Designing and Analyzing Cost-Sharing Mechanisms

under fundamental performance objectives

Dissertation

von Yvonne Bleischwitz

Schriftliche Arbeit zur Erlangung des Grades
Doktor der Naturwissenschaften

an der Fakultät für
Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

Paderborn, 14. April 2008

Alles wird gut.

Preface

A problem is a chance for you to do your best.

Edward Kennedy "Duke" Ellington (1899–1974)

Acknowledgements

I am indepted to many people who have accompanied me on the long and winding road to finally writing up my thesis. First and foremost, I would like to thank my advisor Prof. Dr. Burkhard Monien for his scientific guidance and patience. I have sincerely appreciated his continuous support. Through his own valuable research, comments, and questions, he has encouraged and enlightened me. He taught me to be persistent – and passionate – to accomplish a goal. I am grateful for this and many other lessons learned.

I have especially enjoyed the relaxed and amicable atmosphere within the research group of Prof. Dr. Monien in which I always felt very comfortable. For providing this convenient background for my research, I would like to express thanks to all my present and former colleagues. Special thanks go to:

- Florian Schoppmann, for the fruitful collaboration over the last two years. It was a pleasure to work with you!
- Dr. Rainer Feldmann, for encouraging me to aim for a doctorate degree and reassuring me when I was on the verge of giving up.
- Dr. Ulf-Peter Schröder, for giving valuable advice on scientific and non-scientific issues.
- Bernard Bauer, Marion Rohloff, Sigrid Gundelach, Uli Ahlers, and Thomas Thissen for assisting me in many ways.

The first three years of my research were financed by the 'International Graduate School of Dynamic Intelligent Systems', which I very much appreciate.

Life would not be enjoyable without my dear friends. Thank you for sharing the bright side of life in particular and standing by me through hard times as well!

Most importantly, I thank my family on whose constant encouragement and unconditional love I can always rely. Thank you for enduring my moods and keeping me sane!

I cannot end without thanking Alberto for his love, support, and patience over the last two years, and for his courage to be willing to break new grounds together.

Contents

1

Introduction

1.1 Motivation and Framework

Do you always tell the truth when participating in an auction? Probably not. You would take the opportunity to reduce your price by cheating. Not that we wish to suggest that you have dishonest motives; we rather agree with the common economic assumption that people act *selfishly* in order to maximize their profit. On the other hand, if you sell something by auction, what would you do to prevent such manipulation? What *incentives* would you offer for truth-telling?

This thesis considers truth elicitation in the context of *cost sharing*, where a certain *service* is auctioned off by a *service provider* who selects which players to serve and at what price. Next to designing the rules for this selection process, the service provider faces another difficulty: depending on the kind of service, he has to solve an underlying optimization problem in order to actually make the service accessible to the selected players. In addition, all of the provider's tasks have to be accomplishable in a reasonable amount of time.

We examine cost-sharing problems under both players' and provider's objectives. While on the application level we mainly focus on scheduling scenarios, we also consider the problem of satisfying players' connectivity requirements within a network. In particular, our work is motivated by the following three examples:

Example 1.1. A computing center uses an auction-based approach in order to process customers' jobs. A customer request involves stating the required processing time of his job and a price limit. On the basis of these requests, the computing center accepts a selection of requests and determines corresponding prices. Minimizing the actual cost of scheduling the chosen jobs requires the application of adequate scheduling algorithms. A potential underlying optimization problem is the problem of *scheduling on related parallel machines*, where machines are simply specified by speeds. A possible cost measure is given by the maximum completion time over all jobs (the *makespan*). The problem of minimizing the makespan is denoted by $(\mathrm{Q}||\mathrm{C}_{\max})$ (see Section 3.2.1 and, e.g., [20, 26, 43–46, 51, 52, 58, 68, 69, 138]).

Example 1.2. A county plans to establish a huge industrial area for industries with high energy demand (e.g., iron and steel works, chemical plants, or large cement works). Assume that each company has already been assigned a potential site. To provide access to the electricity network, the county needs to build several transformer stations that reduce voltage for distribution. As the cost for building the transformer stations and establishing the power supply lines is to be shared among the companies, an auction is conducted. Hence, companies with insufficient budget risk to be excluded. Having determined the successful companies, the county has to decide on the number and locations of stations to erect and on the power supply lines between stations and companies. To prevent power shortage due to malfunctioning stations, the county may allow companies to bid on any amount of connections to distinct stations and then realize the number of connections purchased by the auction. The underlying optimization problem of minimizing the total cost of a realized solution is the *fault-tolerant facility location problem* (FaultTolerantFL). If at most one connection to a station is allowed, this restricted problem becomes the *facility location problem* (FacilityLocation) (see Section 3.2.3 and, e.g., [18, 23, 64, 65, 75, 77, 78, 89, 90, 119, 120, 123, 133]).

Example 1.3. Cooperating companies aim to establish a train network for exchanging their goods. Required connections between pairs of companies have already been identified. In order to guarantee punctual delivery of time-critical goods even in case of track failure, the number of disjoint connections between a pair of companies should be proportional to some priority measure. Given each company's willingness to contribute to the train project, it has to be decided on the exclusion or connection reduction of financially weak companies, the payments of the remaining companies, and on a solution that meets the requirements for the remaining companies. The underlying optimization problem with respect to minimizing the cost of the solution can be modeled as a *generalized Steiner forest problem* (GenSteinerForest), or as the restricted *Steiner forest problem* (SteinerForest) if each company has at most one connection requirement(see Section 3.2.4 and, e.g., [1, 9, 56, 74, 80, 135, 136]).

Examples 1.1 to 1.3 address providing players with a *service*. In the context of cost sharing, the notion of service is a broad term. Obviously, 'job-processing' is a service. On a more abstract level, 'power supply' or 'being connected within a transport network' are services as well. Three basic tasks common to all examples are:

1. Collect *bids* from players indicating the amount of money they are willing to pay.
2. On the basis of these bids, determine the set of served players and their payments (usually referred to as *cost shares*).
3. Compute a solution in order to establish the service for the selected players.

We assume that this whole process is conducted by a central authority, referred to as *the service provider*. In order to decide about which players to serve and at what price, the provider applies a *cost-sharing mechanism*, which is simply a protocol defining the rules for this decision process. This mechanism is public knowledge.

There are many conflicting interests from several perspectives. Players are assumed to be rational and to act selfishly in order to maximize their own benefit. Thus, they may communicate *untruthful* bids to the service provider that differ from their *true valuations* for receiving the service. Players might even collude and agree on collective deceit if this is beneficial for all of them. The service provider's primary goal is therefore to create incentives for telling the truth. He achieves this by employing *incentive-compatible* cost-sharing mechanisms. The most common notions of incentive compatibility are *strategyproofness* (SP), where no single player can unilaterally improve his outcome by submitting an untruthful bid, or even *group-strategyproofness* (GSP), where no coalition can jointly submit false bids such that the resulting outcome is at least as good for all coalition members and strictly better for at least one member.

In addition, the service provider wants to recover the cost incurred by serving the selected players, while at the same time he needs to be able to offer competitive prices.

In the best case, leaving aside polynomial-time computability, the provider computes a minimum cost solution for serving the selected players and applies a cost-sharing mechanism whose cost shares sum up to the corresponding cost. A mechanism that always achieves this is called *budget-balanced* (1-BB).

An economic objective is to design *efficient* (1-EFF) mechanisms that optimally trade off the service cost and the valuations of the players. The most established notion of efficiency is maximizing the *social welfare*, which is the sum of the served players' valuations minus the minimum cost of serving them. Equivalent to maximizing the social welfare is minimizing the *social cost*, which is the minimum cost of serving the selected players plus the sum of the valuations of the excluded players.

In the majority of cases it will not be possible to achieve incentive-compatibility, 1-BB, and 1-EFF at the same time, since already SP places high restrictions upon the cost shares. Furthermore, the underlying optimization problems may be NP-hard[1] and the provider has to resort to approximate solutions for the sake of computability. We follow the line of previous work (see, e.g., [16, 19, 66, 73, 76, 81, 82, 86, 97, 106, 114, 115]) and approach this problem by designing incentive-compatible mechanisms with *approximate* budget-balance and *approximate* efficiency guarantees. We especially consider β-budget-balance (β-BB, $\beta \geq 1$), demanding that the sum of the cost shares of the served players is not larger than β times the *optimal* cost and not smaller than the *actual* cost of serving them. We focus on approximating social *cost* efficiency, where a mechanism is called γ-efficient (γ-EFF, $\gamma \geq 1$), if the *actual* social cost of the computed solution is not larger than γ times the *optimal* social cost.

To summarize, given an underlying optimization problem Π, the problem is to determine a mechanism and an approximation algorithm (both polynomial-time computable), such that the mechanism is incentive-compatible, β-BB, and γ-EFF in the strongest possible sense (i.e., preferably GSP, and β, γ as small as possible), where the actual cost is induced by the approximate solution. We term these problems Π-cost-sharing problems. We remark that economics mainly considers cost-sharing problems that ignore how the service is provided to the selected players. These problems are specified by a single cost function, and actual and optimal costs coincide.

Most research on cost-sharing assumes *binary demand*, where players are 'served' or 'not served'. We also consider the *general demand* setting, providing *service levels* ranging from 0 to some maximum number. This is of particular interest when players require different *qualities of service*. For connectivity problems within a network, the service level of a player is the number of his distinct connections. More connections correspond to a higher quality of service, for reasons including throughput and resistance to link failure.

1.2 Classification

Problem settings involving self-interested individuals who pursue competing goals are vital research topics. Many of these settings involve making decisions while individuals' actual preferences are not publicly observable. Important applications include auctions, pricing, deciding about public projects, and voting. Naturally, *economics* and *game theory* provide tools for modeling and solving such problems.

Economics studies the production, distribution, and consumption of goods and services. Its origin in its modern sense is conventionally accredited to Adam Smith [121].

[1] For the complexity theory basics, we refer the reader to [51, 107, 111, 134]

Political economy, considered as a branch of the science of a statesman or legislator, proposes two distinct objects: first, to supply a plentiful revenue or product for the people, or, more properly, to enable them to provide such a revenue or subsistence for themselves; and secondly, to supply the state or commonwealth with a revenue sufficient for the public services. It proposes to enrich both the people and the sovereign [121].

Game theory attempts to mathematically capture behavior in strategic situations, where an individual's success in making choices according to pre-defined strategies depends on the choices of others. An outstanding goal is finding equilibria of strategy choices where individuals are unlikely to change their behavior. The work of John von Neumann and Oskar Morgenstern [132] is widely considered to be path breaking for present-day game theory as it established the basis for the economic application of classical game theory.

We shall attempt to utilize only some commonplace experience concerning human behavior which lends itself to mathematical treatment and which is of economic importance. [...] We shall find it necessary to draw upon techniques of mathematics which have not been used heretofore in mathematical economics, and it is quite possible that further study may result in the future in the creation of new mathematical disciplines. [132].

Microeconomics is a branch of economics which examines the economic behavior of players (including individuals and firms) and their interactions through markets, given scarcity of resources and government regulation. *Mechanism design*, as a subfield of microeconomics, especially studies how private information can be elicited and how this affects making a choice that responds to individual preferences. The seminal work of Leonid Hurwicz [71, 72] marks the birth of mechanism design. It especially introduced the key notion of *incentive-compatibility*. Erik Maskin [91] pioneered work on implementation theory which deals with the problem of the potential co-existence of inferior equilibria along with the desired ones. Roger Myerson [102] introduced incomplete information to mechanism design theory.[2]

While for a long time economic literature (and mechanism design in particular) concerned itself mainly with incentives and not with the algorithmic efficiency of its acquired methods, computer science traditionally did the opposite. Obviously however, there are many applications that need to consider both incentives and computational complexity. An outstanding example is the design of network protocols for the Internet; they need to enable efficient data interchange between a large number of autonomous systems. Most notably, the pioneering work of Nisan and Ronen [104] initiated the study of mechanisms under computational aspects.

1.3 Related Work

We first consider binary demand cost-sharing and review the few results known for general-demand cost-sharing at the end of this section. The optimization problems mentioned below are standard computer science problems. Consult Chapter 3 and Appendix D for details. When we say in the following that a mechanism is β-BB or γ-EFF for a Π-cost-sharing problem, we indirectly imply the existence of an approximation algorithm for Π that induces the actual cost.

[2] We find it noteworthy that the most recent Nobel price for economics in 2007 has been awarded to Hurwicz, Maskin, and Myerson for 'laying the foundations of mechanism design theory'.

General Impossibility Results

For the objectives incentive-compatibility, 1-BB, and 1-EFF, a significant negative result has been known for over thirty years. Already in the 70's, Green and Laffont [60, 61] ruled out the existence of SP mechanisms that simultaneously guarantee 1-BB *and* 1-EFF, even for a simple cost function. As a consequence, at least one objective has to be (at least partially) sacrificed. Moreover, Feigenbaum et al. [36] showed that the request for SP and a constant-factor approximation of budget-balance generally precludes a simultaneous constant-factor approximation of social welfare efficiency. With respect to computational tractability, Feigenbaum et al. [37] proved that there are cost functions derived from optimal solutions to SteinerTree problems, for which the social welfare cannot be approximated to within a constant factor in polynomial time, assuming P $\neq$ NP.

Moulin Mechanisms

Primary works on cost sharing (see, e.g., [76, 81, 97, 106]) completely ignored the efficiency objective and focused on achieving GSP and β-BB for β as small as possible. In this context, *Moulin mechanisms* [97] have attracted the greatest deal of attention. Essential ingredients of Moulin mechanisms are *cross-monotonic cost-sharing methods*. A cost-sharing method is simply a function that maps each set of players to a vector of cost shares; it is cross-monotonic, if the cost-share of a player in a specific set never increases as the set expands. Thomson [125, 126] was one of the first to introduce cross-monotonicity. Naturally, β-BB of a cost-sharing method (defined analogously to β-BB of a mechanism) implies β-BB of the corresponding Moulin mechanism [76]. To summarize, β-BB and cross-monotonic cost-sharing methods imply the existence of β-BB and GSP mechanisms realized by Moulin mechanism.

In fact, almost all known GSP mechanisms are Moulin mechanisms. Moreover, Moulin mechanisms even characterize all 1-BB and GSP mechanisms in case that costs are *submodular*, meaning that the marginal cost of adding a player set S to a player set T is never more than the marginal cost of adding S to $S \cap T$ (confer [97] and Theorem C.3, p. 120). While there are always 1-BB and GSP Moulin mechanisms for submodular cost (we discuss some of them in the next but one paragraph), this is not always true for non-submodular costs. We summarize all known (im)possibility results on the performance of Moulin mechanisms for diverse cost-sharing problems with underlying optimization problems that induce non-submodular costs in Tables A.1 and A.2 on pages 109 and 110, and resort to discussing some of the primary results and those directly related to this thesis.

Among the first 1-BB and cross-monotonic cost-sharing methods for non-submodular costs were the methods by Kent and Skorin-Kapov [81] for optimal SpanningTree costs. Jain and Vazirani [76] presented a whole family of 1-BB and cross-monotonic cost-sharing methods for these costs. As there are approximation algorithms for SteinerTree and TravelingSalesman that compute costs that are within twice the cost of an optimum spanning tree (see [93, 124, 129]), the cost-sharing methods of [76, 81] yield 2-BB cost-sharing methods for the induced cost-sharing problems. In addition, Könemann et al. [82] gave SteinerTree cost-sharing problems for which no cross-monotonic cost-sharing method can be β-BB for $\beta < 2$.

