SODA 2001), January 7-9, 2001, Washington, DC, USA*. Ed. by S. Rao Kosaraju. ACM/SIAM, 2001, pp. 232–233. URL: http://dl.acm.org/citation.cfm?id=365411.365452.

[CKS09] George Christodoulou, Elias Koutsoupias, and Paul G. Spirakis. "On the Performance of Approximate Equilibria in Congestion Games". In: *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*. Ed. by Amos Fiat and Peter Sanders. Vol. 5757. Lecture Notes in Computer Science. Springer, 2009, pp. 251–262. DOI: 10.1007/978-3-642-04128-0_22.

[CKS11] George Christodoulou, Elias Koutsoupias, and Paul G. Spirakis. "On the Performance of Approximate Equilibria in Congestion Games". In: *Algorithmica* 61.1 (2011), pp. 116–140. DOI: 10.1007/s00453-010-9449-2.

[Cou38] Antoine A. Cournot. *Recherches sur les Principes Mathématiques de la Théorie des Richesses*. English translation: *Researches into the Mathematical Principles of the Theory of Wealth* by Nathaniel T. Bacon, Macmillan, New York, 1927. Paris: Hachette, 1838.

[CR06] Ho-Lin Chen and Tim Roughgarden. "Network design with weighted players". In: *SPAA 2006: Proceedings of the 18th Annual ACM Symposium on Parallelism in Algorithms and Architectures, Cambridge, Massachusetts, USA, July 30 - August 2, 2006*. Ed. by Phillip B. Gibbons and Uzi Vishkin. ACM, 2006, pp. 29–38. DOI: 10.1145/1148109.1148114.

[CR09] Ho-Lin Chen and Tim Roughgarden. "Network Design with Weighted Players". In: *Theory of Computing Systems* 45.2 (2009), pp. 302–324. DOI: 10.1007/s00224-008-9128-8.

[CS07] Steve Chien and Alistair Sinclair. "Convergence to approximate Nash equilibria in congestion games". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), 7-9 January, 2007, New Orleans, USA*. Ed. by Nikhil Bansal, Kirk Pruhs, and Clifford Stein. SIAM, 2007, pp. 169–178. URL: http://dl.acm.org/citation.cfm?id=1283383.1283402.

[CS08] Vincent Conitzer and Tuomas Sandholm. "New complexity results about Nash equilibria". In: *Games and Economic Behavior* 63.2 (2008), pp. 621–641. DOI: 10.1016/j.geb.2008.02.015.

[CS11] Steve Chien and Alistair Sinclair. "Convergence to approximate Nash equilibria in congestion games". In: *Games and Economic Behavior* 71.2 (2011), pp. 315–327. DOI: 10.1016/j.geb.2009.05.004.

[CY16] Leizhen Cai and Junjie Ye. "Finding Two Edge-Disjoint Paths with Length Constraints". In: *Graph-Theoretic Concepts in Computer Science, 42nd International Workshop, WG 2016, Istanbul, Turkey, June 22-24, 2016, Revised Selected Papers*. Ed. by Pinar Heggernes. Vol. 9941. Lecture Notes in Computer Science. 2016, pp. 62–73. DOI: 10.1007/978-3-662-53536-3_6.

[Dan90] George B. Dantzig. "Origins of the Simplex Method". In: *A History of Scientific Computing*. ACM, 1990, pp. 141–151. DOI: 10.1145/87252.88081.

[Die17] Reinhard Diestel. *Graph theory*. Fifth edition. Graduate texts in mathematics. Berlin: Springer, 2017. ISBN: 9783662536223.

[Dis+15] Yann Disser, Andreas E. Feldmann, Max Klimm, and Matúš Mihalák. "Improving the Hk-bound on the price of stability in undirected Shapley network design games". In: *Theoretical Computer Science* 562 (2015), pp. 557–564. DOI: 10.1016/j.tcs.2014.10.037.

[DS06] Juliane Dunkel and Andreas S. Schulz. "On the Complexity of Pure-Strategy Nash Equilibria in Congestion and Local-Effect Games". In: *Internet and Network Economics, Second International Workshop, WINE 2006, Patras, Greece, December 15-17, 2006, Proceedings*. Ed. by Paul G. Spirakis, Marios Mavronicolas, and Spyros C. Kontogiannis. Vol. 4286. Lecture Notes in Computer Science. Springer, 2006, pp. 62–73. DOI: 10.1007/11944874_7.

[DS08] Juliane Dunkel and Andreas S. Schulz. "On the Complexity of Pure-Strategy Nash Equilibria in Congestion and Local-Effect Games". In: *Mathematics of Operations Research* 33.4 (2008), pp. 851–868. DOI: 10.1287/moor.1080.0322.

[Eil98] Tali Eilam-Tzoreff. "The disjoint shortest paths problem". In: *Discrete Applied Mathematics* 85.2 (1998), pp. 113–138. DOI: 10.1016/S0166-218X(97)00121-2.

[EIS76] Shimon Even, Alon Itai, and Adi Shamir. "On the Complexity of Timetable and Multicommodity Flow Problems". In: *SIAM Journal on Computing* 5.4 (1976), pp. 691–703. DOI: 10.1137/0205048.

[EMV87] Ranel E. Erickson, Clyde L. Monma, and Arthur F. Veinott. "Send-and-Split Method for Minimum-Concave-Cost Network Flows". In: *Mathematics of Operations Research* 12.4 (1987), pp. 634–664. DOI: 10.1287/moor.12.4.634.

[Fab+03] Alex Fabrikant, Ankur Luthra, Elitza N. Maneva, Christos H. Papadimitriou, and Scott Shenker. "On a network creation game". In: *Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston, Massachusetts, USA, July 13-16, 2003*. Ed. by Elizabeth Borowsky and Sergio Rajsbaum. ACM, 2003, pp. 347–351. DOI: 10.1145/872035.872088.

[Fel+17] Matthias Feldotto, Martin Gairing, Grammateia Kotsialou, and Alexander Skopalik. "Computing Approximate Pure Nash Equilibria in Shapley Value Weighted Congestion Games". In: *Web and Internet Economics - 13th International Conference, WINE 2017, Bangalore, India, December 17-20, 2017, Proceedings*. Ed. by Nikhil R. Devanur and Pinyan Lu. Vol. 10660. Lecture Notes in Computer Science. Springer, 2017, pp. 191–204. DOI: 10.1007/978-3-319-71924-5_14.

[FG07] Dalila B. M. M. Fontes and José F. Gonçalves. "Heuristic solutions for general concave minimum cost network flow problems". In: *Networks* 50.1 (2007), pp. 67–76. DOI: 10.1002/net.20167.

[FHP16] Rupert Freeman, Samuel Haney, and Debmalya Panigrahi. "On the Price of Stability of Undirected Multicast Games". In: *Web and Internet Economics - 12th International Conference, WINE 2016, Montreal, Canada, December 11-14, 2016, Proceedings*. Ed. by Yang Cai and Adrian Vetta. Vol. 10123.

Lecture Notes in Computer Science. Springer, 2016, pp. 354–368. DOI: 10.1007/978-3-662-54110-4_25.

[FHW80] Steven Fortune, John Hopcroft, and James Wyllie. "The directed subgraph homeomorphism problem". In: *Theoretical Computer Science* 10.2 (1980), pp. 111–121. DOI: 10.1016/0304-3975(80)90009-2.