Cross-monotonic cost-sharing methods that are of importance within this thesis are the 3-BB methods proposed by Pál and Tardos [106] for FacilityLocation cost-sharing problems, and the 2-BB methods by Könemann et al. [82] for SteinerFor-

EST cost-sharing problems. These guarantees are tight under the cross-monotonicity requirement as shown by Immorlica et al. [73] and Könemann et al. [83].

The point of departure and motivation for this thesis is the work of Bleischwitz and Monien [10]. For $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems for m machines and n jobs with d different processing times, we showed that in general, no cross-monotonic cost-sharing methods that are β-BB for $\beta < d$ can exist. Furthermore, we gave cross-monotonic cost-sharing methods that are $2d$-BB. In case that either jobs or machines are identical (have the same processing times or speeds), we even presented $\frac{2m}{m+1}$-BB cross-monotonic cost-sharing methods. For cost-sharing problems that aim to minimize the sum of completion times (denoted by $(\mathrm{Q}||\sum \mathrm{C}_i)$), Brenner and Schäfer[16] showed that already for the setting with one machine, cross-monotonic cost-sharing methods can generally not be β-BB for $\beta < \frac{n+1}{2}$.

Motivated by the impossibility result of Feigenbaum et al. [36], Roughgarden and Sundararajan [114] measured efficiency by social *cost* instead of social welfare. They identified a criterion of cost-sharing methods termed *summability*, allowing for simultaneous budget-balance and social cost efficiency approximations by Moulin mechanisms within certain bounds. On the other hand, they as well showed that constant-factor approximations of social welfare efficiency are already impossible for constant costs; here, no Moulin mechanism can be better than H_n-EFF, where n is the number of players and H_n is the n-th harmonic number[3]. Hence, in case that constant costs are induced by some underlying optimization problem, $O(\log n)$-EFF is generally the best that can be hoped for. We remark that this is especially the case for $(\mathrm{Q}||\mathrm{C}_{\max})$, FACILITYLOCATION, and STEINERFOREST.

The work of Roughgarden and Sundararajan [114] allowed to show that the methods of Pál and Tardos [106] for FACILITYLOCATION cost-sharing problems actually yield $O(\log n)$-EFF Moulin mechanism, and that the Moulin mechanisms based on the methods of Könemann et al. [82] for STEINERFOREST cost-sharing problems are $O(\log^2 n)$-EFF [19], which is also asymptotically tight [115][3]. Brenner and Schäfer [16] identified a shortcoming of the approximate efficiency of the Moulin mechanisms in [10] for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems for identical machines by showing that they are generally no better than $O(n)$-EFF. They presented a new cross-monotonic cost-sharing method yielding $O(\log n)$-EFF while slightly impairing the budget-balance approximation. For a summary of the multitude of work on both upper and lower bounds on the approximate social cost efficiency of Moulin mechanisms we refer to Tables A.1 and A.2 on pages 109 and 110.

The Core

Closely related to finding 1-BB cross-monotonic cost-sharing methods is finding solutions for *cooperative games with transferable payoffs* (see, e.g., [8, 96, 98, 103, 105, 137]) that lie in the *core*. Adapted to our scenario, the task is to determine cost shares for a fixed set of players that exactly split their optimal service cost among them and are stable in the sense that no coalition pays more than the optimal cost of serving the coalition alone. The core was proposed in 1881 by Edgeworth [35] who defined the set of coalitionally stable states of an economy as 'final settlements'. It was rediscovered and introduced to game theory by Gillies [53].

The β-core is a relaxation of the core, demanding that the actual service cost is recovered while every coalition does not pay more than β times the optimal cost

[3] The original bounds from [114, 115] assumed 1-BB, due to a different notion of approximate budget-balance. We elaborate on these two models in Section 4.5.

for serving only the coalition (again, in the classical sense, actual and optimal costs coincide and are given by a cost function).

A major drawback of this concept is that the set of players for which the cost shares are determined is fixed. Hence, if the players fail to fully cooperate, it is not specified how to adjust cost-shares for a coalition. Generalized concepts were demanded and investigated by, among others, Dutta and Ray [34], Thomson [125–127], Sprumont [122], Chun [24], and Moulin [95]. Consequently, cost-sharing methods were considered. Cost-sharing methods are in the β-core if for each set of players, the computed cost shares are in the β-core. Natural limitations on the β-BB of cross-monotonic cost-sharing methods can be derived from the observation that every β-BB cross-monotonic cost-sharing method specifies cost shares in the β-core (confer Section 4.4, p. 33).

From the multitude of economic literature that studies the core we only mention some that we think are of interest in this thesis. Traditionally, the non-emptiness of the core is established by the Bondareva-Shapley theorem [15, 117]. The theorem identifies a certain *balance* condition for costs to be necessary and sufficient for the non-emptiness of the core. A way to provide lower bounds for the β-core was proposed by Jain and Vazirani [76] for costs that are implicitly defined by minimum solutions to *covering* integer programs, examples of which are SteinerTree and FacilityLocation. For these costs, Jain and Vazirani [76] showed that β is bounded by the integrality gap of the linear program relaxation of the problem by utilizing that these relaxation costs are *balanced* in the sense of Bondareva and Shapley [15, 117]. Immorlica et al. [73] noted that by using the approach of Jain and Vazirani [76], a lower bound of 1.463 for FacilityLocation cost-sharing problems can be obtained. Furthermore, 2 is a lower bound for SteinerTree cost-sharing problems due to the integrality gap of to 2 (see, e.g., [128]). Meggido [92] had previously shown that the core is empty for SteinerTree cost-sharing problems. Goemans and Skutella [54] and Chudak [23] give cost shares for FacilityLocation cost-sharing problems in the $\frac{e+2}{e}$-core, where $\frac{e+2}{e} \approx 1.736$.

Cost shares based on some considerations of equity between players are for example obtained by the *Shapley value* [116] and the *egalitarian solution* proposed by Dutta and Ray [34]. For submodular costs, Shapley [118] showed that the Shapley value is in the core, and Dutta and Ray [34] observed that this is true for the egalitarian solution as well.

Moulin Mechanisms for Submodular Costs

In case that costs are submodular, the Shapley value and the egalitarian solution play a major role as they are not only in the core, but give 1-BB cross-monotonic cost-sharing methods [33, 70, 112, 122]. Moreover, cross-monotonic cost shares are also computed by the 1-BB *sequential stand alone mechanisms* that consider players in a predefined order and let them pay the marginal cost of being added to the already selected set (confer Appendix B.1, p. 111). However, the Shapley value and the egalitarian solution are more equitable. Furthermore, already for constant costs, the approximation guarantee of sequential stand alone mechanisms with respect to social cost efficiency is no better than n (Appendix B.1, p. 111). Contrary, Roughgarden and Sundararajan [114] showed that Moulin mechanisms applying the 1-BB cross-monotonic cost-sharing methods computed by the Shapley value are H_n-EFF.

Moreover, Moulin and Shenker [99] showed that Moulin mechanisms computing cost shares via the Shapley value have the smallest social welfare efficiency loss among *all* Moulin mechanisms. Mutuswami [101] proved that, under some technical restrictions, Moulin mechanisms with cost shares computed by the egalitarian solution maximize the expected size of the set of service-receiving players over all Moulin mechanisms, assuming that valuations are independently and identically distributed.

The problem of using the Shapley value or the egalitarian solution is that they are in general not polynomial-time computable. However, for MULTICAST problems with fixed transmission trees which induce submodular costs, Feigenbaum et al. [37] observed that the Shapley value *is* polynomial-time computable. We remark that $(\mathrm{Q}||\mathrm{C}_{\max})$ problems do not yield submodular cost (confer Appendix B.1, p. 111).

Group-Strategyproof Non-Moulin Mechanisms

What are the chances to improve on the performance of Moulin mechanisms? More generally, do GSP non-Moulin mechanisms with good performance exist at all?

The first reasonable approach to design such mechanisms was made by Immorlica et al. [73], who gave a GSP and 1-BB non-Moulin mechanism for three players and a simple cost function. To the best of our knowledge, the only other approach was made by Bleischwitz et al. [12], where we gave GSP and 1-BB non-Moulin mechanisms for every symmetric cost function (costs only depend on the cardinality of the set) for three players. We further observed in [12] that for 4 players, symmetry is not sufficient for the existence of GSP and 1-BB mechanisms. We do not elaborate on these results, as this goes beyond the scope of this thesis.

In the case that only non-negative bids are allowed, Immorlica et al. [73] were the first to note that there are trivial mechanisms that are GSP and 1-BB for every cost function. According to an arbitrary order, the mechanisms find the first player that can pay for himself and *all* the players behind him in that order. The whole burden of paying the cost is put on this player while the subsequent players receive the service for free. Next to their inequity, these mechanisms also yield the unrealistic property that players have in general no choice to refuse the service. If we allow negative bids to enable that choice, these mechanisms can only be guaranteed to be SP. Penna and Ventre [108] pursue a similar approach for STEINERTREE cost-sharing problems which exhibits the same unsatisfactory properties.

These sparse results suggest that our initial questions seem exceedingly difficult to answer. To tackle these questions, we summarize what can be concluded from the rare characterizations of GSP mechanisms.

Moulin [97] showed that cost shares computed by a GSP mechanism are solely dependent on the set of served players (confer Theorem C.1, p. 119). Thus, the 'new' mechanisms have to induce cost-sharing methods. However, these cost-sharing methods must not be cross-monotonic, as Moulin [97] established that GSP mechanisms employing cross-monotonic cost shares are equivalent to Moulin mechanisms (confer Corollary C.4, p. 123).

A result by Immorlica et al. [73] in particular implies that a GSP mechanism yields cross-monotonic cost shares if and only if it is *upper-continuous*, i.e., if for every player i it holds that if i gets the service for every bid value greater than x holding other bids fixed, then i gets the service if he bids x. They also showed that a 1-BB and GSP mechanism yields cross-monotonic cost shares if and only if the computed cost shares lie in the core. Hence, the 'new' mechanisms must violate *upper-continuity* and, if 1-BB is aspired, must not lie in the core.

Penna and Ventre [109] characterize upper-continuous and GSP mechanisms that offer no opportunity for players to refuse the service.

The characterization by Moulin [97] tells us that for submodular costs, one has to resort to mechanisms that are not 1-BB. In case that costs are *supermodular*, meaning that the marginal cost of adding a player set S to a player set T is never less than the marginal cost of adding S to $S \cap T$, the sequential stand alone mechanisms can be adapted to be 1-BB and GSP. Due to their unsatisfactory efficiency and the fact that (Q||$\mathrm{C_{max}}$) cost-sharing problems do not induce supermodular costs in general (see Appendix B.1, p. 111), we do not further follow this approach.

Relaxing GSP

The scarcity of contributions to finding GSP non-Moulin mechanisms with reasonable performance or characterizing them, respectively, suggests that these tasks are intrinsically complicated. An alternative is to consider relaxations of GSP.

Mehta et. al [94] proposed *acyclic mechanisms* that strictly generalize Moulin mechanisms and outperform Moulin mechanisms in many cases (confer Table A.3, p. 110). The price they pay is the deterioration of performance with respect to incentive-compatibility as acyclic mechanisms can only be guaranteed to be *weakly* GSP (WGSP), meaning that only those coalitions are prevented that make all of its members better off. Specific acyclic mechanisms were given by Devanur et al. [31] for SetCover and FacilityLocation cost-sharing problems. At the time when they were introduced, only SP was established. Mehta et al. [94] reconsidered these mechanisms and obtained the efficiency approximations in Table A.3 on page 110.

Further restricting incentive-compatibility to SP, 1-BB and SP can always be achieved by the sequential stand alone mechanisms (lacking good efficiency approximations). On the other hand, 1-EFF and SP cost-sharing mechanisms can be derived from the seminal *Groves mechanisms* [25, 63, 131] which are essentially the *only* 1-EFF mechanisms among *all* SP mechanisms [60]. The main two drawbacks of Groves mechanisms are that they cannot guarantee a reasonable budget-balance approximation [61], and that they are generally not GSP [62] (confer Appendix B.2, p. 114, Example B.7, p. 116, Example B.8, p. 117, and Example B.9, p. 117).

General Demand Cost-Sharing

For general-demand cost-sharing, incremental and acyclic mechanisms have been generalized [94, 97]. However, incremental mechanisms are only known to be GSP for costs that are either sub- or supermodular (in a generalized way). Specifically, this questions their applicability to general demand cost-sharing when already the restricted binary demand problems do not induce sub- or supermodular costs. This is in particular the case for FaultTolerantFL and GenSteinerForest cost-sharing problems (confer Appendix B.1, p. 111).

For each FaultTolerantFL cost-sharing problem, Mehta et al. [94] gave $O(L^2)$-BB and $O(L^2 \cdot (1 + \log n))$-EFF generalized acyclic mechanisms, where L is the maximum connectivity requirement. In the full version of their paper, they further gave H_n-BB and $(H_n \cdot (H_n + H_L + 1))$-EFF generalized acyclic mechanisms for the non-metric case[4].

[4] These results are adjusted to our notion of β-BB, confer Section 4.5.

1.4 Summary of Results

Next to developing and generalizing Moulin mechanisms, we tackle one of the most important tasks for present cost-sharing: designing non-Moulin mechanisms with comparable or even better performance than Moulin mechanisms.

- We investigate what can be achieved for symmetric costs if we hold up the demand for GSP (published in [12]). Already for 4 players, 1-BB is generally infeasible. We omit this result and rather show that $\frac{\sqrt{17}+1}{4}$-BB can be obtained if costs are subadditive as well, where $\frac{\sqrt{17}+1}{4} \approx 1.28$ (Theorem 5.14, p. 55). Notably, we identify symmetric and subadditive costs for which no Moulin mechanism can be better than 2-BB (Theorem 4.7, p. 35). For any set of served players, our $\frac{\sqrt{17}+1}{4}$-BB mechanisms charge at most two different cost shares and are polynomial-time computable if the cost function can be evaluated in polynomial time. We further show that $\frac{\sqrt{17}+1}{4}$-BB is tight when using our technique (Theorem 5.17, p. 58).
  Generally, we give mechanisms for symmetric costs that use so-called *two-price cost-sharing forms* (2P-CSFs) and show that our mechanisms are GSP if the employed 2P-CSFs meet a certain validity requirement (Theorem 5.12, p. 54). Moreover, polynomial-time computability of a 2P-CSF implies polynomial-time computability of the corresponding mechanism (Lemma 5.13, p. 55).
  The drawback of our framework is that it is generally only applicable if efficiency is only a secondary goal, as we identify symmetric and subadditive costs for which the corresponding mechanism is no better than $\Omega(n)$-EFF (Lemma 5.20, p. 60).
- We break new grounds on assumptions for coalition forming by proposing that coalitions are unlikely to form if some member would lose service and are likely to form if at least one member wins service, even when paying his true valuation (published in [11]). We term mechanisms that prevent coalitions with respect to this behavior *GSP against collectors* (CGSP) (confer Definition 6.1, p. 68).
  We show that in particular, CGSP is incomparable to GSP, and that CGSP is stronger than all relaxations of GSP considered so far (Lemma 6.3, p. 68). Surprisingly, we show that already a relaxation of both CGSP and GSP induces a unique cost-sharing method (Theorem 6.4, p. 70). This strictly improves on the characterization result by Moulin [97] (confer Theorem C.1, p. 119).
  Utilizing the idea of [34] for computing the egalitarian solution for submodular costs, we give *egalitarian* mechanisms that are CGSP and 1-BB for arbitrary costs (Lemma 6.9, p. 74 and Theorem 6.10, p. 75). We establish their $2H_n$-EFF in case that costs are subadditive (thus in particular for submodular costs) and show that this is tight up to a factor of 2 (Theorems 6.18, p. 79 and 6.23, p. 80).
  On the other hand, approximability of social cost efficiency for arbitrary costs is infeasible (Lemma 6.24, p. 80). However, the cost function that provides this impossibility result is rather unnatural as costs 'explode'. We extract a condition on the cost shares that quantifies efficiency loss and thus may be helpful for investigating the approximate social cost efficiency of egalitarian mechanisms for reasonable cost functions in the future (confer Theorem 6.21, p. 79).
  We further provide a framework for polynomial-time computability of egalitarian mechanisms in case that each player is endowed with some size. The clue here are *monotonic* cost functions that do not increase when one player is substituted by another player with smaller size (see Section 6.7, p. 80).