[Fia+06] Amos Fiat, Haim Kaplan, Meital Levy, Svetlana Olonetsky, and Ronen Shabo. "On the Price of Stability for Designing Undirected Networks with Fair Cost Allocations". In: *33rd International Colloquium on Automata, Languages, and Programming, ICALP 2006, July 10-14, 2006, Venice, Italy, Proceedings, Part I*. Ed. by Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener. Vol. 4051. Lecture Notes in Computer Science. Springer, 2006, pp. 608–618. DOI: 10.1007/11786986_53.

[FKS04] Dimitris Fotakis, Spyros C. Kontogiannis, and Paul G. Spirakis. "Selfish Unsplittable Flows". In: *31st International Colloquium on Automata, Languages, and Programming, ICALP 2004, July 12-16, 2004, Turku, Finland, Proceedings*. Ed. by Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella. Vol. 3142. Lecture Notes in Computer Science. Springer, 2004, pp. 593–605. DOI: 10.1007/978-3-540-27836-8_51.

[FKS05] Dimitris Fotakis, Spyros C. Kontogiannis, and Paul G. Spirakis. "Selfish unsplittable flows". In: *Theoretical Computer Science* 348.2 (2005), pp. 226–239. DOI: 10.1016/j.tcs.2005.09.024.

[Fot+02] Dimitris Fotakis, Spyros C. Kontogiannis, Elias Koutsoupias, Marios Mavronicolas, and Paul G. Spirakis. "The Structure and Complexity of Nash Equilibria for a Selfish Routing Game". In: *29th International Colloquium on Automata, Languages, and Programming, ICALP 2002, July 8-13, 2002, Malaga, Spain, Proceedings*. Ed. by Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan J. Eidenbenz, and Ricardo Conejo. Vol. 2380. Lecture Notes in Computer Science. Springer, 2002, pp. 123–134. DOI: 10.1007/3-540-45465-9_12.

[Fot+09] Dimitris Fotakis, Spyros C. Kontogiannis, Elias Koutsoupias, Marios Mavronicolas, and Paul G. Spirakis. "The structure and complexity of Nash equilibria for a selfish routing game". In: *Theoretical Computer Science* 410.36 (2009), pp. 3305–3326. DOI: 10.1016/j.tcs.2008.01.004.

[FPT04] Alex Fabrikant, Christos H. Papadimitriou, and Kunal Talwar. "The complexity of pure Nash equilibria". In: *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC 2004), Chicago, IL, USA, June 13-16, 2004*. Ed. by László Babai. ACM, 2004, pp. 604–612. DOI: 10.1145/1007352.1007445.

[FR12] Michal Feldman and Tom Ron. "Capacitated Network Design Games". In: *Algorithmic Game Theory - 5th International Symposium, SAGT 2012, Barcelona, Spain, October 22-23, 2012. Proceedings*. Ed. by Maria J. Serna. Vol. 7615. Lecture Notes in Computer Science. Springer, 2012, pp. 132–143. DOI: 10.1007/978-3-642-33996-7_12.

[FR15] Michal Feldman and Tom Ron. "Capacitated Network Design Games". In: *Theory of Computing Systems* 57.3 (2015), pp. 576–597. DOI: 10.1007/s00224-014-9540-1.

[FRT03] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. "A tight bound on approximating arbitrary metrics by tree metrics". In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC 2003), June 9-11, 2003, San Diego, CA, USA*. Ed. by Lawrence L. Larmore and Michel X. Goemans. ACM, 2003, pp. 448–455. DOI: 10.1145/780542.780608.

[FRT04] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. "A tight bound on approximating arbitrary metrics by tree metrics". In: *Journal of Computer and System Sciences* 69.3 (2004), pp. 485–497. DOI: 10.1016/j.jcss.2004.04.011.

[GI06] Fabrizio Grandoni and Giuseppe F. Italiano. "Improved Approximation for Single-Sink Buy-at-Bulk". In: *Algorithms and Computation, 17th International Symposium, ISAAC 2006, Kolkata, India, December 18-20, 2006, Proceedings*. Ed. by Tetsuo Asano. Vol. 4288. Lecture Notes in Computer Science. Springer, 2006, pp. 111–120. DOI: 10.1007/11940128_13.

[GJ90] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co., 1990. ISBN: 0716710455.

[GKR03] Anupam Gupta, Amit Kumar, and Tim Roughgarden. "Simpler and better approximation algorithms for network design". In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC 2003), June 9-11, 2003, San Diego, CA, USA*. Ed. by Lawrence L. Larmore and Michel X. Goemans. ACM Association for Computing Machinery, 2003, pp. 365–372. DOI: 10.1145/780542.780597.

[GKW19] Marinus Gottschau, Marcus Kaiser, and Clara Waldmann. "The undirected two disjoint shortest paths problem". In: *Operations Research Letters* 47.1 (2019), pp. 70–75. DOI: 10.1016/j.orl.2018.11.011.

[GMM01] Sudipto Guha, Adam Meyerson, and Kamesh Munagala. "A constant factor approximation for the single sink edge installation problems". In: *Proceedings on 33rd Annual ACM Symposium on Theory of Computing (STOC 2001), July 6-8, 2001, Heraklion, Crete, Greece*. Ed. by Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis. ACM, 2001, pp. 383–388. DOI: 10.1145/380752.380827.

[GMV05] Michel X. Goemans, Vahab S. Mirrokni, and Adrian Vetta. "Sink equilibria and convergence". In: *Proceedings of the 46th Annual Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October, 2005, Pittsburgh, USA*. 2005, pp. 142–151. DOI: 10.1109/SFCS.2005.68.

[GNS21] Yiannis Giannakopoulos, Georgy Noarov, and Andreas S. Schulz. "Computing Approximate Equilibria in Weighted Congestion Games via Best-Responses". In: *Mathematics of Operations Research* (Sept. 2021). DOI: 10.1287/moor.2021.1144.

[GP20] Yiannis Giannakopoulos and Diogo Poças. "A Unifying Approximate Potential for Weighted Congestion Games". In: *Algorithmic Game Theory - 13th International Symposium, SAGT 2020, Augsburg, Germany, September 16-18, 2020, Proceedings*. Ed. by Tobias Harks and Max Klimm. Vol. 12283. Lecture Notes in Computer Science. Springer, 2020, pp. 99–113. DOI: 10.1007/978-3-030-57980-7_7.

[GP91a] Geoffrey M. Guisewite and Panos M. Pardalos. "Algorithms for the single-source uncapacitated minimum concave-cost network flow problem". In: *Journal of Global Optimization* 1.3 (1991), pp. 245–265. DOI: 10.1007/BF00119934.

[GP91b] Geoffrey M. Guisewite and Panos M. Pardalos. "Performance of Local Search In Minimum Concave-Cost Network Flow Problems:" in: *Recent Advances in Global Optimization*. Princeton University Press, 1991, pp. 50–75. DOI: doi:10.1515/9781400862528.50.

[GP93] Geoffrey M. Guisewite and Panos M. Pardalos. "A polynomial time solvable concave network flow problem". In: *Networks* 23.2 (1993), pp. 143–147. DOI: 10.1002/net.3230230208.

[GR10] Fabrizio Grandoni and Thomas Rothvoß. "Network Design via Core Detouring for Problems without a Core". In: *37th International Colloquium on Automata, Languages, and Programming, ICALP 2010, July 6-10, 2010, Bordeaux, France, Proceedings, Part I*. Ed. by Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis. Vol. 6198. Lecture Notes in Computer Science. Springer, 2010, pp. 490–502. DOI: 10.1007/978-3-642-14165-2_42.