- We apply Moulin mechanisms and our new mechanisms to $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems and their subproblems[5]. For their β-BB and γ-EFF, see Table 1.1.

Table 1.1. BB and EFF guarantees of best known polynomial-time mechanisms for $(\mathrm{Q}||\mathrm{C}_{\max})$, EFF and BB entries in one 'box' are simultaneously achieved by one mechanism.

	GSP Moulin Mech.		**GSP 2P-CSF Mech.**	**CGSP Egalitarian Mech.**	
Problem	BB	EFF	BB	BB	EFF
$(\mathrm{Q}\|\|\mathrm{C}_{\max})$	$\mathbf{2d}$	$\mathbf{2d \cdot (1+H_n)}$	$\frac{\sqrt{\mathbf{17}}+\mathbf{1}}{\mathbf{4}} \cdot \mathbf{d}$	$\mathbf{2}$	$\mathbf{4H_n}$
	(Thm. 4.14, p. 38), [14]		(Lem. 5.21, p. 60), [12]	(Thm. 6.34, p. 86), [11]	
$(\mathrm{P}\|\|\mathrm{C}_{\max})$[1]	$\frac{2m}{m+1}$	$\Omega(n)$	$\frac{\sqrt{\mathbf{17}}+\mathbf{1}}{\mathbf{4}} \cdot \mathbf{d}$	$\mathbf{1+\varepsilon}$	$\mathbf{2(1+\varepsilon)H_n}$
	[10]	[16]	(Lem. 5.21, p. 60), [12]	(Thm. 6.35, p. 86), [11]	
	$\frac{2m-1}{m}$	$\frac{2m-1}{m} \cdot (1+H_n)$		$\mathbf{\frac{4}{3}-\frac{1}{3m}}$	$\mathbf{2(\frac{4}{3}-\frac{1}{3m})H_n}$
	[16]	[16]		(Thm. 6.32, p. 84), [11]	
$(\mathrm{Q}\|p_i{=}1\|\mathrm{C}_{\max})$	$\mathbf{\frac{2m}{m+1}}$	$\mathbf{\frac{2m}{m+1} \cdot (1+H_n)}$	$\frac{\sqrt{\mathbf{17}}+\mathbf{1}}{\mathbf{4}}$	$\mathbf{1}$	$\mathbf{2H_n}$
	(Thm. 4.12, p. 36), [14]		(Thm. 5.18, p. 59), [12]	(Lem. 6.36, p. 86), [11]	
$(\mathrm{P}\|p_i{=}1\|\mathrm{C}_{\max})$	$\mathbf{\frac{2m}{m+1}}$	$\mathbf{\frac{2m}{m+1} \cdot (1+H_n)}$	$\mathbf{1}$	$\mathbf{1}$	$\mathbf{2H_n}$
	(Thm. 4.12, p. 36), [14]		(Lem. 5.19, p. 59), [12]	(Lem. 6.36, p. 86), [11]	

n, m, d, H_n: number of jobs, machines, different processing times, n-th harmonic number
[1] CGSP mechanisms: Upper result based on PTAS with running time exponential in $\frac{1}{\varepsilon}$

  - A fine grained analysis for our Moulin mechanisms for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ establishes $O(1)$-EFF for many cases (Theorem 4.16, p. 39). We further show in Section 4.6.1 that no Moulin mechanism for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems can be better than $(\max\{d, \frac{2m}{m+1}\})$-BB and $(\max\{d, H_n\})$-EFF in general.
  - We generalize our 2P-CSFs and obtain GSP and 1-BB for a scheduling problem with non-symmetric costs (see Section 5.6.2).
  - The β-BB with $\beta > 1$ for egalitarian mechanisms is due to polynomial-time computability. In addition, the egalitarian mechanisms for $(\mathrm{Q}||\mathrm{C}_{\max})$ stem from 2-BB and $4H_n$-EFF egalitarian mechanisms for the related BinPacking cost-sharing problems (Theorem 6.33, p. 85). We further obtain polynomial-time 1-BB and $2H_n$-EFF egalitarian mechanisms for other scheduling models.
  - Despite the devastating bound of d for the β-BB of cross-monotonic cost-sharing methods for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems, we give cost-sharing methods in the $\frac{2m}{m+1}$-core (Theorem 4.17, p. 42), which is the best possible (Theorem 4.8, p. 36).
- For general demand, we generalize Moulin mechanisms and provide a general framework for designing GSP mechanisms (published in [13], see Section 7.3). For FaultTolerantFL cost-sharing problems, these mechanisms strictly improve on the results by Mehta et al. [94] by tightening *weak* GSP to GSP and reducing approximate budget-balance and efficiency each by a factor of L (the maximum connectivity requirement) (Theorem 7.13, p. 96). We are the first to consider GenSteinerForest cost-sharing problems, and give GSP mechanisms that are $O(\log L)$-BB and $O(\log^2 n \cdot \log L)$-EFF (Theorem 7.23, p. 104). For both problems, our mechanisms are polynomial-time computable.

[5] $(\mathrm{P}||\mathrm{C}_{\max})$: identical speeds of machines, $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$: identical job-processing times, $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$: both identical speeds and processing times (confer Section 3.2.1)

1.5 Publications

The results described in this thesis are published in parts as joint work in the *Proceedings of the 6th Italian Conference on Algorithms and Complexity* (CIAC'06) [10], the *Proceedings of the 32th International Symposium on Mathematical Foundations of Computer Science* (MFCS'07) [12], the *Proceedings of the 3rd International Workshop on Internet and Network Economics* (WINE'07) [11], the *Elsevier Information Processing Letters* (IPL'08) [14], and the *Proceedings of the 1st International Symposium on Algorithmic Game Theory* (SAGT'08) [13].

The work from [10] has also been accepted for publication in the *Elsevier Journal of Discrete Algorithms*.

1.6 Organization

Chapter 2 explains the general cost-sharing model and gives the necessary definitions. The cost-sharing problems with their underlying optimization problems that we consider in this thesis are presented in Chapter 3 together with related work besides cost sharing, as well as properties that we exploit within the proofs.

Our results are arranged in Chapters 4 to 7. Chapter 4 covers Moulin mechanisms, Chapter 5 is concerned with mechanisms on two-price cost-sharing forms, Chapter 6 examines egalitarian mechanisms, and Chapter 7 deals with generalizing Moulin mechanisms for general demand cost-sharing.

We provide further information in the appendix. Particularly, Appendix A provides a complete overview of existing Moulin and acyclic mechanisms. With respect to other mechanisms, we introduce incremental mechanisms in Appendix B.1, and give consideration to the significance of Groves mechanisms for mechanism design in Appendix B.2, where we stress their applicability to cost-sharing. Appendix C provides a revised proof of Moulin's theorem in [97] adapted to our notation, stating that for submodular costs, Moulin mechanisms are essentially the only 1-BB and GSP mechanisms, and particularly implying that GSP mechanisms with cross-monotonic cost shares are equivalent to Moulin mechanisms. Finally, Appendix D defines further optimization problems that we discuss in the related work section and that are stated in the tables in Appendix A.

2

The Model

2.1 Organization

Section 2.2 states the notational conventions used throughout this thesis. Most of the results presented in this thesis are for *binary demand* cost sharing where players are 'served' or 'not served'. Therefore, we first introduce the restricted model for binary demand in Section 2.3. We give the necessary adaptations to *general demand* cost sharing where players receive *service levels* in Section 2.4.

2.2 Notation

Let $n \in \mathbb{N}$. We define $[n] := \{1, \ldots, n\}$, $[n]_0 := \{0, \ldots, n\}$, and denote the n-th harmonic number by $H_n := \sum_{i=1}^{n} \frac{1}{i} \in (\log n, 1 + \log n)$.

- Sets:
  - The function $\mathsf{in} : 2^{[n]} \to \{0, 1\}^n$ indicates membership in a set, where for all $S \subseteq [n]$ and all $i \in [n]$, $\mathsf{in}_i(S) = 1 \Leftrightarrow i \in S$.
  - The *rank* of an element $i \in S \subseteq [n]$ is given by $\mathsf{rank}(i, S) := |\{j \in S \mid j \leq i\}|$.
  - For $S \subseteq [n]$ and $k \in [|S|]$, $\mathit{MIN}_k\, S$ is the set of the k smallest elements in S.
- Vectors:
  - The vectors $\mathbf{0}, \mathbf{1}$, and $\boldsymbol{e}_i$ denote the zero, one, and i-th standard basis vector (dimension will be clear from the context).
  - For $\boldsymbol{x} \in \mathbb{R}^n$ and $S \subseteq [n]$, we define $x(S) := \sum_{i \in S} x_i$.
  - For vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we write $\boldsymbol{x} \geq \boldsymbol{y}$ if for all $i \in [n]$, $x_i \geq y_i$.
  - Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and $S \subseteq [n]$, we define $\boldsymbol{x}_S := (x_i)_{i \in S} \in \mathbb{R}^{|S|}$ and $\boldsymbol{x}_{-S} := \boldsymbol{x}_{[n] \setminus S}$. We define $(\boldsymbol{x}_S, \boldsymbol{y}_{-S}) \in \mathbb{R}^n$ to denote the vector containing the components of $\boldsymbol{x}$ for S and the components of $\boldsymbol{y}$ for $[n] \setminus S$.
  - We restrict a vector $\boldsymbol{x} \in \mathbb{N}_0^n$ to values of at most $j \in \mathbb{N}$ by defining $\boldsymbol{x}^{\leq j} \in [j]_0^n$, where $x_i^{\leq j} := \min\{x_i, j\}$ for all $i \in [n]$.
  - We define $\boldsymbol{x}^j := \mathsf{in}(\{i \in [n] \mid x_i \geq j\})$, indicating all entries of $\boldsymbol{x}$ with value at least j.

2.3 Binary Demand Cost Sharing

There are n players, numbered from 1 to n, i.e., the set of players is $[n]$. We denote the *true valuation* of player $i \in [n]$ by $v_i \in \mathbb{R}$, and let $b_i \in \mathbb{R}$ be the actual *bid* submitted by player i. Players' true valuations are private information. We call $\boldsymbol{v} \in \mathbb{R}^n$ the *valuation vector* and $\boldsymbol{b} \in \mathbb{R}^n$ the *bid vector*.

2.3.1 Cost-Sharing Mechanisms

A bid vector $\boldsymbol{b} \in \mathbb{R}^n$ constitutes the input for a cost-sharing mechanism:

Definition 2.1. *A* cost-sharing mechanism

$$M = (Q, x)$$

is a pair of functions $Q : \mathbb{R}^n \to 2^{[n]}$ and $x : \mathbb{R}^n \to \mathbb{R}^n_{\geq 0}$, where

- $Q(\boldsymbol{b}) \in 2^{[n]}$ *is the set of players to be served and*
- $x(\boldsymbol{b}) \in \mathbb{R}^n_{\geq 0}$ *is the vector of cost shares.*

There are three standard properties of cost-sharing mechanisms:

Definition 2.2. *A cost-sharing mechanism $M = (Q, x)$ meets*

- no positive transfers *(NPT), if players never get paid:*

$$\text{for all } \boldsymbol{b} \in \mathbb{R}^n : x(\boldsymbol{b}) \geq \mathbf{0}$$

- voluntary participation *(VP), if players never pay more than they bid and are only charged when served:*

$$\text{for all } \boldsymbol{b} \in \mathbb{R}^n, i \in [n] : \big(i \in Q(\boldsymbol{b})\big) \Rightarrow \big(x_i(\boldsymbol{b}) \leq b_i\big) \text{ and } \big(i \notin Q(\boldsymbol{b})\big) \Rightarrow \big(x_i(\boldsymbol{b}) = 0\big)$$

- consumer sovereignty *(CS), if for every player $i \in [n]$ there is a threshold bid $b_i^+ \in \mathbb{R}_{\geq 0}$ such that i is served if bidding at least b_i^+, regardless of the other bids:*

$$\text{for all } i \in [n] \text{ there is a } b_i^+ \in \mathbb{R}_{\geq 0} \text{ such that for all } \boldsymbol{b} \in \mathbb{R}^n : i \in Q(b_i^+, \boldsymbol{b}_{-i})$$

Note that in our model, VP and NPT imply that players may opt to not participate (by submitting a negative bid). This property together with CS is referred to as *strict* CS. An alternative approach to ensure strict CS when negative bids are not allowed is to require that payments are always positive. The economic term for this requirement is 'no free riders'.

Assumption 2.1. We assume that all cost-sharing mechanisms referred to within this thesis meet VP, NPT and strict CS, unless stated otherwise.

We consider *quasi-linear* utilities derived from the output of a mechanism:

Definition 2.3. *The utility function $u_i : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$* of player $i \in [n]$ is given by

$$u_i(\boldsymbol{b}, v_i) := v_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - x_i(\boldsymbol{b}) \ .$$

The second argument of u_i denotes the true valuation of player i. For $b_i \neq v_i$, $u_i((v_i, \boldsymbol{b}_{-i}), v_i)$ and $u_i(\boldsymbol{b}, v_i) = u_i((b_i, \boldsymbol{b}_{-i}), v_i)$ correspond to i's utility for truth-telling and for submitting the untruthful bid b_i, respectively. We remark that our definition of utilities assumes a given mechanism $M = (Q, x)$. More generally, we could define utilities independent of mechanisms by $u_i' : 2^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ where $u_i'(Q, \boldsymbol{x}, v_i) = v_i \cdot \mathsf{in}_i(Q) - x_i$. Then, $u_i(\boldsymbol{b}, v_i)$ corresponds to $u_i'(Q(\boldsymbol{b}), x(\boldsymbol{b}), v_i)$. As we mostly consider utilities in conjunction with mechanisms, we use Definition 2.3 for notational convenience.

Definition 2.4 introduces *welfare equivalence* of two mechanisms. In particular, if two mechanisms are welfare-equivalent and are assumed to receive true valuations, each player derives the same utility from both mechanisms.

Definition 2.4. *Two cost-sharing mechanisms $M = (Q, x)$ and $M' = (Q', x')$ are* welfare equivalent *if for all bid vectors $\boldsymbol{b} \in \mathbb{R}^n$ and all players $i \in [n]$ it holds that $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - x_i(\boldsymbol{b}) = b_i \cdot \mathsf{in}_i(Q'(\boldsymbol{b})) - x_i'(\boldsymbol{b})$.*

2.3.2 Incentive-Compatibility

Besides fulfilling the mandatory NPT, VP, and strict CS, cost-sharing mechanisms should create incentives for all players to tell the truth out of self-interest. The most common notions of incentive-compatibility are *strategyproofness, group-strategyproofness,* and *weak group-strategyproofness.* A cost-sharing mechanism is strategyproof if no player can strictly increase his utility by misreporting his valuation, independent of the bids submitted by the other players:

Definition 2.5. *A mechanism is* strategyproof *(SP) if for every player i and every true valuation $v_i \in \mathbb{R}$ there is no bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that $u_i(\boldsymbol{b}, v_i) > u_i((v_i, \boldsymbol{b}_{-i}), v_i)$.*

A stronger notion of incentive compatibility is group-strategyproofness that even prevents manipulation by coalitions. A mechanism is group-strategyproof if no coalition of players can jointly misreport (some or all of) their valuations such that this strictly increases the utility of at least one of its members and does not strictly decrease the utility of any other member, independent of the bids from non-coalitional players:

Definition 2.6. *A mechanism is* group-strategyproof *(GSP) if for every coalition $K \subseteq [n]$ and every true valuation vector $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$, there is no bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that*

- *$u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for all $i \in K$ and*
- *$u_i(\boldsymbol{b}, v_i) > u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for at least one $i \in K$.*

A relaxation of GSP which is still stronger than SP is the notion of weak GSP. A mechanism is weakly GSP if no coalition of players can jointly misreport (some or all of) their valuations such that this strictly increases the utility of *all* of its members, independent of the bids from non-coalitional players:

Definition 2.7. *A mechanism is* weakly GSP *(WGSP) if for every coalition $K \subseteq [n]$ and every true valuation vector $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$, there is no bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that*

- *$u_i(\boldsymbol{b}, v_i) > u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for all $i \in K$.*

2.3.3 The Service Cost

Incentive compatibility on its own is not sufficient for a cost-sharing mechanism, since it does not provide performance guarantees with respect to the cost of serving the selected players. We specify this cost by the cost function $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$ and require that $C(\emptyset) = 0$ and that C is non-decreasing, i.e., for all $S, T \subseteq [n]$ with $T \subseteq S$ it holds that $C(T) \leq C(S)$. We define some properties of cost functions that are relevant within this thesis:

Definition 2.8. *A cost function C is* submodular *if for any two sets $S, T \subseteq [n]$ it holds that the marginal cost of adding a set of players S to a set of players T is never more than the marginal cost of adding S to the set $S \cap T$:*

$$C(S \cup T) - C(T) \leq C(S) - C(S \cap T)$$

Definition 2.9. *A cost function C is* supermodular *if for any two sets $S, T \subseteq [n]$ it holds that the marginal cost of adding a set of players S to a set of players T is never less than the marginal cost of adding S to the set $S \cap T$:*

$$C(S \cup T) - C(T) \geq C(S) - C(S \cap T)$$

Definition 2.10. *A cost function C is* subadditive *if for any two sets $S, T \subseteq [n]$ it holds that the marginal cost of adding a set of players S to a set of players T is never more than the stand-alone cost of the set S:*

$$C(S \cup T) - C(T) \leq C(S)$$

Definition 2.11. *A cost function C is* symmetric *if for any two sets $S, T \subseteq [n]$ it holds that the cost of a set S only depends on its cardinality $|S|$:*

$$\big(|S| = |T|\big) \Longrightarrow \big(C(S) = C(T)\big)$$

For a symmetric cost function C we use the respective lower case letter to define $c : [n] \to \mathbb{R}_{\geq 0}$ by $c(i) := C(S)$ for every $i \in [n]$ and an arbitrary $S \subseteq [n]$ with $|S| = i$.

For applications, costs typically stem from solutions to a combinatorial minimization problem and are defined only implicitly (confer Examples 1.1– 1.3). In the following, $C(S)$ denotes the value of a minimum-cost solution for the instance induced by the player set S. Since computing *optimal* costs may take exponential time, the service provider therefore resorts to approximate solutions with *actual* costs $C'(S)$. Section 2.3.6 explains in detail how these costs are derived from optimization problems.

2.3.4 Budget-Balance

The *budget-balance* performance measure relates the overall cost share to the cost functions C and C'. We want a mechanism $M = (Q, x)$ to recover the cost $C'(Q(\boldsymbol{b}))$ incurred by serving the selected players $Q(\boldsymbol{b})$, while at the same time the overall cost-share should be reasonably bounded with respect to the optimal cost $C(Q(\boldsymbol{b}))$ of serving $Q(\boldsymbol{b})$:

Definition 2.12. *A mechanism $M = (Q, x)$ is β-*budget-balanced *(β-BB) for $\beta \in \mathbb{R}_{\geq 1}$ and cost functions C and C' if for all $\boldsymbol{b} \in \mathbb{R}^n$ it holds that*

$$C'(Q(\boldsymbol{b})) \leq \sum_{i \in Q(\boldsymbol{b})} x_i(\boldsymbol{b}) \leq \beta \cdot C(Q(\boldsymbol{b})) \ .$$

We say that a mechanism is *budget-balanced* if it is 1-BB. In that case, $C' = C$.

We remark that an alternative definition of β-BB has been used in some works (e.g. [14, 19, 94, 113–115]), where $\frac{1}{\beta} \cdot C'(Q(\boldsymbol{b})) \leq \sum_{i \in Q(\boldsymbol{b})} x_i(\boldsymbol{b}) \leq C(Q(\boldsymbol{b}))$ is required.

2.3.5 Efficiency

With respect to the *efficiency* performance measure, we introduce two established ways to quantify efficiency loss: the *social welfare* objective and the *social cost* objective. As both objectives are defined on the true valuations of the players, a mechanism has to rely on receiving truthful bids in order to meet these objectives. Naturally, the question if all players can be assumed to bid truthfully within a specific scenario depends on the underlying interpretation of incentive-compatibility.

A mechanism is *social welfare efficient* for a cost function C if, assuming truthful bids, it always selects a set S of players that maximizes $\sum_{i \in S} \max\{v_i, 0\} - C(S)$. It is *social cost efficient* for C if, assuming truthful bids, it always selects a set of players that minimizes $C(S) + \sum_{i \notin S} \max\{v_i, 0\}$. Formally, for a set $S \subseteq [n]$, true valuations $\boldsymbol{v}$, and a cost function C, define the *social welfare* $SW_C(S, \boldsymbol{v})$ by

$$SW_C(S, \boldsymbol{v}) := \sum_{i \in S} \max\{v_i, 0\} - C(S) \ .$$

Correspondingly, we define the *social cost* $SC_C(S, \boldsymbol{v})$ to be

$$SC_C(S, \boldsymbol{v}) := C(S) + \sum_{i \notin S} \max\{v_i, 0\} \ .$$

Definition 2.13. *A mechanism $M = (Q, x)$ is* social welfare efficient *for cost function C if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all sets $S \subseteq [n]$ it holds that*

$$SW_C(Q(\boldsymbol{v}), \boldsymbol{v}) \geq SW_C(S, \boldsymbol{v}) \ .$$

Definition 2.14. *A mechanism $M = (Q, x)$ is* social cost efficient *for cost function C if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all sets $S \subseteq [n]$ it holds that*

$$SC_C(Q(\boldsymbol{v}), \boldsymbol{v}) \leq SC_C(S, \boldsymbol{v}) \ .$$

Since $SC_C(S, \boldsymbol{v}) = \sum_{i \in [n]} \max\{v_i, 0\} - SW_C(S, \boldsymbol{v})$ for every cost function C, every $\boldsymbol{v} \in \mathbb{R}^n$, and every $S \subseteq [n]$, a subset maximizes the social welfare if and only if it minimizes the social cost. We refer to social welfare efficient (and thus social cost efficient) mechanisms simply as *efficient* mechanisms.

In this thesis, we consider approximating the social cost objective:

Definition 2.15. *A mechanism $M = (Q, x)$ is γ-*social cost efficient *(γ-EFF) for $\gamma \in \mathbb{R}_{\geq 1}$ and cost functions C and C' if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all subsets $S \subseteq [n]$ it holds that*

$$SC_{C'}(Q(\boldsymbol{v}), \boldsymbol{v}) \leq \gamma \cdot SC_C(S, \boldsymbol{v}) \ .$$

2.3.6 Cost-Sharing Problems

A cost-sharing problem in the classical economic sense is simply specified by a cost function C. The aim is to define a mechanism with strong incentive-compatibility and good performance with respect to β-BB and γ-EFF, where $C' = C$ in Definitions 2.12 and 2.15. In a sense, a cost-sharing problem is a multiobjective optimization problem, and an optimal solution is a mechanism that is GSP, 1-BB, and 1-EFF (which we have discussed to be generally infeasible in Section 1.3).

Definition 2.16. *A binary demand* cost-sharing problem *is specified by a cost function $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$. The aim is to give a cost-sharing mechanism M such that*

1. *M is incentive-compatible in a sense as strong as possible.*
2. *M is β-BB for C, $C' = C$, and $\beta \in \mathbb{R}_{\geq 1}$ as small as possible.*
3. *M is γ-EFF for C, $C' = C$, and $\gamma \in \mathbb{R}_{\geq 1}$ as small a possible.*

In this thesis we especially consider cost-sharing problems for which C is only implicitly defined by an underlying optimization problem. Furthermore, we seek for polynomial-time computability of cost-sharing mechanisms as well as solutions to provide the service to players. The latter requires the application of approximation algorithms where the actual costs C' come into play.

In order to clarify how the cost functions C and C' are actually defined, consider an optimization problem Π whose instances are partially defined by a set of players. Let $(I_S)_{S\subseteq[n]}$ be a tuple of 2^n instances of Π that only differ from each other by the set of players. All other parameters, especially player-specific data, are fixed. For every $S \subseteq [n]$, we define $C(S)$ to be the value of a minimum cost solution for instance I_S. Given an approximation algorithm ALG for Π, we further define $\mathsf{ALG}(I_S)$ to be the solution computed by ALG for instance I_S and $C_{\mathsf{ALG}}(S)$ to be the value of this solution. Then, $C'(S) := C_{\mathsf{ALG}}(S)$ for all $S \subseteq [n]$.

Definition 2.17. *A binary demand Π-*cost-sharing problem *for an optimization problem Π is specified by a tuple of instances $(I_S)_{S\subseteq[n]}$ of Π. The aim is to give a cost-sharing mechanism M and an approximation algorithm ALG such that for C and C_{ALG} defined by $(I_S)_{S\subseteq[n]}$,*

1. *M is incentive-compatible in a sense as strong as possible.*
2. *M is β-BB for C, C_{ALG}, and $\beta \in \mathbb{R}_{\geq 1}$ as small as possible.*
3. *M is γ-EFF for C, C_{ALG}, and $\gamma \in \mathbb{R}_{\geq 1}$ as small a possible.*
4. *M and ALG are polynomial-time computable in the size of the* succinct *representation of $(I_S)_{S\subseteq[n]}$.*

We conclude this subsection with a note on the terminology. When considering a cost-sharing problem specified by cost function C, we say that a mechanism M is β-BB for C meaning that M is β-BB in the sense of Definition 2.12 with $C' = C$. When considering a Π-cost-sharing problem specified by $(I_S)_{S\subseteq[n]}$, the cost function C immediately results from $(I_S)_{S\subseteq[n]}$. Thus, given an algorithm ALG for Π, we simply say that a mechanism M is β-BB for C_{ALG} meaning that M is β-BB in the sense of Definition 2.12 for C and $C' = C_{\mathsf{ALG}}$. Furthermore, when we do not want to specify the approximation algorithm, we only say that M is β-BB. We adapt this terminology to γ-EFF as well. Furthermore, we use it for β-BB and the β-core for cost-sharing methods introduced in Section 2.3.7.

2.3.7 Cost-Sharing Methods

For the decision process of selecting players and determining their payments in particular, many mechanisms employ cost-sharing methods:

Definition 2.18. *A* cost-sharing method *is a function $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ that maps each set of players $S \subseteq [n]$ to a vector of cost shares $\xi(S)$, where for all $S \subseteq [n]$ and all $i \notin S$ it holds that $\xi_i(S) = 0$.*

Notably, for any GSP mechanism $M = (Q, x)$ we can define a unique cost-sharing method ξ by setting $\xi(S) := x(\boldsymbol{b})$ for $\boldsymbol{b} \in \mathbb{R}$ with $b_i = b_i^+$ if $i \in S$ and $b_i < 0$ otherwise. This can be deduced from a result by Moulin [97] (and a more general result in this thesis), stating that for all $\boldsymbol{b}, \boldsymbol{b}'$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$ it holds that $x(\boldsymbol{b}) = x(\boldsymbol{b}')$ (confer Theorems 6.4 and C.1).

β-BB of cost-sharing mechanisms directly translates to cost-sharing methods:

Definition 2.19. *A cost-sharing method ξ is β-*budget-balanced *(β-BB) for $\beta \in \mathbb{R}_{\geq 1}$ and cost functions C and C' if for all $S \subseteq [n]$ it holds that*

$$C'(S) \leq \sum_{i\in S} \xi_i(S) \leq \beta \cdot C(S) \ .$$

In terms of stability and fairness there are two main attributes of cost-sharing methods, the *β-core property* and *cross-monotonicity*. The β-core requires that the actual cost is recovered while no coalition jointly pays more than β times the cost of serving only the coalition:

Definition 2.20. *A cost-sharing method ξ is in the β*-core *for $\beta \in \mathbb{R}_{\geq 1}$ and cost functions C and C' if for all $S, T \subseteq [n]$ with $T \subseteq S$ it holds that*

- $\sum_{i \in S} \xi_i(S) \geq C'(S)$ *and*
- $\sum_{i \in T} \xi_i(S) \leq \beta \cdot C(T)$.

Cross-monotonicity requires that the cost-share charged to any player in a group does not increase as the group expands:

Definition 2.21. *A cost-sharing method $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ is* cross-monotonic *if for all $S, T \subseteq [n]$ and all $i \in [S]$ it holds that $\xi_i(S \cup T) \leq \xi_i(S)$.*

The rather abstract *α-summability* property was introduced as an important tool that along with the approximate budget-balance of a cost-sharing method characterizes the approximate social cost efficiency of the corresponding Moulin mechanisms (confer Section 4.5).