[Gui95] Geoffrey M. Guisewite. "Network Problems". In: *Handbook of Global Optimization*. Ed. by Reiner Horst and Panos M. Pardalos. Boston, MA: Springer US, 1995, pp. 609–648. ISBN: 978-1-4615-2025-2. DOI: 10.1007/978-1-4615-2025-2_11.

[Gün17] Henning Günther. *smtlib2: A type-safe interface to communicate with an SMT solver*. 2017. URL: https://hackage.haskell.org/package/smtlib2.

[Gup+07] Anupam Gupta, Amit Kumar, Martin Pál, and Tim Roughgarden. "Approximation via cost sharing: Simpler and better approximation algorithms for network design". In: *Journal of the ACM* 54.3 (2007), p. 11. DOI: 10.1145/1236457.1236458.

[Has05] Mehdi Hassani. "Approximation of the Lambert W Function". In: *Research Report Collection 8(4), Victoria University, Melbourn* (2005).

[HK10] Tobias Harks and Max Klimm. "On the Existence of Pure Nash Equilibria in Weighted Congestion Games". In: *37th International Colloquium on Automata, Languages, and Programming, ICALP 2010, July 6-10, 2010, Bordeaux, France, Proceedings, Part I*. Ed. by Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis. Vol. 6198. Lecture Notes in Computer Science. Springer, 2010, pp. 79–89. DOI: 10.1007/978-3-642-14165-2_8.

[HK12] Tobias Harks and Max Klimm. "On the Existence of Pure Nash Equilibria in Weighted Congestion Games". In: *Mathematics of Operations Research* 37.3 (2012), pp. 419–436. DOI: 10.1287/moor.1120.0543.

[HKM09] Tobias Harks, Max Klimm, and Rolf H. Möhring. "Characterizing the Existence of Potential Functions in Weighted Congestion Games". In: *Algorithmic Game Theory – Second International Symposium, SAGT 2009, Paphos, Cyprus, October 18-20, 2009. Proceedings*. Ed. by Marios Mavronicolas and Vicky G. Papadopoulou. Vol. 5814. Lecture Notes in Computer Science. Springer, 2009, pp. 97–108. DOI: 10.1007/978-3-642-04645-2_10.

[HKM11] Tobias Harks, Max Klimm, and Rolf H. Möhring. "Characterizing the Existence of Potential Functions in Weighted Congestion Games". In: *Theory of Computing Systems* 49.1 (July 2011), pp. 46–70. DOI: 10.1007/s00224-011-9315-x.

[HKS14] Christoph Hansknecht, Max Klimm, and Alexander Skopalik. "Approximate Pure Nash Equilibria in Weighted Congestion Games". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014, Barcelona, Spain*. Ed. by Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore. Vol. 28. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014, pp. 242–257. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2014.242.

[Hyv+16] Antti E. J. Hyvärinen, Matteo Marescotti, Leonardo Alt, and Natasha Sharygina. "OpenSMT2: An SMT Solver for Multi-core and Cloud Computing". In: *Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings*. Ed. by Nadia Creignou and Daniel Le Berre. Vol. 9710. Lecture Notes in Computer Science. Springer, 2016, pp. 547–553. DOI: 10.1007/978-3-319-40970-2_35. URL: https://github.com/usi-verification-and-security/opensmt.

[Inc21] OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences*. 2021. URL: http://oeis.org/A000081.

[JPY88] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. "How easy is local search?" In: *Journal of Computer and System Sciences* 37.1 (1988), pp. 79–100. DOI: 10.1016/0022-0000(88)90046-3.

[JR04] Raja Jothi and Balaji Raghavachari. "Improved Approximation Algorithms for the Single-Sink Buy-at-Bulk Network Design Problems". In: *Algorithm Theory - SWAT 2004, 9th Scandinavian Workshop on Algorithm Theory, Humlebaek, Denmark, July 8-10, 2004, Proceedings*. Ed. by Torben Hagerup and Jyrki Katajainen. Vol. 3111. Lecture Notes in Computer Science. Springer, 2004, pp. 336–348. DOI: 10.1007/978-3-540-27810-8_29.

[JR09] Raja Jothi and Balaji Raghavachari. "Improved approximation algorithms for the single-sink buy-at-bulk network design problems". In: *Journal of Discrete Algorithms* 7.2 (2009). Selected papers from the 2nd Algorithms and Complexity in Durham Workshop ACiD 2006, pp. 249–255. DOI: j.jda.2008.12.003.

[Kar72] Richard M. Karp. "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*. Ed. by Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger. Boston, MA: Springer US, 1972, pp. 85–103. ISBN: 978-1-4684-2001-2. DOI: 10.1007/978-1-4684-2001-2_9.

[Kha80] Leonid G. Khachiyan. "Polynomial algorithms in linear programming". In: *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), pp. 53–72. DOI: 10.1016/0041-5553(80)90061-0.

[KKR12] Ken-ichi Kawarabayashi, Yusuke Kobayashi, and Bruce Reed. "The disjoint paths problem in quadratic time". In: *Journal of Combinatorial Theory, Series B* 102.2 (2012), pp. 424–435. DOI: 10.1016/j.jctb.2011.07.004.

[KM12] Yasushi Kawase and Kazuhisa Makino. "Nash Equilibria with Minimum Potential in Undirected Broadcast Games". In: *WALCOM: Algorithms and Computation - 6th International Workshop, WALCOM 2012, Dhaka, Bangladesh, February 15-17, 2012. Proceedings*. Ed. by Md. Saidur Rahman and Shin-Ichi Nakano. Vol. 7157. Lecture Notes in Computer Science. Springer, 2012, pp. 217–228. DOI: 10.1007/978-3-642-28076-4_22.

[KM13] Yasushi Kawase and Kazuhisa Makino. "Nash equilibria with minimum potential in undirected broadcast games". In: *Theoretical Computer Science* 482 (2013), pp. 33–47. DOI: 10.1016/j.tcs.2013.02.031.

[KM72] Victor Klee and George J. Minty. "How good is the simplex algorithm?" In: *Inequalities III, Proceedings of the Third Symposium on Inequalities held at the University of California, Los Angeles, Calif., September 1–9, 1969, dedicated to the memory of Theodore S. Motzkin*. Ed. by Oved Shisha. Academic Press, 1972, pp. 159–175.

[KP99] Elias Koutsoupias and Christos H. Papadimitriou. "Worst-case Equilibria". In: *STACS 99, 16th Annual Symposium on Theoretical Aspects of Computer Science, Trier, Germany, March 4-6, 1999, Proceedings*. Ed. by Christoph Meinel and Sophie Tison. Vol. 1563. Lecture Notes in Computer Science. Springer, 1999, pp. 404–413. DOI: 10.1007/3-540-49116-3_38.

[KR11] Konstantinos Kollias and Tim Roughgarden. "Restoring Pure Equilibria to Weighted Congestion Games". In: *38th International Colloquium on Automata, Languages, and Programming, ICALP 2011, July 4-8, 2011, Zurich, Switzerland, Proceedings, Part II*. Ed. by Luca Aceto, Monika Henzinger, and Jirí Sgall. Vol. 6756. Lecture Notes in Computer Science. Springer, 2011, pp. 539–551. DOI: 10.1007/978-3-642-22012-8_43.