Definition 2.22. *A cost-sharing method $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ is α*-summable *(α-SUM) for $\alpha \in \mathbb{R}_{\geq 0}$ and cost function C if for all $S \subseteq [n]$ and every order $i_1, \ldots, i_{|S|}$ of S with $S_j := \{i_1, \ldots, i_j\}$ it holds that $\sum_{j=1}^{|S|} \xi_{i_j}(S_j) \leq \alpha \cdot C(S)$.*

2.4 General Demand Cost Sharing

For general demand, each player $i \in [n]$ has a *maximum level of service* $L_i \in \mathbb{N}$ he can receive. We define $L := \max_{i \in [n]}\{L_i\}$ to be the *maximum service level.* We call a representation of service levels for each player *allocation.*

Definition 2.23. *We define $\mathcal{A} := [L_1]_0 \times \ldots \times [L_n]_0$ to denote the* allocation space. *An* allocation *is a vector $\boldsymbol{a} \in \mathcal{A}$.*

In our model, each player $i \in [n]$ submit his bids for the different levels as a *bid vector* $\boldsymbol{b}_i \in \mathbb{R}^{L_i}$ consisting of the *marginal bids* $b_{i,\ell}$ of receiving level ℓ additionally to level $\ell - 1$. The vector $\boldsymbol{v}_i$ of true valuations (composed of the *marginal valuations*) is represented analogously. The overall true valuation of i for level k is then $\sum_{\ell=1}^{k} v_{i,\ell}$. Our work is based on the common assumption of *diminishing marginal returns*, i.e., marginal valuations are non-increasing in the service level:

Assumption 2.2. For all $i \in [n]$ and all $\boldsymbol{v} \in \mathbb{R}^{L_i}$ it holds that $v_{i,1} \geq \ldots \geq v_{i,L_i}$.

Clearly, marginal bids are subject to Assumption 2.2 as well, as otherwise lying would be immediately discovered.

Definition 2.24. *We denote the* bid space *by $\mathcal{R} := \mathbb{R}^{L_1} \times \ldots \times \mathbb{R}^{L_n}$.*

We call $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n) \in \mathcal{R}$ the *valuation matrix* and $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) \in \mathcal{R}$ a *bid matrix.*

Obviously, cost-sharing mechanisms for general demand have to compute an allocation instead of only a set of served players:

Definition 2.25. *A* general demand cost-sharing mechanism

$$M = (q, x)$$

is a pair of functions $q : \mathcal{R} \to \mathcal{A}$ *and* $x : \mathcal{R} \to \mathbb{R}^n_{\geq 0}$*, where*

- $q(\boldsymbol{B}) \in \mathcal{A}$ *is the allocation according to which players are served, and*
- $x(\boldsymbol{B}) \in \mathbb{R}^n_{\geq 0}$ *is the vector of cost shares.*

NPT, VP and strict CS are generalized as follows:

Definition 2.26. *A general demand cost-sharing mechanism* $M = (q, x)$ *meets*

- *NPT, if for all* $\boldsymbol{B} \in \mathcal{R} : x(\boldsymbol{B}) \geq 0$.
- *VP, if for all* $\boldsymbol{B} \in \mathcal{R}$ *and all* $i \in [n]$: $\big(q_i(\boldsymbol{B}) > 0 \implies x_i(\boldsymbol{B}) \leq \sum_{\ell=1}^{q_i(\boldsymbol{B})} b_{i,\ell}\big)$ *and* $\big(q_i(\boldsymbol{B}) = 0\big) \implies \big(x_i(\boldsymbol{B}) = 0\big)$.
- *strict CS, if for all* $i \in [n]$ *and all* $\ell \in [L_i]_0$*, there is a bid vector* $\boldsymbol{b}_i^{+\ell} \in \mathbb{R}^{L_i}$ *such that for all* $\boldsymbol{B} \in \mathcal{R} : q_i(\boldsymbol{b}_i^{+\ell}, \boldsymbol{B}_{-i}) = \ell$.

Just like for binary demand, we consider quasi-linear utilities:

Definition 2.27. *The utility function* $u_i : \mathcal{R} \times \mathbb{R}^{L_i} \to \mathbb{R}$ *of player* $i \in [n]$ *is of the form*

$$u_i(\boldsymbol{B}, \boldsymbol{v}_i) := \sum_{\ell=1}^{q_i(\boldsymbol{B})} v_{i,\ell} - x_i(\boldsymbol{B}) \ .$$

The definitions of SP, GSP, WGSP, β-BB and γ-EFF naturally carry over to general demand, where cost functions are now given by $C : \mathcal{A} \to \mathbb{R}_{\geq 0}$, and the social cost is now defined by

$$SC_C(\boldsymbol{a}, \boldsymbol{V}) := C(\boldsymbol{a}) + \sum_{i=1}^{n} \sum_{\ell=a_i+1}^{L_i} \max\{0, v_{i,\ell}\} \ .$$

We look at optimization problems Π whose instances are partially defined by an allocation $a \in \mathcal{A}$. Then, a general demand Π-cost-sharing problem is specified by a tuple of instances $(I_{\boldsymbol{a}})_{\boldsymbol{a} \in \mathcal{A}}$ that only differ from each other by the allocation.

2.4.1 Marginal Cost-Sharing Methods

Our general demand mechanisms use *marginal* cost-sharing methods that specify separate cost shares for each service level.

Definition 2.28. *We define* $\mathcal{R}_{\geq 0} := \mathbb{R}^{L_1}_{\geq 0} \times \ldots \times \mathbb{R}^{L_n}_{\geq 0}$ *to denote the* space of marginal cost shares. *A* marginal cost-sharing method *is a function* $\chi : \mathcal{A} \to \mathcal{R}_{\geq 0}$*, where for all* $\boldsymbol{a} \in \mathcal{A}$*, all* $i \in [n]$*, and all* $\ell \in [L_i]$ *with* $\ell > a_i$ *it holds that* $\chi_{i,\ell}(\boldsymbol{a}) = 0$.

We also define β-BB for marginal cost-sharing methods:

Definition 2.29. *A marginal cost-sharing method* χ *is* β*-budget-balanced (*β*-BB) for* $\beta \in \mathbb{R}_{\geq 1}$ *and cost functions* C *and* C' *if for all* $\boldsymbol{a} \in \mathcal{A}$ *it holds that*

$$C'(\boldsymbol{a}) \geq \sum_{i=1}^{n} \sum_{\ell=1}^{a_i} \chi_{i,\ell}(\boldsymbol{a}) \leq \beta \cdot C(\boldsymbol{a}) \ .$$

3

The Cost-Sharing Problems

3.1 Organization

We give the Π-cost-sharing problems and the underlying optimization problems Π examined in this thesis in Section 3.2. Section 3.3 provides related work on optimization and game-theoretic settings other than cost-sharing. Section 3.4 gives relevant properties of solutions to specific scheduling optimization problems.

3.2 The Problems

3.2.1 Scheduling on Related Machines

Given n jobs, m machines, a vector $\boldsymbol{p} \in \mathbb{N}^n$ of processing times p_i for each job $i \in [n]$, and a vector $\boldsymbol{s} \in \mathbb{N}^m$ of speeds s_g for each machine $g \in [m]$, a scheduling instance for $S \subseteq [n]$ is defined by $I_S := (\boldsymbol{p}_S, \boldsymbol{s})$.

Jobs are called *identical* if $\boldsymbol{p} = \mathbf{1}$. Machines are called *identical* if $\boldsymbol{s} = \mathbf{1}$. For arbitrary speeds, machines are termed *related*.

Assumption 3.1. It holds that $p_1 \geq \ldots \geq p_n$ and $s_1 \geq \ldots \geq s_m$.

The aim is to assign the jobs in S to the machines such that a given objective function is minimized. Formally, an *assignment* for $S \subseteq [n]$ is a function $\phi : S \to [m]$, where $\phi(i) = g$ indicates that job i is assigned to machine g. For $T \subseteq S$, we let $m_\phi(T)$ be the set of machines that ϕ uses to assign T:

$$m_\phi(T) := \{\phi(i)\}_{i \in T} \ .$$

We mainly focus on minimizing the maximum *completion time* over all machines, termed *makespan*. Given an assignment $\phi : S \to [m]$, the completion time of $g \in [m]$ is simply $\frac{p(\{i \in S \mid \phi(i) = g\})}{s_g}$. We define the makespan of an optimal assignment for S by

$$\mathsf{MSP}(S) := \min_{\phi} \left\{ \max_{g \in m_\phi(S)} \left\{ \frac{p(\{i \in S \mid \phi(i) = g\})}{s_g} \right\} \right\} \ .$$

In the context of cost-sharing, we assume that the service provider acts as a machine administrator and that each player i owns exactly one job. Thus, a player receives the service if and only if his job is scheduled. Accordingly, we will use $S \subseteq [n]$ to denote players and jobs interchangeably. The succinct representation of $(I_S)_{S \subseteq [n]}$ is $(\boldsymbol{p}, \boldsymbol{s})$. The function $\mathsf{MSP} : 2^{[n]} \to \mathbb{R}_{\geq 0}$ defines the optimal service cost. For identical jobs, MSP is symmetric. In that case we define $\mathsf{msp} : [n] \to \mathbb{R}_{\geq 0}$ by $\mathsf{msp}(i) := \mathsf{MSP}(S)$ for every $i \in [n]$ and an arbitrary $S \subseteq [n]$ with $|S| = i$ (confer Definition 2.11, p. 16).

We remark that we do not further specify how jobs on a specific machine are scheduled. However, in this thesis we marginally consider minimizing the sum of completion times of *jobs*, where such a specification is necessary. The completion time of a job i assigned to machine g depends on the order in which all jobs assigned to g are processed. Let T be the set of jobs processed on g previous to i. Then the completion time of i is $\frac{p(T \cup \{i\})}{s_g}$. The makespan then also corresponds to the maximum completion time over all jobs.

Scheduling Notation

We adopt the $(\alpha|\beta|\gamma)$ notation introduced by Graham et al. [59] to classify scheduling problems, where α describes the machine environment, β provides job characteristics and scheduling constraints, and γ states the objective function to minimize.

In this thesis, we consider $\alpha \in \{1, \mathrm{P}m, \mathrm{Q}m\}$, where 1 stands for the setting of one single machine, $\mathrm{P}m$ for m identical machines, and $\mathrm{Q}m$ for m related machines. When m is clear from the context, we simply write P or Q. The β field takes the values $\beta \in \{p_i = p, p_i \in N, r_j, \mathrm{pmtn}\}$, where $p_i = p$ means that all jobs have processing time p, $p_i \in N$ restricts processing time to those specified by $N \subseteq \mathbb{N}$, r_i denotes the case in which all jobs have release dates r_i, and pmtn states that preemption is allowed. Note that for $\beta \in \{r_i, \mathrm{pmtn}\}$, schedules have to ensure that no job is executed previous to its release date or account for preemption, respectively. Furthermore, $\gamma \in \{\mathrm{C}_{\max}, \sum \mathrm{C}_i, \sum w_i \mathrm{C}_i\}$, for the objectives to minimize the makespan, the sum of all job completion times, or the weighted sum of all job completion times for a given $\boldsymbol{w} \in \mathbb{N}^n$. Our core results on scheduling within this thesis are for $(\mathrm{Q}||\mathrm{C}_{\max})$ problems and their subproblems $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$, $(\mathrm{P}||\mathrm{C}_{\max})$, and $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$. However, we touch on the other problems as well.

3.2.2 Bin Packing

Given n objects and a vector $\boldsymbol{\varsigma} \in (0,1]^n$ of object sizes ς_i for each object $i \in [n]$, a BinPacking instance for $S \subseteq [n]$ is defined by $I_S := \boldsymbol{\varsigma}_S$.

Assumption 3.2. Objects are ordered such that $\varsigma_1 \geq \ldots \geq \varsigma_n$.

The aim is to find an assignment of the objects in S to bins that uses a minimum amount of bins and does not exceed the unit capacity of any bin. We let $\mathsf{BP}(S)$ be the minimum number of bins needed to assign object set S.

In the context of cost-sharing, we assume that the service provider owns the bins and that each player i owns exactly one object. Thus, a player receives the service if and only if his object is assigned to some bin. We will use $S \subseteq [n]$ to denote players and objects interchangeably. The succinct representation of $(I_S)_{S \subseteq [n]}$ is $\boldsymbol{\varsigma}$. The function $\mathsf{BP} : 2^{[n]} \to \mathbb{R}_{\geq 0}$ defines the optimal service cost.

3.2.3 Facility Location and Fault Tolerant Facility Location

Given n players, a set of facilities F, a vector $\boldsymbol{o} \in \mathbb{N}^{|F|}$ of opening costs o_f for each facility $f \in F$, and a function $d : ([n] \cup F) \times ([n] \cup F) \to \mathbb{N}_0$ defining the distances between all pairs of players and facilities, a FaultTolerantFL instance for an allocation $\boldsymbol{a} \in [|F|]_0^n$ is defined by $I_{\boldsymbol{a}} := (F, \boldsymbol{o}, d, \boldsymbol{a})$.

Assumption 3.3. We assume that d is a *metric*, i.e., for all $i, j, k \in [n] \cup F$, d satisfies $(d(i,j) = 0 \Leftrightarrow i = j)$, $d(i,j) = d(j,i)$, and $d(i,j) \leq d(i,k) + d(k,j)$.

The aim is to open a set of facilities and connect each player $i \in [n]$ to a_i distinct open facilities, such that the total opening and connection cost is minimized.

For $k \in \mathbb{N}$, let $\mathcal{F}_k := \{F' \subseteq F \mid |F'| \geq k\}$ be all sets of facilities with cardinality at least k. For $i \in [n]$ and $F' \in \mathcal{F}_{\max_j\{a_j\}}$, let F'_i denote a set of a_i closest distinct facilities in F' to i. Then the cost of an optimal solution is

$$\mathsf{FTFL}(\boldsymbol{a}) := \min_{F' \in \mathcal{F}_{\max_j\{a_j\}}} \left\{ \sum_{f \in F'} o_f + \sum_{i \in [n]} \sum_{f \in F'_i} d(i,f) \right\} .$$

In the context of general demand cost-sharing, given maximum service levels L_i for all $i \in [n]$ that define the allocation space $\mathcal{A} = [L_1]_0 \times \ldots \times [L_n]_0$, we only consider FaultTolerantFL problems with $|F| \geq \max_i L_i$ and $\boldsymbol{a} \in \mathcal{A}$. A succinct representation of $(I_{\boldsymbol{a}})_{\boldsymbol{a} \in \mathcal{A}}$ is $(\boldsymbol{o}, D)$, where $D \in \mathbb{N}_0^{n+|F|} \times \mathbb{N}_0^{n+|F|}$ is a triangular matrix containing the respective values computed by function d. For clarity, we use the representation $(F, \boldsymbol{o}, d)$. The function $\mathsf{FTFL} : \mathcal{A} \to \mathbb{R}_{\geq 0}$ defines the optimal service cost.

Restricting $\boldsymbol{a}$ to $\boldsymbol{a} \in \{0,1\}^n$, we obtain FacilityLocation optimization and binary demand cost-sharing problems. For a FacilityLocation cost-sharing problem, the optimal cost function is denoted by $\mathsf{FL} : 2^{[n]} \to \mathbb{R}_{\geq 0}$, where $\mathsf{FL}(S) := \mathsf{FTFL}(\boldsymbol{a})$ for all $S \subseteq [n]$ and $\boldsymbol{a} := \mathsf{in}(S)$.

3.2.4 Steiner Forest and Generalized Steiner Forest

Given n players, an undirected graph $G = (V, E)$, vector $\boldsymbol{w} \in \mathbb{N}^{|E|}$ of edge weights w_e for all $e \in E$, and vectors $\boldsymbol{s} \in V^n$ and $\boldsymbol{t} \in V^n$ representing pairs (s_i, t_i) of nodes for each player $i \in [n]$, a GenSteinerForest instance for an allocation $\boldsymbol{a} \in \mathbb{N}_0^n$ is defined by $I_{\boldsymbol{a}} := (G, \boldsymbol{w}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{a})$.

The aim is to determine a subgraph with minimum overall edge weight that has a_i edge-disjoint paths between s_i and t_i for all $i \in [n]$. The cost of an optimal solution for allocation $\boldsymbol{a}$ is denoted by $\mathsf{GSF}(\boldsymbol{a})$.

Assumption 3.4. We assume a simplification allowing to use an arbitrary amount of edge copies, where the cost of each edge copy is equal to the cost of the edge, i.e., we seek for a multiset of edges that contain a_i edge disjoint path between s_i and t_i for all $i \in [n]$.

In the context of general demand cost-sharing, given maximum service levels L_i for all $i \in [n]$ that define the allocation space $\mathcal{A} = [L_1]_0 \times \ldots \times [L_n]_0$, we only consider GenSteinerForest problems with $\boldsymbol{a} \in \mathcal{A}$. A succinct representation of $(I_{\boldsymbol{a}})_{\boldsymbol{a} \in \mathcal{A}}$ is $(\mathcal{G}, \boldsymbol{s}, \boldsymbol{t})$, where $\mathcal{G} \in \mathbb{N}^{|V|,|V|}$ represents a triangular adjacency matrix with weight entries. For clarity, we use the representation $(G, \boldsymbol{w}, \boldsymbol{s}, \boldsymbol{t})$. The function $\mathsf{GSF} : \mathcal{A} \to \mathbb{R}_{\geq 0}$ defines the optimal service cost.

Restricting $\boldsymbol{a}$ to $\boldsymbol{a} \in \{0,1\}^n$, we obtain SteinerForest optimization and binary demand cost-sharing problems. For a SteinerForest cost-sharing problem, the optimal cost function is denoted by $\mathsf{SF} : 2^{[n]} \to \mathbb{R}_{\geq 0}$, where $\mathsf{SF}(S) := \mathsf{GSF}(\boldsymbol{a})$ for all $S \subseteq [n]$ and $\boldsymbol{a} := \mathsf{in}(S)$.

3.3 Related Work

Garey and Johnson showed that already $(\mathrm{P2}||\mathrm{C}_{\max})$ and $(\mathrm{P}||\mathrm{C}_{\max})$ problems are NP-hard [51] and even strongly NP-hard [52], respectively. On the positive side, Hochbaum and Shmoys [68] developed a PTAS for $(\mathrm{P}||\mathrm{C}_{\max})$ and subsequently also a PTAS for $(\mathrm{Q}||\mathrm{C}_{\max})$ [69]. Their algorithms are based on a decision procedure which tests if there exists a schedule for a given problem instance where all jobs are completed by time t. Naturally, this decision problem can be viewed as a BinPacking problem (with variable bin sizes for $(\mathrm{Q}||\mathrm{C}_{\max})$). The minimum t is computed by a simple binary search procedure. Similar approaches have been used to develop other approximation algorithms for $(\mathrm{P}||\mathrm{C}_{\max})$ and $(\mathrm{Q}||\mathrm{C}_{\max})$ based on the Multifit algorithm proposed by Coffman et al. [26] (see, e.g., [20, 43, 45, 46, 138]).

A very simple approach for solving $(\mathrm{Q}||\mathrm{C}_{\max})$ problems is to consider the jobs in an arbitrary order and assign each job to a machine for which the completion time is minimized, taking into account the jobs that have been assigned already. This approximation algorithm is known as List-Scheduling. Graham [58] showed that it yields an approximation ratio of $2 - \frac{1}{m}$ for $(\mathrm{P}||\mathrm{C}_{\max})$. Since the worst case arises when the last job has the largest processing time, Graham [58] suggested the LPT algorithm (**l**ongest **p**rocessing **t**ime first) that considers the jobs by non-increasing processing times. LPT is optimal for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$, achieves an approximation ratio of $\frac{4}{3} - \frac{1}{3m}$ for $(\mathrm{P}||\mathrm{C}_{\max})$ [58], and an approximation ratio of $\frac{5}{3}$ for $(\mathrm{Q}||\mathrm{C}_{\max})$ [44]. Its running time is $O(n \log n + nm)$ in general and $O(n \log n + n \log m)$ for $(\mathrm{P}||\mathrm{C}_{\max})$ when using a priority queue for job assignment. If jobs are identical, we do not have to sort the jobs and can run LPT for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ problems in time $O(n \log m)$, again using a priority queue. For $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ problems, clearly $O(n)$ time is sufficient.

Recently, makespan scheduling problems have been considered in a game-theoretic setting where there is no central authority that assigns the jobs. Instead, each player assigns his job to a machine himself. This scenario is motivated by interpreting machines as links (e.g., different kinds of streets like highways, freeways, or lanes) on which a player routes his traffic. The *latency* of his traffic then corresponds to the completion time of the respective machine. For this scenario, one aims at computing a Nash equilibrium assignment, meaning that no player can improve the latency of his traffic by unilaterally deviating from the current assignment (i.e., choosing another link). Notably, the LPT algorithm has the property that in every iteration, the current assignment is a Nash equilibrium [42].

It is well known that Nash equilibria do not always correspond to an optimal assignment. To better understand the performance of Nash equilibria, Papadimitriou and Koutsoupias [85] proposed to consider the ratio between the optimum and the worst Nash equilibrium with special regard to $(\mathrm{P}||\mathrm{C}_{\max})$ problems. This gave rise to many other works on this topic, see, e.g., [30, 40, 41, 47–49, 84] and the references therein. We also refer the interested reader to the surveys of Gairing et al. [50] and Czumaj [29].

Makespan scheduling has also been considered for mechanism design problems in which the private data is part of the input of the optimization problem. Nisan and Ronen [104] introduced a model in which players own machines instead of jobs. They considered *unrelated* machines, meaning that each machine g has data $t_{g,1}, \ldots, t_{g,n}$ denoting the times that g needs to process jobs $1, \ldots, n$. This data is the private information of each player. Given the players' bids on the execution times of their machines, a mechanism computes an allocation and payments given as a benefit to the players. The utility of a player is the negative time his machine needs to process its assigned jobs plus the payment he receives. The goal is to design an SP mechanism according to these utilities while computing an allocation that approximates the optimal makespan as good as possible. Note that in this model, approximating budget-balance is not a primary goal; in general, the mechanism does not care how much he pays the players.

Nisan and Ronen [104] gave a polynomial-time computable SP mechanism whose allocation approximates the optimal makespan within a factor of n. In addition, they showed that no SP mechanism (polynomial-time or not) can achieve a better approximation factor than 2. This lower bound was recently improved by Christodoulou et al. [22] to $1+\sqrt{2}$ for 3 or more machines. On the other hand, Nisan and Ronen [104] gave a randomized SP mechanism that yields a $\frac{7}{4}$-approximation with respect to the expected makespan. This approximation was recently improved by Lu and Yu [88] to 1.6737. Archer and Tardos [5] considered the restricted ($\mathrm{Q}||\mathrm{C}_{\max}$) model and designed an exponential time SP mechanism that computes an optimal assignment. Moreover, they gave a randomized SP mechanism yielding an approximation factor of 3, which was later improved to 2 by Archer [3]. In the same framework, Andelmann et al. [2] presented a deterministic SP mechanism computing a 5-approximation.

Nisan and Ronen [104] also considered a model in which payments are given to the players only after the execution of the jobs. In case that the mechanism knows the times that the machines spend on computing each job afterwards, it may punish those players who pretended to be capable of processing jobs more quickly. These mechanisms are termed *mechanisms with verification*. For mechanisms with verification for scheduling problems, we refer to the work of Nisan and Ronen [104], Auletta et al. [6, 7], Ventre [130], and Ferrante et al. [39]. For further reading on scheduling problems, we suggest the books [87, 110, 129] and [17].

For BinPacking, there exists a negative result stating that for any $\varepsilon > 0$, BinPacking is NP-hard to approximate within a factor of $\frac{3}{2} - \varepsilon$ (see, e.g. [129]). Thus, there is no PTAS for BinPacking, assuming that $\mathrm{P} \neq \mathrm{NP}$. On the other hand, there are asymptotic PTASs for BinPacking proposed by Fernandez de la Vega and Lueker [38] and Karmarkar and Karp [79].

We now review two simple approximation algorithms for BinPacking problems relevant to this thesis, the FFD (**f**irst **f**it **d**ecreasing) and the NFD (**n**ext **f**it **d**ecreasing) heuristic. Here, objects are first ordered by non-increasing sizes. According to this order, FFD assigns an object to a new bin, if there is no existing bin that has sufficient free capacity to hold the object. Otherwise, it is assigned to the first such bin (where bins are ordered by the times they were first used). For NFD, an object is assigned to a new bin, if it is the first object in the order or does not fit into the same bin as the previous object. Otherwise, it is assigned to this bin. The running time of NFD and FFD is $O(n \log n)$.

Very recently, Dósa [32] showed that the tight bound of FFD is $\frac{11}{9} \cdot opt + \frac{6}{9}$. Furthermore, FFD is known to produce an optimal packing if all object sizes are powers of two, as shown by Coffman et al. [27]. NFD is a 2-approximation algorithm (e.g. [28]) and, as shown by Murgolo [100], *monotonic* (contrary to FFD). Monotonicity is of vital importance with respect to the applicability of specific results presented in this thesis. It requires that decreasing the size of a single object does not increase the solution value. For more information on BinPacking problems, we refer to the survey of Coffman et al. [28].

For FacilityLocation, Guha and Kuller [64] showed a lower bound of 1.463 on polynomial-time approximability, assuming NP $\notin$ DTIME($n^{O(\log\log n)}$). The first constant factor approximation for FacilityLocation was given by Shmoys et al. [120], whose 3.16-approximation algorithm is based on LP rounding. Similar techniques were used to improve their result, e.g., by Chudak [23] and Guha and Khuller [64], finally leading to 1.582-approximations presented by Sviridenko [123]. Jain and Vazirani [77] gave a combinatorial primal-dual 3-approximation algorithm. A subsequent primal-dual algorithm for FacilityLocation by Mahdian et al. [89] provided an 1.861-approximation. This result was further improved by Mahdian et al. [90] and Jain et al. [75] to an 1.52-approximation. Recently, Byrka [18] presented the currently best known 1.5-approximation algorithm based on LP rounding. For a survey on FacilityLocation problems, we confer to the lecture notes of Vygen [133].

FaultTolerantFL problems were first considered by Jain and Vazirani [78] who gave a $3H_L$-approximation algorithm (where L is the maximum requirement) by iteratively applying their 3-approximation algorithm [77]. Algorithms based on LP rounding improved on their result: Guha et al. [65] obtained factor 2.47 and, currently best, Shmoys et al. [119] provided a factor of 2.076.

SteinerForest problems are NP-hard as particularly the subclass of SteinerTree problems is NP-hard due to Karp [80], and even APX-hard due to Bern and Plassmann [9]. On the positive side, Agrawal et al. [1] provided a first approximation algorithm for SteinerForest that computes $2 - \frac{1}{n}$ approximations in polynomial time. Previously, only exact solutions or approximations for special classes of graphs had been considered (see, e.g., [135, 136]).

Agrawal et al. [1] also presented the first polynomial-time algorithm for GenSteinerForest with a guarantee of $(2 - \frac{1}{n}) \cdot \lceil \log(L+1) \rceil$, where L is the largest requirement. Goemans and Williamson [56] simplified and generalized the algorithm from [1]. Jain [74] significantly improved on the results in [1, 56] by presenting a 2-approximation algorithm for GenSteinerForest. His algorithm also works for the setting where edge copies are not allowed. For both settings, these approximation guarantees are the best known.

We note that Agrawal et al. [1] term SteinerForest and GenSteinerForest problems 'minimum cost R-join' and 'minimum cost R-multijoin' problems, respectively. GenSteinerForest problems are also called 'survivable network problems' by, e.g., Chien [21] or Gomory and Hu [57].

3.4 Solution Properties for Makespan Scheduling

Properties of Optimal Solutions

Fix $\boldsymbol{p} \in \mathbb{N}^n$ and $\boldsymbol{s} \in \mathbb{N}^m$. First observe that obviously, the optimal makespan is non-decreasing, i.e., for all $S, T \subseteq [n]$ with $S \subseteq T$ it holds that $\mathsf{MSP}(S) \leq \mathsf{MSP}(T)$ and for all $i, j \in [n]$ with $i \leq j$ it holds that $\mathsf{msp}(i) \leq \mathsf{msp}(j)$, respectively. Moreover, if for a set $S \subseteq [n]$ there is a $t \in \mathbb{N}$ such that $p_i = t$ for all $i \in S$, then $\mathsf{MSP}(S) = t \cdot \mathsf{msp}(|S|)$.

Further properties are summarized in Lemma 3.1 and Lemma 3.2.

Lemma 3.1. *For given $\boldsymbol{p} \in \mathbb{N}^n$ and $\boldsymbol{s} \in \mathbb{N}^m$, the function $\mathsf{MSP} : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ as defined in Section 3.2.1 satisfies the properties below:*

- *For all $S, T \subseteq [n]$ it holds that :*

$$\mathit{MSP}(S \cup T) \leq \mathit{MSP}(S) + \mathit{MSP}(T) \tag{3.1}$$

- *For a set $S \subseteq [n]$ and each optimal assignment $\phi : S \rightarrow [m]$ for S,*

$$\frac{p(S)}{s(m_\phi(S))} \leq \mathit{MSP}(S) \ . \tag{3.2}$$

Proof. Property (3.1) states that optimal makespan costs are subadditive. Consider $S, T \subseteq [n]$ and two corresponding optimal assignments with cost $\mathsf{MSP}(S)$ and $\mathsf{MSP}(T)$. Define a feasible assignment for $S \cup T$ by simply combining the two assignments and breaking ties for $i \in S \cap T$ arbitrarily. This gives a new feasible assignment with makespan cost not larger than $\mathsf{MSP}(S) + \mathsf{MSP}(T)$.

To see (3.2), observe that by definition, $p(\{i \in S \mid \phi(i) = g\}) \leq s_g \cdot \mathsf{MSP}(S)$ for all $g \in m_\phi(S)$. Summing up over all $g \in m_\phi(S)$ yields the desired result.

□

Lemma 3.2. *For a given $\boldsymbol{s} \in \mathbb{N}^m$, the function $\mathit{msp} : [n] \rightarrow \mathbb{R}_{\geq 0}$ as defined in Section 3.2.1 satisfies the properties below, with $\sigma := s([m])$ denoting the sum of machine speeds:*

$$\begin{aligned}
\textit{For all } i,j \in [n] : \ & \mathit{msp}(i+j) \leq \mathit{msp}(i) + \mathit{msp}(j), && (3.3)\\
& \textit{especially } \frac{\mathit{msp}(2i)}{2i} \leq \frac{\mathit{msp}(i)}{i} && \\
\textit{For all } i \in [n] : \ & \mathit{msp}(i) \geq \frac{i}{\sigma} && (3.4)\\
\textit{For all } i \in [n] : \ & \mathit{msp}(i \cdot \sigma) = i && (3.5)\\
\textit{For all } i \in \{m+1, \ldots, n\} : \ & \mathit{msp}(i) \leq \frac{i}{\sigma} \cdot \frac{2m}{m+1} && (3.6)\\
\textit{For all } i \in [n] : \ & \min_{j \in [i]} \frac{\mathit{msp}(j)}{j} \geq \frac{1}{\sigma} && (3.7)\\
\textit{For all } i \in [n] : \ & \textit{If } k \in [i] \textit{ is maximum with} && (3.8)\\
& \frac{\mathit{msp}(k)}{k} = \min_{j \in [i]} \frac{\mathit{msp}(j)}{j} \ , && \\
& \textit{then } \mathit{msp}(i) \leq 2 \cdot \mathit{msp}(k) \ . &&
\end{aligned}$$

Proof. Properties (3.3) and (3.4) follow from (3.1) and (3.2), respectively.