[KR15] Konstantinos Kollias and Tim Roughgarden. "Restoring Pure Equilibria to Weighted Congestion Games". In: *ACM Transactions of Economics and Computation* 3.4 (2015), 21:1–21:24. DOI: 10.1145/2781678.

[KS16] Daniel Kroening and Ofer Strichman. *Decision Procedures - An Algorithmic Point of View, Second Edition*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016. ISBN: 978-3-662-50496-3. DOI: 10.1007/978-3-662-50497-0.

[KS19] Yusuke Kobayashi and Ryo Sako. "Two disjoint shortest paths problem with non-negative edge length". In: *Operations Research Letters* 47.1 (2019), pp. 66–69. DOI: 10.1016/j.orl.2018.11.012.

[Li09] Jian Li. "An O(log n / log log n) upper bound on the price of stability for undirected Shapley network design games". In: *Information Processing Letters* 109.15 (2009), pp. 876–878. DOI: 10.1016/j.ipl.2009.04.015.

[LL13] Euiwoong Lee and Katrina Ligett. "Improved bounds on the price of stability in network cost sharing games". In: *Proceedings of the fourteenth ACM Conference on Electronic Commerce, EC 2013, Philadelphia, PA, USA, June 16-20, 2013*. Ed. by Michael J. Kearns, R. Preston McAfee, and Éva Tardos. ACM, 2013, pp. 607–620. DOI: 10.1145/2492002.2482562.

[LMS90] Chung-Lun Li, S.Thomas McCormick, and David Simchi-Levi. "The complexity of finding two disjoint paths with min-max objective function". In: *Discrete Applied Mathematics* 26.1 (1990), pp. 105–115. DOI: 10.1016/0166-218X(90)90024-7.

[LO01] Lavy Libman and Ariel Orda. "Atomic Resource Sharing in Noncooperative Networks". In: *Telecommunication Systems* 17.4 (Aug. 2001), pp. 385–409. DOI: 10.1023/A:1016770831869.

[LO97] Lavy Libman and Ariel Orda. "Atomic Resource Sharing in Noncooperative Networks". In: *Proceedings IEEE INFOCOM '97, The Conference on Computer Communications, Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Driving the Information Revolution, Kobe, Japan, April 7-12, 1997*. IEEE Computer Society, 1997, pp. 1006–1013. DOI: 10.1109/INFCOM.1997.631115.

[Loc21] William Lochet. "A Polynomial Time Algorithm for the k-Disjoint Shortest Paths Problem". In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA 2021), 10-13 January, 2021, Virtual Conference*. Ed. by Dániel Marx. SIAM, 2021, pp. 169–178. DOI: 10.1137/1.9781611976465.12.

[LST12] Renato Paes Leme, Vasilis Syrgkanis, and Éva Tardos. "The curse of simultaneity". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*. Ed. by Shafi Goldwasser. ACM, 2012, pp. 60–67. DOI: 10.1145/2090236.2090242.

[Lyn75] James F. Lynch. "The Equivalence of Theorem Proving and the Interconnection Problem". In: *SIGDA Newsletter* 5.3 (Sept. 1975), pp. 31–36. DOI: 10.1145/1061425.1061430.

[Mar10] Simon Marlow. *Haskell 2010 Language Report*. 2010. URL: https://www.haskell.org/definition/haskell2010.pdf.

[Men27] Karl Menger. "Zur allgemeinen Kurventheorie". In: *Fundamenta Mathematicae* 10.1 (1927), pp. 96–115. DOI: 10.4064/fm-10-1-96-115.

[MFF13] Marta S. R. Monteiro, Dalila B. M. M. Fontes, and Fernando A. C. C. Fontes. "Concave minimum cost network flow problems solved with a colony of ants". In: *Journal of Heuristics* 19.1 (2013), pp. 1–33. DOI: 10.1007/s10732-012-9214-6.

[Mil96] Igal Milchtaich. "Congestion Games with Player-Specific Payoff Functions". In: *Games and Economic Behavior* 13.1 (1996), pp. 111–124. DOI: 10.1006/game.1996.0027.

[MMM18] Akaki Mamageishvili, Matúš Mihalák, and Simone Montemezzani. "Improved bounds on equilibria solutions in the network design game". In: *International Journal of Game Theory* 47.4 (2018), pp. 1113–1135. DOI: 10.1007/s00182-017-0600-z.

[MMP00] Adam Meyerson, Kamesh Munagala, and Serge A. Plotkin. "Cost-Distance: Two Metric Network Design". In: *41st Annual Symposium on Foundations of Computer Science (FOCS 2000), 12-14 November, 2000, Redondo Beach, USA*. IEEE Computer Society, 2000, pp. 624–630. DOI: 10.1109/SFCS.2000.892330.

[MP12] Simon Marlow and Simon Peyton Jones. "The Glasgow Haskell Compiler". In: *The Architecture of Open Source Applications, Volume 2*. The Architecture of Open Source Applications, Volume 2. Lulu, Jan. 2012. URL: https://www.microsoft.com/en-us/research/publication/the-glasgow-haskell-compiler/.

[MS96] Dov Monderer and Lloyd S. Shapley. "Potential Games". In: *Games and Economic Behavior* 14.1 (1996), pp. 124–143. DOI: 10.1006/game.1996.0044.

[Nas50] John F. Nash. "Equilibrium points in n-person games". In: *Proceedings of the National Academy of Sciences* 36.1 (1950), pp. 48–49. DOI: 10.1073/pnas.36.1.48.

[Neu28] John von Neumann. "Zur Theorie der Gesellschaftsspiele". In: *Mathematische Annalen* 100.1 (Dec. 1928), pp. 295–320. DOI: 10.1007/BF01448847.

[NM47] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1947. URL: http://press.princeton.edu/titles/7802.html.

[Oht80] Tatsuo Ohtsuki. "The two disjoint path problem and wire routing design". In: *Graph Theory and Algorithms, 17th Symposium of Research Institute of Electric Communication, Tohoku University, Sendai, Japan, October 24-25, 1980, Proceedings*. Ed. by Nobuji Saito and Takao Nishizeki. Vol. 108. Lecture Notes in Computer Science. Springer, 1980, pp. 207–216. DOI: 10.1007/3-540-10704-5_18.

[Olv+10] Frank W. J. Olver, Daniel W. Lozier, Ronald F. Boisvert, and Charles W. Clark. *NIST Handbook of Mathematical Functions*. Cambridge University Press, 2010. URL: http://dlmf.nist.gov/4.5.E13.

[Pap01] Christos H. Papadimitriou. "Algorithms, Games, and the Internet". In: *28th International Colloquium on Automata, Languages, and Programming, ICALP 2001, July 8-12, 2001, Crete, Greece, Proceedings*. Ed. by Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen. Vol. 2076. Lecture Notes in Computer Science. Springer, 2001, pp. 1–3. DOI: 10.1007/3-540-48224-5_1.

[Pap94] Christos H. Papadimitriou. "Computational Complexity". In: Addison-Wesley, 1994. Chap. 18. ISBN: 0201530821.

[Pig20] Arthur C. Pigou. *The Economics of Welfare*. London: Macmillan and Co., Limited, 1920.

[PS07] Panagiota N. Panagopoulou and Paul G. Spirakis. "Algorithms for pure Nash equilibria in weighted congestion games". In: *Journal of Experimental Algorithmics* 11 (Feb. 2007), p. 27. DOI: 10.1145/1187436.1216584.