Property (3.5) is due to the fact that for each optimal assignment for $i \cdot \sigma$ identical jobs, all machines have the same completion time, since exactly $i \cdot s_g$ identical jobs are placed on machine g for all $g \in [m]$. Any other assignment has to strictly increase the completion time of at least one machine and thus the makespan.

To prove (3.6), consider an optimal assignment ϕ for $i \geq m+1$ identical jobs. Without loss of generality assume that the set of jobs is $[i]$. Let T be the set of machines whose completion times are equal to $\mathsf{msp}(i)$. We change ϕ to ϕ' by moving a job k with $\phi(k) \in T$ from machine $\phi(k)$ to machine h if the new completion time of h is strictly smaller than $\mathsf{msp}(i)$. This can be done at most $|T| - 1$ times as ϕ is optimal. Now consider a machine $g \in T$ whose completion time has not changed. It holds that

$$p(\{j \in [i] \mid \phi'(j) = g\}) = \mathsf{msp}(i) \cdot s_g \ ,$$

and for all $h \in [m] \backslash \{g\}$ we have that

$$p(\{j \in [i] \mid \phi'(j) = h\}) + 1 \geq \mathsf{msp}(i) \cdot s_h \ .$$

Summing up over all machines yields $i + m - 1 \geq \mathsf{msp}(i) \cdot \sigma$. Consequently, with $i + (m-1) \cdot \frac{i}{m+1} \geq \mathsf{msp}(i) \cdot \sigma$ we get the bound.

In order to show Property (3.7), let $\min_{j \in [i]} \frac{\mathsf{msp}(j)}{j} = \frac{\mathsf{msp}(k)}{k}$ for a $k \in [i]$. By (3.4) it holds that $\frac{\mathsf{msp}(k)}{k} \geq \frac{1}{\sigma}$.

Finally, for (3.8), observe that $\frac{\mathsf{msp}(2k)}{2k} \leq \frac{\mathsf{msp}(k)}{k}$ by (3.3). If now $i \geq 2k$, obviously k cannot be maximum. Thus, $i < 2k$. Since msp is non-decreasing and by (3.3), $\mathsf{msp}(i) \leq \mathsf{msp}(2k) \leq 2 \cdot \mathsf{msp}(k)$.

□

The LPT Algorithm and Properties of LPT Solutions

Given a $(\mathrm{Q}||\mathrm{C}_{\max})$ problem instance $(\boldsymbol{p}_S, \boldsymbol{s})$ for $S \subseteq [n]$, Algorithm 3.1 formally computes the LPT assignment for S:

Algorithm 3.1 (computing LPT assignments).
Input: set $S \subseteq [n]$, processing times $\boldsymbol{p}_S \in \mathbb{N}^{|S|}$, speeds $\boldsymbol{s} \in \mathbb{N}^m$
Output: assignment $\phi : S \to [m]$

$T := \emptyset$ ▷ set of assigned jobs
while $S \neq \emptyset$ **do**
  $h := \arg\min_{g \in [m]} \dfrac{p_{\min S} + p(\{j \in T \ \mid \phi(j) = g\})}{s_g}$
  $\phi(\min S) := h$
  $T := T \cup \{\min S\}$; $S := S \setminus \{\min S\}$
return ϕ

We now formalize that LPT computes a Nash Equilibrium (in the sense of [42], see page 24). Assume that exactly job set T is already assigned by LPT's assignment ϕ. Then, for each job $i \in T$ with $\phi(i) = h$ it holds that

$$\text{for all } g \in [m] \setminus h : \ \frac{p(\{j \in T \mid \phi(j) = g\}) + p_i}{s_g} \geq \frac{p(\{j \in T \mid \phi(j) = h\})}{s_h} \ .$$

We utilize the Nash equilibrium property in Lemma 3.3.

Lemma 3.3. *Consider a* $(\mathrm{Q}||\mathrm{C}_{\max})$ *problem instance* $(\boldsymbol{p}_S, \boldsymbol{s})$ *for* $S \subseteq [n]$. *Let* $\hat{S} \subseteq S$ *be the jobs that* LPT *assigns until there is exactly one machine that has a completion time equal to* $C_{\mathsf{LPT}}(S)$. *Furthermore, let* $\phi' : S \to [m]$ *be the assignment that* LPT *computes for* S.

- *There is an optimal assignment* $\phi : T \to [m]$ *for every* $T \subseteq \hat{S}$ *such that*

$$m_\phi(T) \subseteq m_{\phi'}(\hat{S}) \ . \tag{3.9}$$

- *If at least two jobs of* $\hat{S}$ *are assigned to the same machine then*

$$C_{\mathsf{LPT}}(S) \leq \frac{2 \cdot |m_{\phi'}(\hat{S})|}{|m_{\phi'}(\hat{S})| + 1} \cdot \frac{p(\hat{S})}{s(m_{\phi'}(\hat{S}))} \ . \tag{3.10}$$

Proof. First we prove Property (3.9). Fix $T \subseteq \hat{S}$. If $C_{\mathsf{LPT}}(\hat{S}) = \mathsf{MSP}(T)$, we define ϕ by adopting the assignment for T given by ϕ'. If $C_{\mathsf{LPT}}(\hat{S}) > \mathsf{MSP}(T)$, consider a machine $g \notin m_{\phi'}(\hat{S})$. It is sufficient to show that $g \notin m_{\phi}(T)$ for any optimal assignment ϕ for T. Among all jobs in $\hat{S}$, let i be the job last assigned by LPT. It holds that $p_i = \min\{p_j \mid j \in \hat{S}\}$. Since i was not assigned to g, it holds that $\frac{p_i}{s_g} \geq C_{\mathsf{LPT}}(S) = C_{\mathsf{LPT}}(\hat{S}) > \mathsf{MSP}(T)$. Consequently, no job with size larger or equal to p_i is assigned to g in an optimal assignment for T. Since p_i is the smallest processing time in $\hat{S}$, no job in T is assigned to g in an optimal assignment. Hence, $g \notin m_{\phi}(T)$.

We continue to prove Inequality (3.10). For the sake of readability, we write $\tau := |m_{\phi'}(\hat{S})|$. Among all jobs in $\hat{S}$, let i be the job last assigned by LPT, and define $g := \phi'(i)$. Especially, $p_i = \min\{p_j \mid j \in \hat{S}\}$. Since the assignment computed by LPT for $\hat{S}$ is a Nash equilibrium,

$$p(\{i \in \hat{S} \mid \phi'(i) = g\}) = C_{\mathsf{LPT}}(S) \cdot s_g$$

and for all $h \in m_{\phi'}(\hat{S}) \setminus g$,

$$p(\{i \in \hat{S} \mid \phi'(i) = h\}) + p_i \geq C_{\mathsf{LPT}}(S) \cdot s_h \ .$$

Summation yields

$$p(\hat{S}) + (\tau - 1) \cdot p_i \geq C_{\mathsf{LPT}}(S) \cdot s(m_{\phi'}(\hat{S})) \ .$$

Since p_i is the smallest processing time of a job in $\hat{S}$, and the set $\hat{S}$ consists of at least $\tau + 1$ jobs, it holds that $p(\hat{S}) \geq (\tau + 1) \cdot p_i$. This leads to

$$\left(1 + \frac{\tau - 1}{\tau + 1}\right) \cdot p(\hat{S}) \geq C_{\mathsf{LPT}}(S) \cdot s(m(\hat{S})) \ ,$$

yielding Inequality (3.10). □

Finally, we prove a property of C_{LPT} in Lemma 3.4 that reminds of subadditivity but is significantly weaker. It requires that all players in the one set have the same processing time which is not larger than any processing time in the other set:

Lemma 3.4. *For $\boldsymbol{p} \in \mathbb{N}^n$ and $\boldsymbol{s} \in \mathbb{N}^m$, consider two $(\mathrm{Q}||\mathrm{C}_{\max})$ instances $(\boldsymbol{p}_S, \boldsymbol{s})$ and $(\boldsymbol{p}_T, \boldsymbol{s})$ for $S, T \subseteq [n]$ with $S \cap T = \emptyset$ and $p_i = t$ for all $i \in T$, where $t \in \mathbb{N}$ such that $t \leq \min_{i \in S}\{p_i\}$. Then $C_{\mathsf{LPT}}(S) + C_{\mathsf{LPT}}(T) \geq C_{\mathsf{LPT}}(S \cup T)$.*

Proof. Let ϕ_S, ϕ_T and $\phi_{S \cup T}$ be the assignments that LPT computes for S,T and $S \cup T$, respectively. Without loss of generality we may assume that LPT for $S \cup T$ assigns elements in S with size t prior to elements in T, implicating $\phi_S(i) = \phi_{S \cup T}(i)$ for all $i \in S$. We additionally assume without loss of generality that the runs for LPT on input T and $S \cup T$ process the jobs in T in the same order. Let $T := \{i_1, \ldots, i_{|T|}\}$ according to this order and define $T_j := \{i_1, \ldots, i_j\}$ for all $j \in [|T|]$.

Assume now that $C_{\mathsf{LPT}}(S) + C_{\mathsf{LPT}}(T) < C_{\mathsf{LPT}}(S \cup T)$ and consider the first element i_j in T such that $C_{\mathsf{LPT}}(S) + C_{\mathsf{LPT}}(T_j) < C_{\mathsf{LPT}}(S \cup T_j)$. In particular, $C_{\mathsf{LPT}}(S \cup T_{j-1}) < C_{\mathsf{LPT}}(S \cup T_j)$. We show that

$$\text{for all } g \in [m] : |\{i \in T_{j-1} \mid \phi_{S \cup T}(i) = g\}| \geq |\{i \in T_{j-1} \mid \phi_T(i) = g\}| \quad (3.11)$$

$$\text{there exists } g' \in [m] : |\{i \in T_{j-1} \mid \phi_{S \cup T}(i) = g'\}| > |\{i \in T_{j-1} \mid \phi_T(i) = g'\}| \quad (3.12)$$

Assume that (3.11) does not hold for a machine $h \in [m]$. We get a contradiction by

$$\begin{aligned} C_{\mathsf{LPT}}(S) + C_{\mathsf{LPT}}(T_j) &\geq \frac{p(\{i \in S \mid \phi_{S \cup T}(i) = h\}) + p(\{i \in T_j \mid \phi_T(i) = h\})}{s_h} \\ &\geq \frac{p(\{i \in S \cup T_{j-1} \mid \phi_{S \cup T}(i) = h\}) + t}{s_h} \\ &\geq \frac{p(\{i \in S \cup T_j \mid \phi_{S \cup T}(i) = \phi_{S \cup T}(i_j)\})}{s_{\phi_{S \cup T}(i_j)}} \\ &= C_{\mathsf{LPT}}(S \cup T_j) \ . \end{aligned}$$

If (3.11) holds with equality for all $g \in [m]$, we get

$$\begin{aligned} C_{\mathsf{LPT}}(S \cup T_j) &\leq \frac{p(\{i \in S \cup T_{j-1} \mid \phi_{S \cup T}(i) = \phi_T(i_j)\}) + t}{s_{\phi_T(i_j)}} \\ &= \frac{p(\{i \in S \mid \phi_S(i) = \phi_T(i_j)\}) + p(\{i \in T_j \mid \phi_T(i) = \phi_T(i_j)\})}{s_{\phi_T(i_j)}} \\ &\leq C_{\mathsf{LPT}}(S) + C_{\mathsf{LPT}}(T_j) \end{aligned}$$

Now (3.11) and (3.12) clearly yield a contradiction.

□

4

Moulin Mechanisms and Cost-Sharing Methods in the Approximate Core

4.1 Contribution

- For every $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problem with d different processing times, we give Moulin mechanisms that are $2d$-BB and $2d \cdot (1 + H_n)$-EFF for C_{LPT} and computable in polynomial time. Up to a factor of 2, this is in general the best budget-balance approximation possible for cross-monotonic cost shares [10]. We further derive that no Moulin mechanism for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems can generally be better than $\max\{d, H_n\}$-EFF.
- Restricting attention to $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems, we show that for every such problem, there are Moulin mechanisms that are even $\frac{2m}{m+1}$-BB and $\frac{2m}{m+1} \cdot (1 + H_n)$-EFF for C_{LPT} and computable in polynomial time. By identifying a $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problem for which there is no cost-sharing method in the β-core for $\beta < \frac{2m}{m+1}$, we show that $\frac{2m}{m+1}$-BB is generally the best that can be achieved by Moulin mechanisms. Since the efficiency bound of H_n is also obtained for a $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problem, the efficiency guarantee of our Moulin mechanisms for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems is asymptotically tight.
- The upper and lower bounds on the social cost efficiency of our presented Moulin mechanisms are obtained by investigating the summability of our cost-sharing methods and applying a result by Roughgarden and Sundararajan [114]. We conduct a fine-grained analysis of the approximate social cost efficiency of our Moulin mechanisms for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems that shows that the upper bounds are overly pessimistic. It turns out that for many cases, we can guarantee $\frac{2m}{m+1}$-EFF instead of $O(\log n)$-EFF.
- Despite the devastating bound of d for the approximate budget-balance for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems, we introduce cost sharing methods for every such problem that are in the $\frac{2m}{m+1}$-core for C_{LPT}. Though they do not lead to GSP mechanisms, they nevertheless distribute the cost in a stable way. The cost-shares for a fixed set can be computed in polynomial time.

The results we introduce in this chapter are published in [10] and [14]. Although $2d$-BB and $\frac{2m}{m+1}$-BB Moulin mechanisms for $(\mathrm{Q}||\mathrm{C}_{\max})$ and $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ were already introduced in [10], we only present the mechanisms from [14] as the underlying cross-monotonic cost-sharing methods are much simpler and thus allow for the first summability and efficiency results.

4.2 Organization

Section 4.3 introduces Moulin mechanisms. Subsequently, Section 4.4 details the interconnection between cost sharing methods in the β-core and β-BB cross-monotonic cost-sharing methods. The bounds proposed by Roughgarden and Sundararajan [114] on the approximate social-cost-efficiency of Moulin mechanisms are provided in Section 4.5. Section 4.6 presents our results on applications to $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems and its subproblems. Section 4.7 concludes.

4.3 Moulin Mechanisms

Given a bid vector $\boldsymbol{b} \in \mathbb{R}^n$, the Moulin mechanism $Moulin_\xi = (Q, x)$ for cost-sharing method ξ can be computed by a very simple algorithm: Initially, consider serving all players. Then repeatedly eliminate players whose bids are below their current cost shares (as implied by ξ) until all remaining players can afford their cost shares. Formally, $Moulin_\xi$ is computed by Algorithm 4.1.