[Ree+91] Bruce A. Reed, Neil Robertson, Alexander Schrijver, and Paul D. Seymour. "Finding disjoint trees in planar graphs in linear time". In: *Graph Structure Theory, Proceedings of a AMS-IMS-SIAM Joint Summer Research Conference on Graph Minors held June 22 to July 5, 1991, at the University of Washington, Seattle, USA*. Ed. by Neil Robertson and Paul D. Seymour. Vol. 147. Contemporary Mathematics. American Mathematical Society, 1991, pp. 295–301.

[Ros73] Robert W. Rosenthal. "A class of games possessing pure-strategy Nash equilibria". In: *International Journal of Game Theory* 2 (1973), pp. 65–67. DOI: 10.1007/BF01737559.

[Rou09] Tim Roughgarden. "Intrinsic robustness of the price of anarchy". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009) , Bethesda, MD, USA, May 31 - June 2, 2009*. Ed. by Michael Mitzenmacher. ACM, 2009, pp. 513–522. DOI: 10.1145/1536414.1536485.

[Rou12] Tim Roughgarden. "Intrinsic robustness of the price of anarchy". In: *Communications of the ACM* 55.7 (2012), pp. 116–123. DOI: 10.1145/2209249.2209274.

[Rou15] Tim Roughgarden. "Intrinsic Robustness of the Price of Anarchy". In: *Journal of the ACM* 62.5 (2015), 32:1–32:42. DOI: 10.1145/2806883.

[RS85] Neil Robertson and Paul D. Seymour. "Disjoint Paths—A Survey". In: *SIAM Journal on Algebraic Discrete Methods* 6.2 (1985), pp. 300–305. DOI: 10.1137/0606030.

[RS95] Neil Robertson and Paul D. Seymour. "Graph Minors .XIII. The Disjoint Paths Problem". In: *Journal of Combinatorial Theory, Series B* 63.1 (1995), pp. 65–110. DOI: 10.1006/jctb.1995.1006.

[Sal+97] F. Sibel Salman, Joseph Cheriyan, R. Ravi, and S. Subramanian. "Buy-at-Bulk Network Design: Approximating the Single-Sink Edge Installation Problem". In: *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1997), 5-7 January 1997, New Orleans, Louisiana, USA*. Ed. by Michael E. Saks. ACM/SIAM, 1997, pp. 619–628. URL: http://dl.acm.org/citation.cfm?id=314161.314397.

[Sch00] Alexander Schrijver. *Theory of linear and integer programming*. Reprinted. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 2000. ISBN: 0471982326.

[Sch94] Alexander Schrijver. "Finding k Disjoint Paths in a Directed Planar Graph". In: *SIAM Journal on Computing* 23.4 (1994), pp. 780–788. DOI: 10.1137/S0097539792224061.

[Sey80] Paul D. Seymour. "Disjoint paths in graphs". In: *Discrete Mathematics* 29.3 (1980), pp. 293–309. DOI: 10.1016/0012-365X(80)90158-2.

[Shi80] Yossi Shiloach. "A Polynomial Solution to the Undirected Two Paths Problem". In: *Journal of the ACM* 27.3 (July 1980), pp. 445–456. DOI: 10.1145/322203.322207.

[SP78] Yossi Shiloach and Yehoshua Perl. "Finding Two Disjoint Paths Between Two Pairs of Vertices in a Graph". In: *Journal of the ACM* 25.1 (Jan. 1978), pp. 1–9. DOI: 10.1145/322047.322048.

[Spe09] Heike Sperber. "How to find Nash equilibria with extreme total latency in network congestion games?" In: *1st International Conference on Game Theory for Networks, GAMENETS 2009, Istanbul, Turkey, May 13-15, 2009*. Ed. by Tamer Basar and Hitay Özbay. IEEE, 2009, pp. 158–163. DOI: 10.1109/GAMENETS.2009.5137397.

[Spe10] Heike Sperber. "How to find Nash equilibria with extreme total latency in network congestion games?" In: *Mathematical Methods of Operations Research* 71.2 (Apr. 2010), pp. 245–265. DOI: 10.1007/s00186-009-0293-6.

[SS03] Andreas S. Schulz and Nicolás Stier-Moses. "On the Performance of User Equilibria in Traffic Networks". In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2003), 12-14 January, 2003, Baltimore, USA*. Baltimore, Maryland: Society for Industrial and Applied Mathematics, 2003, pp. 86–87. ISBN: 0898715385. DOI: 10.5555/644108.644121.

[Suu74] J. W. Suurballe. "Disjoint paths in a network". In: *Networks* 4.2 (1974), pp. 125–145. DOI: 10.1002/net.3230040204.

[SV08] Alexander Skopalik and Berthold Vöcking. "Inapproximability of pure nash equilibria". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC 2008), Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, 2008, pp. 355–364. DOI: 10.1145/1374376.1374428.

[SY91] Alejandro A. Schäffer and Mihalis Yannakakis. "Simple Local Search Problems that are Hard to Solve". In: *SIAM Journal on Computing* 20.1 (1991), pp. 56–87. DOI: 10.1137/0220004.

[Syr10] Vasilis Syrgkanis. "The Complexity of Equilibria in Cost Sharing Games". In: *Internet and Network Economics - 6th International Workshop, WINE 2010, Stanford, CA, USA, December 13-17, 2010. Proceedings*. Ed. by Amin Saberi. Vol. 6484. Lecture Notes in Computer Science. Springer, 2010, pp. 366–377. DOI: 10.1007/978-3-642-17572-5_30.

[Tal02] Kunal Talwar. "The Single-Sink Buy-at-Bulk LP Has Constant Integrality Gap". In: *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*. Ed. by William J. Cook and Andreas S. Schulz. Vol. 2337. Lecture

Notes in Computer Science. Springer, 2002, pp. 475–486. DOI: 10.1007/3-540-47867-1_33.

[Tho05] Torsten Tholey. "Finding Disjoint Paths on Directed Acyclic Graphs". In: *Graph-Theoretic Concepts in Computer Science, 31st International Workshop, WG 2005, Metz, France, June 23-25, 2005, Revised Selected Papers*. Ed. by Dieter Kratsch. Vol. 3787. Lecture Notes in Computer Science. Springer, 2005, pp. 319–330. DOI: 10.1007/11604686_28.

[Tho80] Carsten Thomassen. "2-Linked Graphs". In: *European Journal of Combinatorics* 1.4 (1980), pp. 371–378. DOI: 10.1016/S0195-6698(80)80039-4.

[Tin02] Cesare Tinelli. "A DPLL-Based Calculus for Ground Satisfiability Modulo Theories". In: *Logics in Artificial Intelligence, European Conference, JELIA 2002, Cosenza, Italy, September, 23-26, Proceedings*. Ed. by Sergio Flesca, Sergio Greco, Nicola Leone, and Giovambattista Ianni. Vol. 2424. Lecture Notes in Computer Science. Springer, 2002, pp. 308–319. DOI: 10.1007/3-540-45757-7_26.

[Tuy+95] Hoang Tuy, Saied Ghannadan, Athanasios Migdalas, and Peter Värbrand. "The Minimum Concave Cost Network Flow Problem with fixed numbers of sources and nonlinear arc costs". In: *Journal of Global Optimization* 6.2 (1995), pp. 135–151. DOI: 10.1007/BF01096764.

[Tuy00] Hoang Tuy. "The MCCNF Problem With a Fixed Number of Nonlinear Arc Costs: Complexity and Approximation". In: *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*. Ed. by Panos M. Pardalos. Boston, MA: Springer US, 2000, pp. 525–544. ISBN: 978-1-4757-3145-3. DOI: 10.1007/978-1-4757-3145-3_30.

[Val79] Leslie G. Valiant. "The complexity of computing the permanent". In: *Theoretical Computer Science* 8.2 (1979), pp. 189–201. DOI: 10.1016/0304-3975(79)90044-6.

[War52] John G. Wardrop. "Some Theoretical Aspects of Road Traffic Research". In: *Proceedings of the Institution of Civil Engineers* 1.3 (1952), pp. 325–362. DOI: 10.1680/ipeds.1952.11259.

[War99] Julie A. Ward. "Minimum-Aggregate-Concave-Cost Multicommodity Flows in Strong-Series-Parallel Networks". In: *Mathematics of Operations Research* 24.1 (1999), pp. 106–129. DOI: 10.1287/moor.24.1.106.

[WKD20] Zhengtian Wu, Hamid R. Karimi, and Chuangyin Dang. "A Deterministic Annealing Neural Network Algorithm for the Minimum Concave Cost Transportation Problem". In: *IEEE Transactions on Neural Networks and Learning Systems* 31.10 (2020), pp. 4354–4366. DOI: 10.1109/TNNLS.2019.2955137.

[ZN12] Cong Zhang and Hiroshi Nagamochi. "The Next-to-Shortest Path in Undirected Graphs with Nonnegative Weights". In: *Eighteenth Computing: The Australasian Theory Symposium, CATS 2012, Melbourne, Australia, January 2012*. Ed. by Julián Mestre. Vol. 128. CRPIT. Australian Computer Society, 2012, pp. 13–20. URL: http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV128Zhang.html.

Global Notation

2-DPP	2 disjoint paths problem
2-DSPP	2 disjoint shortest paths problem
k-DPP	k disjoint paths problem
k-DSPP	k disjoint shortest paths problem
$\Phi(\sigma)$	potential function, $\Phi(\sigma) = \sum_{r \in R} \sum_{i=1}^{n_\sigma(r)} c_r(i)$
$S \circ R$	$= \{(u, w) \in M \times M \mid \exists v \in M : u\, R\, v \wedge v\, S\, w\}$, composition of binary relations R and S on set M
$A \triangle B$	$= (A \setminus B) \cup (B \setminus A)$, symmetric difference of sets A and B
$f \in \mathcal{O}(g)$	asymptotically upper bounded, $\exists c \in \mathbb{R}_{>0} \exists n_0 \in \mathbb{N} \forall n \geq n_0 : f(n) \leq cg(n)$
$f \in \Omega(g)$	asymptotically lower bounded, $\exists c \in \mathbb{R}_{>0} \exists n_0 \in \mathbb{N} \forall n \geq n_0 : f(n) \geq cg(n)$
$f \in \Theta(g)$	asymptotically upper and lower bounded, $f \in \mathcal{O}(g) \wedge f \in \Omega(g)$
Circuit Satisfiability	decision problem: Given a Boolean circuit with n input bits and one output bit, is there an assignment for the input bits such that the circuit outputs 1?
Exact-Cover by 3-Sets (X3C)	decision problem: Input: ground set X, family $\mathcal{S}$ of 3-element subsets of X. Is there an exact cover $\mathcal{C} \subseteq \mathcal{S}$ of X?
α-equilibrium	α-approximate pure Nash equilibrium. Profile where no player can decrease her cost by at least a factor of α by unilaterally deviating
equilibrium	pure Nash equilibrium (NE). Profile where no player can decrease her cost by unilaterally deviating
fair cost allocation	cost sharing scheme, where every resource has a constant cost that is shared equally by all players using the resource

LP	Linear Program
PLS	class of polynomial-time local search problems
#P(sharp P)	class of functions that count the number of accepting paths of a non-deterministic Turing machine
NP	class of decision problems solvable in polynomial time by a non-deterministic Turing machine
PoA	Price of Anarchy. Maximal ratio of social cost of an equilibrium and social cost of any profile $$\mathrm{PoA}(\mathcal{G}) = \max_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\mathrm{OPT}(\mathcal{G}))}$$
PoS	Price of Stability. Minimal ratio of social cost of an equilibrium and social cost of any profile $$\mathrm{PoS}(\mathcal{G}) = \min_{\sigma^* \text{ equilibrium in } \mathcal{G}} \frac{C(\sigma^*)}{C(\mathrm{OPT}(\mathcal{G}))}$$
QF_LRA	quantifier-free formulas over the linear fragment of the theory of reals
SAT	Satisfiability of Boolean Formulas
SMT	Satisfiability Modulo Theories

Notation for Chapter 3

$\alpha(d)$	lower bound of non-existence of equilibria in polynomial congestion games of degree d, $= \sup\{\min\left\{\frac{1+na(1+w)^d}{(1+nw)^k+na}, \frac{(1+w)^k+aw^d}{(nw)^k+a(1+w)^d}\right\} : n \in \mathbb{N}, k \in \{1,\dots,d\}, w \in [0,1], a \in [0,1]\}$, grows as $\Omega\left(\frac{\sqrt{d}}{\ln d}\right)$
$\varepsilon(\mu)$	$= \frac{3^{d+d/2}}{\mu-1}$
$\varepsilon'(\mu)$	$= \frac{3^{d+d/2}}{\mu}$
$\mathcal{G}_\mu^d(\mathcal{C})$	unweighted polynomial congestion game where in any α-equilibrium the players emulate the computation of the Boolean circuit $\mathcal{C}$ for any $\alpha < 3^{d/2}$
$\mathcal{N}_\mu^d(\mathcal{C})$	unweighted polynomial network game where in any α-equilibrium the players emulate the computation of the Boolean circuit $\mathcal{C}$ for any $\alpha < 3^{d/2}$
$\mathcal{G}_n$	weighted congestion game with general cost functions without α-equilibria for $\alpha < \varphi_n$
$\mathcal{G}^d_{(n,k,w,a)}$	weighted polynomial congestion game without $\alpha(d)$-equilibria
$\mathcal{G}_\mu^d(\overline{\mathcal{C}}) \to \overline{\mathcal{G}}_{n_2}$	combined game of circuit gadget and non-existence gadget for general cost functions
$\mathcal{G}_\mu^d(\mathcal{C}) \to \overline{\mathcal{G}}_\gamma$	combined game of circuit gadget and non-existence gadget for polynomial cost functions
φ_n	unique positive solution to $(x+1)^n = x^{n+1}$
$\bar{\varphi}_{n-1}$	$= \lfloor \frac{n}{\ln n} \rfloor$ integer under-approximation of φ_{n-1}

Notation for Chapter 4

$\mathcal{B}_f(n,x,y)$	bridge instance with n players, weights x and y and underlying function f
$B(v,e)$	$= \left\{w \in V : d_{T,\bar{f}}(v,w) \leq \frac{f(1)-f(2)}{2}\gamma_e\right\}$ absorption-ball
$c_\sigma^{+1}(e)$	$= c_e(n_\sigma(e)+1)$
$D_T(v)$	$= \{u \in V : v \in T[u,r]\}$, descendants of node v in tree T (including v)
$D_T^W(w)$	$= \{v \in D_T(W) : w'(v) = w\}$, descendants of nodes in W w.r.t. tree T, where w is the first node met in W
$\vec{d}_{T,g}(u,v)$	$= \sum_{i=1}^{k} \gamma_{e_i} g(i)$ where e_i are the edges of $T[u,v]$ ordered by their position starting from u; unidirectional broadcast distance
$d_{T,g}(u,v)$	$= \sum_{i=1}^{k} \gamma_{e_i} g(i)$ where e_i are the edges of $T[u,v]$ ordered by decreasing γ; bidirectional broadcast distance
$F(k)$	$= \sum_{i=1}^{k} f(i)$
$\bar{f}(k)$	$= \frac{F(k)}{k}$
$\mathcal{F}_f(n,x,y)$	fan instance with n players, weights x and y and underlying function f
$\mathcal{F}^*$	subclass of fan instances where the star S_n is one of the best equilibria
$\mathcal{F}_{\text{lin}}$	class of functions of the form $f_s(k) = s + \frac{1-s}{k}$ for $s \in [0,1)$
$\mathcal{F}_{\text{poly}}$	class of functions of the form $f_\alpha(k) = k^{\alpha-1}$ for $\alpha \in [0,1)$
$\mathcal{G}_f(L,D,W)$	general uniform network game used in the PLS-reduction, L is the length of heavy paths, D the number of dummy players associated to every heavy edge, and W the scaling factor of each heavy edge

$\left(\mathcal{G}_{(X,\mathcal{S})}, \beta\right), \left(\bar{\mathcal{G}}_{(X,\mathcal{S})}, \beta\right)$	network games constructed from an instance $(X, \mathcal{S})$ of Exact-Cover by 3-Sets (X3C)
$\mathcal{H}((w_1, r_1), \ldots, (w_n, r_n))$	$= \frac{w_1+\ldots+w_n}{\frac{w_1}{r_1}+\ldots+\frac{w_n}{r_n}}$, weighted harmonic mean of elements $r_1, \ldots, r_n \in \mathbb{R}_{>0}$ with weights $w_1, \ldots, w_n \in \mathbb{R}_{\geq 0}$
homogeneous profile	w.r.t. a spanning tree T, if $\forall v, w \in V : \Phi(\sigma) \leq \Phi\left(\sigma^{T[v \to w]}\right)$
(LP_z^T)	the LP used to compute $\mathrm{PoS}^f_{\mathcal{F}^*}(n)$
$M(f)$	$= \max_{n \in \mathbb{N}_{>0}} \frac{\sum_{i=1}^{n} f(i)}{n f(n)}$, upper bound on the PoS for uniform network games
(NECond)	characterization for a spanning tree being an equilibrium in a broadcast game
$n_\sigma^U(e)$	$= \lvert\{u \in U : e \in \sigma_u\}\rvert$, congestion on e counting only players in U
$\mathrm{PoS}^f_{\mathcal{F}^*}(n)$	PoS in the class of uniform fan instances with underlying function f and at most n players where the star S_n is one of the best equilibria
r_{absorb}	$= \frac{f(1)-f(2)}{2}\gamma_e$ absorption-radius at v w.r.t. improving edge e
r_{charge}	$= \frac{r_{\text{absorb}}}{\alpha}$, where $\alpha > 2$; charging-radius
$\rho(z)$	$= \frac{\sum_{i=2}^{n'} y^*_{z,i}}{\sum_{i=2}^{n'} x^*_{z,i}}$
$R(T_t)$	$= \frac{C_{(x^*_{z_n^t}, y^*_{z_n^t})}(S_n)}{C_{(x^*_{z_n^t}, y^*_{z_n^t})}(T_t)}$, lower bound on $\mathrm{PoS}^{f_s}_{\mathcal{F}^*}(n)$ and $\mathrm{PoS}^{f_\alpha}_{\mathcal{F}^*}(n)$; objective function value of (LP_z^T) for tree T_t and z_n^t
S_n	star rooted at r in fan instance $\mathcal{F}_f(n, x, y)$
sharing function with economies of scale	function $f : \mathbb{N}_{\geq 1} \to \mathbb{R}_{\geq 0}$ satisfying • f is strictly decreasing or constantly zero • $kf(k)$ is non-decreasing and $(k+1)f(k+1) - kf(k)$ is non-increasing

$\sigma^{T,w}$	profile resulting from σ by replacing parts of the support by tree T
$\sigma^{T[v\to w]}$	profile resulting from σ by adding the path $T[v,w]$ to the support
$\sigma^{T[B(v,e)]}$	profile resulting from σ by absorbing around v
$\sigma^{\mathcal{C}}$	profile in $\left(\mathcal{G}_{(X,\mathcal{S})},\beta\right), \left(\bar{\mathcal{G}}_{(X,\mathcal{S})},\beta\right)$ corresponding to an exact cover $\mathcal{C}$, where $\sigma^{\mathcal{C}}_S = \{S,r\}$ and $\sigma^{\mathcal{C}}_x = \{\{x,C(x)\},\{C(x),r\}\}$
$\mathcal{S}^\sigma$	$= \{S \in \mathcal{S} : n_\sigma(\{S,r\}) > \beta\}$, cover/packing corresponding to an equilibrium in $\left(\mathcal{G}_{(X,\mathcal{S})},\beta\right), \left(\bar{\mathcal{G}}_{(X,\mathcal{S})},\beta\right)$
tree-move	replacing spanning tree T by spanning tree $T' = T \setminus \{e_T(v)\} \cup \{e\}$ for $e = \{u,v\} \in E \setminus T$ with $c_\tau(T[v \nearrow u]) > c_e(1) + c_\tau^{+1}(T[u \nearrow v])$
T_t	tree containing all edges $r-1-2-\ldots-n$ and $\{r,t\}$ in fan instance $\mathcal{F}_f(n,x,y)$
$T[u \nearrow v]$	$= T[u, \mathrm{lca}\,(u,v)]$
(x^*_z, y^*_z)	the all non-zero basic solution to (LP^T_z) given by (4.36) and (4.37)
$z^*_n(\zeta)$	$= (\zeta,0,\ldots,0,1) \in \{0,1\}^n$ for $\zeta \in \{0,1\}$
z^t_n	$= (0,1,\ldots,1,0,\ldots,0,1) \in \{0,1\}^n$, where $z_1 = 0$ and $z_t,\ldots,z_{n-1} = 0$ and all other entries are 1

Notation for Chapter 5

$u \overset{\rightarrow}{\vdots}_L v$	there are disjoint $u_i \to v_i$ paths in L
$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \overset{\ell}{=\!=}_{E'} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$	there is a shortest $u_1 - v_1$ path and a shortest $u_2 - v_2$ path in E' w.r.t. ℓ that are disjoint
$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \overset{L_1}{\underset{L_2}{\rightrightarrows}} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$	there is a $u_1 \to v_1$ paths in L_1 and a $u_2 \to v_2$ path in L_2 which are disjoint
$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \overset{L_1}{\underset{L_2}{\rightleftarrows}} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$	there is a $u_1 \to v_1$ paths in L_1 and a $v_2 \to u_2$ path in L_2 which are disjoint
A^s	$= \quad \{(u,v) : \{u,v\} \in E^s \wedge d^s(u) < d^s(v)\}$, oriented edges of the shortest paths network rooted at s w.r.t. ℓ
E^s	$= \{\{u,v\} \in E : \ell(\{u,v\}) = \lvert d^s(u) - d^s(v)\rvert\}$, edges of the shortest paths network rooted at s w.r.t. ℓ
$d^s(v)$	$= \min_{s-v \text{ path } P} \quad \sum_{e \in P} \ell(e)$, length of a shortest $s-v$ path w.r.t. ℓ
G^A	$= G/E$, contracting all undirected edges in a mixed graph

G^E	$= (V, E)$, only considering the undirected edges of a mixed graph
$\overrightarrow{G}$	$(V, E_0 \cup A^{s_1} \cup A^{s_2})$, shortest path orientation of $G = (V, E)$ w.r.t. s_1, s_2, and ℓ
$\overrightarrow{G}^{\text{uni}}$	$(V, E_0 \cup (A^{s_1} \cap A^{s_2}))$, the unidirectional layer of the shortest path orientation
$\text{Id}(V)$	$= \{(v, v) : v \in V\}$
$V^k_{\neq}$	set of k tuples of pairwise different elements of V

Index

A Appendix for Chapter 3

Lemma 43. *Define the sequence* $g : \mathbb{N}_{\geq 2} \to \mathbb{R}$ *by*

$$g(d) = \left(1 + d^{-\frac{1}{2(k_d+1)}}\right)^{k_d} + \frac{2\sqrt{d}}{\ln d \left(1 + \frac{\ln d}{2d}\right)^d} \qquad \text{where } k_d = \left\lceil \frac{\ln d}{2 \ln \ln d} \right\rceil.$$

Then $\lim_{d\to\infty} g(d) = 1$.

Proof. Define

$$g_1(d) = \left(1 + d^{-\frac{1}{2(k_d+1)}}\right)^{k_d} \quad \text{and} \quad g_2(d) = \frac{2\sqrt{d}}{\left(1 + \frac{\ln d}{2d}\right)^d \ln d}$$

so that $g(d) = g_1(d) + g_2(d)$. We will show the desired convergence by establishing that $\lim_{d\to\infty} g_1(d) = 1$ and $\lim_{d\to\infty} g_2(d) = 0$. We will make use of the following inequalities (see, e.g., [Olv+10, Eq. 4.5.13]):

$$\exp\left(\frac{xy}{x+y}\right) < \left(1 + \frac{x}{y}\right)^y < \exp(x), \quad \text{for all } x, y > 0. \tag{A.1}$$

First, we show $\lim_{d\to\infty} g_1(d) = 1$. As d and k_d are positive, we have $g_1(d) > 1$ for every d. Furthermore, g_1 is increasing in k_d, and $k_d < \frac{\ln d}{2 \ln \ln d} + 1$. Thus,

$$g_1(d) < \left(1 + d^{-\frac{1}{\frac{\ln d}{\ln \ln d}+4}}\right)^{\frac{\ln d}{2 \ln \ln d}+1}.$$

Using the second inequality of (A.1) with $y = \frac{\ln d}{2 \ln \ln d} + 1$ and $x = y d^{-\frac{\ln \ln d}{\ln d + 4 \ln \ln d}}$, we can further bound

$$g_1(d) < \exp\left(\frac{\frac{\ln d}{2 \ln \ln d} + 1}{d^{\frac{\ln \ln d}{\ln d + 4 \ln \ln d}}}\right). \tag{A.2}$$

We will show that the argument of the exponential function on the r.h.s. of (A.2) goes to 0 for $d \to \infty$, thus proving the claim. Replacing $\ln d = \exp(\ln \ln d)$ in the numerator and $d = \exp(\ln d)$ in the denominator, that argument can be written as

$$\frac{\exp(\ln \ln d)\left(\frac{1}{2 \ln \ln d} + \frac{1}{\ln d}\right)}{\exp\left(\frac{\ln d \ln \ln d}{\ln d + 4 \ln \ln d}\right)} = \left(\frac{1}{2 \ln \ln d} + \frac{1}{\ln d}\right) \exp\left(\frac{4(\ln \ln d)^2}{\ln d + \ln \ln d}\right). \tag{A.3}$$

The argument of the exponential function on the r.h.s. of (A.3) goes to 0, as $\ln d$ is the dominating term in the denominator for $d \to \infty$. Thus, the whole expression in (A.3) goes to 0.

Next, we show that $\lim_{d\to\infty} g_2(d) = 0$. As $d \geq 2$, we have that $g_2(d) > 0$ for every d. Using the first inequality of (A.1) with $x = \frac{\ln d}{2}$ and $y = d$, we have

$$\left(1 + \frac{\ln d}{2d}\right)^d > \exp\left(\frac{d \ln d}{\ln d + 2d}\right).$$

Thus, we obtain an upper bound on g_2 by writing $\sqrt{d} = \exp\left(\frac{\ln d}{2}\right)$:

$$g_2(d) < \frac{2\exp\left(\frac{\ln d}{2}\right)}{\ln d \exp\left(\frac{d \ln d}{\ln d + 2d}\right)} = \frac{2}{\ln d} \exp\left(\frac{(\ln d)^2}{2\ln d + 4d}\right). \tag{A.4}$$

As $4d$ is the dominating term in the denominator of the argument of the exponential function, the argument goes to 0 for $d \to \infty$, and thus the r.h.s. of (A.4) goes to 0, showing the claim. □

B Appendix for Chapter 4

Lemma 44.

$$\forall n \in \mathbb{N}_{\geq 3} \forall k \in \{1, \ldots, n-1\} : \sum_{j=0}^{k} \binom{k}{j} \frac{1}{(-1)^j (n-j)} = \frac{(-1)^k}{(n-k)\binom{n}{k}}$$

Proof. Proof by induction on k. Base case $k = 1$:

$$\sum_{j=0}^{k} \binom{k}{j} \frac{1}{(-1)^j (n-j)} = \frac{1}{n} - \frac{1}{n-1} = -\frac{1}{n(n-1)} = \frac{(-1)^k}{(n-k)\binom{n}{k}}.$$

Step $k \to k+1$:

$$\begin{aligned}
\sum_{j=0}^{k+1} \binom{k+1}{j} \frac{1}{(-1)^j (n-j)} &= \sum_{j=0}^{k+1} \left(\binom{k}{j} + \binom{k}{j-1} \right) \frac{1}{(-1)^j (n-j)} \\
&= \sum_{j=0}^{k} \binom{k}{j} \frac{1}{(-1)^j (n-j)} + \binom{k}{k+1} \frac{1}{(-1)^{k+1} (n-(k+1))} \\
&\quad + \sum_{j=0}^{k+1} \binom{k}{j-1} \frac{1}{(-1)^j (n-j)} \\
&= \sum_{j=0}^{k} \binom{k}{j} \frac{1}{(-1)^j (n-j)} + \sum_{j=0}^{k} \binom{k}{j} \frac{1}{(-1)^{j+1} ((n-1)-j)} \\
&= \frac{(-1)^k}{(n-k)\binom{n}{k}} - \frac{(-1)^k}{(n-1-k)\binom{n-1}{k}} \\
&= (-1)^k k! \frac{1}{(n-1)\cdots(n-k)} \left(\frac{1}{n} - \frac{1}{n-k-1} \right) \\
&= (-1)^{k+1} \frac{(k+1)!}{n(n-1)\cdots(n-k)(n-k-1)} \\
&= \frac{(-1)^{k+1}}{(n-(k+1))\binom{n}{k+1}}
\end{aligned}$$

where we used the induction hypothesis in equation 4 for k and n and for k and $n-1$ for the respective summands. □