Algorithm 4.1 (computing $Moulin_\xi(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

1: $Q := [n]$
2: **while** there exists $i \in Q$ with $b_i < \xi_i(Q)$ **do**
3: $\quad Q := Q \setminus \{j\}$ for an arbitrary $j \in Q$ with $b_j < \xi_j(Q)$
4: **return** $(Q, \xi(Q))$

It is easy to verify that for every β-BB cost-sharing method ξ, $Moulin_\xi$ is β-BB and meets NPT, VP, and strict CS. Since Theorem 4.1 is a seminal result that has lead to almost all known GSP mechanisms so far, we provide its rather straightforward proof (see [76, 99] for similar proofs).

Theorem 4.1 ([97]). *For any cross-monotonic cost-sharing method ξ, $Moulin_\xi$ is GSP.*

Proof. Assume that $Moulin_\xi = (Q, x)$ is not GSP. Then, there exists a coalition $K \subseteq [n]$ with true valuations $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$ and a bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for all $i \in K$, with at least one strict inequality. Without loss of generality, let $\boldsymbol{v} := (\boldsymbol{v}_K, \boldsymbol{b}_{-K})$.

- We first show that $Q(\boldsymbol{b}) \subseteq Q(\boldsymbol{v})$:
  We show that every player who receives the service for input $\boldsymbol{b}$ also receives the service for input $\boldsymbol{v}$. Assume the opposite and consider the first time that $Moulin_\xi$ with input $\boldsymbol{v}$ rejects a player $j \in Q(\boldsymbol{b})$ by deleting j from set Q in line 3 of Algorithm 4.1, in particular $v_j < \xi_j(Q)$. For this set Q, it is $Q \supseteq Q(\boldsymbol{b})$ and by cross-monotonicity, $\xi_j(Q) \leq \xi_j(Q(\boldsymbol{b}))$. From j in $Q(\boldsymbol{b})$, it follows that $\xi_j(Q(\boldsymbol{b})) \leq b_j$. However, $v_j < \xi_j(Q) \leq \xi_j(Q(\mathbf{b})) \leq b_j$ implies that $j \in K$ which contradicts $u_j(\boldsymbol{v}, v_j) = 0 > v_j - \xi_j(Q(\boldsymbol{b})) = u_j(\boldsymbol{b}, v_j)$. Therefore, $Q(\boldsymbol{b}) \subseteq Q(\boldsymbol{v})$.
- We now show that there is no $j \in K$ with $u_j(\boldsymbol{b}, v_j) > u_j(\boldsymbol{v}, v_j)$:
  For each player $j \in K$ with $u_j(\boldsymbol{b}, v_j) > u_j(\boldsymbol{v}, v_j)$ it has to hold that $j \in Q(\boldsymbol{b})$ and thus $j \in Q(\boldsymbol{v})$ as well. Exploiting cross-monotonicity again, we get the contradiction that $u_j(\boldsymbol{b}, v_j) = v_j - \xi_j(Q(\boldsymbol{b})) \leq v_j - \xi_j(Q(\boldsymbol{v})) = u_j(\boldsymbol{v}, v_j)$.

□

In order to determine the runtime of $Moulin_\xi$ for a specific ξ, we introduce Lemma 4.2:

Lemma 4.2. *For each cost-sharing method ξ such that for every $S \subseteq [n]$, $\xi(S)$ is computable in time $O(t)$, $Moulin_\xi$ is computable in time $O(n \cdot t)$.*

Proof. In each iteration with current set Q, $Moulin_\xi$ has to compute $\xi_i(Q)$ for all $i \in Q$ in the worst case. Furthermore, there are at most n iterations.

□

4.4 Lower Bounds on Budget-Balance

It is easy to see that each cross-monotonic cost-sharing method ξ that is β-BB for C and C' is in the β-core for C and C':

Lemma 4.3. *Let ξ be a cross-monotonic cost-sharing method that is β-BB for C and C'. Then ξ is in the β-core for C and C'.*

Proof. The fact that for all $S \subseteq [n]$ it holds that $\sum_{i\in S} \xi_i(S) \geq C'(S)$ trivially follows from the β-BB of ξ. Additionally utilizing cross-monotonicity of ξ, we see that for all $T, S \subseteq [n]$ with $T \subseteq S$ it holds that

$$\sum_{i\in T} \xi_i(S) \leq \sum_{i\in T} \xi_i(T) \leq \beta \cdot C(T) \ .$$

□

From Lemma 4.3 we can conclude that if there is no cost-sharing method in the β-core for C and C', no β-BB cross-monotonic cost-sharing method can exist. On the other hand, there *can* be cost-sharing methods that are in the β-core despite not being cross-monotonic. As an example, our results give cost-sharing methods for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems in the $\frac{2m}{m+1}$-core for C_{LPT}, while no cross-monotonic cost-sharing method for $(\mathrm{Q}||\mathrm{C}_{\max})$ can in general be better than d-BB, where d is the number of different processing times [10].

4.5 Bounds on Social Cost Efficiency

Roughgarden and Sundararajan [114] provide Theorem 4.4 in order to approximate social cost efficiency of Moulin mechanisms:

Theorem 4.4 ([114]). *Let C' and C be cost functions, and ξ be a cross-monotonic cost-sharing method. Let $\alpha \geq 0$ be the smallest number such that ξ is α-SUM for C, and let $\beta \geq 1$ be the smallest number such that ξ is β-BB for C' and C. Then $Moulin_\xi$ is $(\alpha + \beta)$-EFF and no better than $\max\{\alpha, \beta\}$-EFF for C' and C.*

We remark that in the original work of Roughgarden and Sundararajan [114], β-BB is defined as $\frac{1}{\beta} \cdot C'(S) \leq \sum_{i\in S} \xi_i(S) \leq C(S)$. However, adapting the proof to our definition of budget-balance (confer Definition 2.19) yields the same bounds. We give the modified proof below for reasons of clearness. In general, when adapting a Moulin mechanism $Moulin_\xi$ that is β-BB in the sense of [114] to our definition of β-BB, we have to multiply the cost-shares given by ξ by β. However, this means that if ξ was α-SUM before, it is now $(\beta \cdot \alpha)$-SUM and the guaranteed efficiency approximation by Theorem 4.4 has impaired.

The advantage to define β-BB in the sense of [114] is that a natural trade-off between budget-balance and efficiency can be observed: dividing the cost shares of a β-BB (in the sense of [114]) and α-SUM cost-sharing method by $\gamma > 1$ yields a $(\gamma \cdot \beta)$-BB (in the sense of [114]) and $(\frac{\alpha}{\gamma})$-SUM cost-sharing method. Thus, sacrificing budget-balance to gain social cost efficiency is possible within the bounds of Theorem 4.4. However, sacrificing budget-balance according to our definition of β-BB means to multiply the cost shares by a factor $\gamma > 1$, and automatically leads to sacrificing summability and efficiency, accordingly.

We nevertheless think that attaching more importance to cost-recovery is far more reasonable, as the service provider incurs a loss otherwise.

Proof (of Theorem 4.4). Fix cost functions C and C', a cross-monotonic cost-sharing method ξ, and α, β that meet the requirements in Theorem 4.4.

We first look at the lower bounds. The proof from [114] showing that *Moulin*$_\xi$ is no better than α-EFF does not involve any budget-balance considerations and still applies. Now let $S \subseteq [n]$ with $\sum_{i \in S} \xi_i(S) = \beta \cdot C(S)$ and define $\boldsymbol{v}$ by $v_i := \xi_i(S) - \varepsilon$ for all $i \in S$ and $v_i := -\varepsilon$ otherwise. Then, by cross-monotonicity of ξ, *Moulin*$_\xi$ with input $\boldsymbol{v}$ serves no player and induces a social cost of $\beta \cdot C(S) - n \cdot \varepsilon$. On the other hand, the optimal social cost is not larger than $C(S)$. Therefore, *Moulin*$_\xi$ is no better than β-EFF.

We continue to show the upper bound. Fix $\boldsymbol{v} \in \mathbb{R}^n$ and let Q be the set that is output by *Moulin*$_\xi$ with input $\boldsymbol{v}$. Let S^* be the set that minimizes social cost. By definition of social cost,

$$SC_{C'}(Q, \boldsymbol{v}) - SC_{C'}(Q \cap S^*, \boldsymbol{v}) = C'(Q) - C'(Q \cap S^*) - \sum_{i \in Q \setminus S^*} v_i \ . \tag{4.1}$$

Furthermore, by β-BB and cross-monotonicity,

$$\begin{aligned} \sum_{i \in Q \setminus S^*} \xi_i(Q) &= \sum_{i \in Q} \xi_i(Q) - \sum_{i \in Q \cap S^*} \xi_i(Q) \\ &\geq C'(Q) - \sum_{i \in Q \cap S^*} \xi_i(Q \cap S^*) \\ &\geq C'(Q) - \beta \cdot C(Q \cap S^*) \ . \end{aligned} \tag{4.2}$$

Using Inequality (4.2) within Equation (4.1) and utilizing that $\xi_i(Q) \leq v_i$ for all $i \in Q$, we get

$$\begin{aligned} SC_{C'}(Q, \boldsymbol{v}) &\leq SC_{C'}(Q \cap S^*, \boldsymbol{v}) + \beta \cdot C(Q \cap S^*) - C'(Q \cap S^*) \\ &= \sum_{i \notin Q \cap S^*} v_i + \beta \cdot C(Q \cap S^*) \\ &= \sum_{i \notin S^*} v_i + \sum_{i \in S^* \setminus Q} v_i + \beta \cdot C(Q \cap S^*) \end{aligned}$$

Let $(S^* \setminus Q) = \{i_1, \ldots, i_{|S^* \setminus Q|}\}$ ordered reverse to the order in which *Moulin*$_\xi$ deletes these players, and for each $j \in [|S^* \setminus Q|]$, define $(S^* \setminus Q)_j := \{i_1, \ldots, i_j\}$ to be the first j players in $S^* \setminus Q$. Furthermore, let Q_j be the set from which i_j is deleted by

Moulin$_\xi$. In particular, $Q_j \supseteq (S^* \setminus Q)_j$. By cross-monotonicity and the fact that ξ is α-SUM, we conclude

$$\begin{aligned} SC_{C'}(Q, \boldsymbol{v}) &< \sum_{i \notin S^*} v_i + \sum_{j=1}^{|S^* \setminus Q|} \xi_{i_j}(Q_j) + \beta \cdot C(Q \cap S^*) \\ &\leq \sum_{i \notin S^*} v_i + \sum_{j=1}^{|S^* \setminus Q|} \xi_{i_j}((S^* \setminus Q)_j) + \beta \cdot C(Q \cap S^*) \\ &\leq \sum_{i \notin S^*} v_i + \alpha \cdot C(S^* \setminus Q) + \beta \cdot C(S^*) \\ &\leq (\alpha + \beta) \cdot SC_C(S^*, \boldsymbol{v}) \ . \end{aligned}$$

□

We adopt an idea of Roughgarden and Sundararajan [114] to show a lower bound of H_n on the approximate social cost efficiency for constant costs. The bound from [114] holds under 1-BB and improves for relaxations (its value is $\frac{H_n}{\beta}$); in our model, H_n is a lower bound under any budget-balance approximation. Lemma 4.5 is a direct corollary of Lemma 4.6 and Theorem 4.4.

Lemma 4.5. *For a cost function C with $C(S) = a$ for an $a \in \mathbb{R}_{>0}$ and all $S \subsetneq [n]$, there is no Moulin mechanism Moulin$_\xi$ with $\sum_{i \in S} \xi_i(S) \geq a$ for all $S \subseteq [n]$ that is γ-EFF for $\gamma < H_n$.*

Lemma 4.6. *For a cost function C with $C(S) = a$ for an $a \in \mathbb{R}_{>0}$ and all $S \subsetneq [n]$, there is no cost-sharing method ξ with $\sum_{i \in S} \xi_i(S) \geq a$ for all $S \subseteq [n]$ that is α-SUM for $\alpha < H_n$.*

Proof. Consider an arbitrary cost-sharing method ξ with $\sum_{i \in S} \xi_i(S) \geq a$ for all $S \subsetneq [n]$. We define an order $i_1, \ldots, i_n$ of $[n]$ with $S_j := \{i_1, \ldots, i_j\}$ for all $j \in [n]$ and show that $\sum_{j=1}^{n} \xi_{i_j}(S_j) \geq H_n \cdot C([n])$.

Note that due to $\sum_{i \in [n]} \xi_i([n]) \geq a$ there is a player $k \in [n]$ with $\xi_k([n]) \geq \frac{a}{n}$. We let $i_n := k$. Similarly, from $\sum_{i \in [n] \setminus \{k\}} \xi_i([n] \setminus \{k\}) \geq a$, we know that there is a player $k' \in [n] \setminus \{k\}$ with $\xi_{k'}([n] \setminus \{k\}) \geq \frac{a}{n-1}$, and we let $i_{n-1} := k'$. Iteratively, we define $i_{n-2}, \ldots, i_1$ by the same procedure. Then, $\sum_{j=1}^{n} \xi_{i_j}(S_j) \geq \sum_{j=1}^{n} \frac{a}{j} = H_n \cdot C([n])$.

□

4.6 Applications To Scheduling

4.6.1 Lower Bounds on Budget-Balance and Efficiency

In order to obtain lower bounds on β-BB for cross-monotonic cost-sharing methods for scheduling problems, we use the interrelation to the β-core and show that already for identical jobs and machines, β cannot be smaller than $\frac{2m}{m+1}$ in general. The situation gets worse when allowing for different processing times and speeds [10].

Theorem 4.7. *There is a* $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ *cost-sharing problem for which there is no cross-monotonic cost-sharing method that is β-BB for $\beta < \frac{2m}{m+1}$.*

Theorem 4.7 is a simple corollary of Lemma 4.3 and Theorem 4.8:

Theorem 4.8. *There is a* $(\mathrm{P}|p_i{=}1|\mathrm{C_{max}})$ *cost-sharing problem for which there is no cost-sharing method in the* β*-core for* $\beta < \frac{2m}{m+1}$.

Proof. Consider a $(\mathrm{P}|p_i{=}1|\mathrm{C_{max}})$ cost-sharing problem with $m = n-1$ machines and a cost-sharing method ξ in the β-core. For every $S \subseteq [n]$ with $|S| = m$ it holds that $\sum_{i\in S} \xi_i([n]) \le \beta \cdot \mathsf{MSP}(S) = \beta$. Thus, there exists a player $j \in [n]$ with $\xi_j([n]) \le \frac{\beta}{m}$. It follows that

$$\begin{aligned}\sum_{i\in[n]} \xi_i([n]) &\le \frac{\beta}{m} + \sum_{i\in[n]\setminus\{j\}} \xi_i([n]) \\ &\le \frac{\beta}{m} + \beta \\ &= \frac{m+1}{m} \cdot \beta \ .\end{aligned}$$

From $2 = \mathsf{MSP}([n]) \le \sum_{i\in[n]} \xi_i([n]) \le \frac{m+1}{m} \cdot \beta$, we conclude that $\beta \ge \frac{2m}{m+1}$.

□

Theorem 4.9 ([10]). *There is a* $(\mathrm{Q}||\mathrm{C_{max}})$ *cost-sharing problem with* d *different processing times for which there is no cross-monotonic cost-sharing method that is* β*-BB for* $\beta < d$.

For approximating social cost efficiency we can conclude that in general, we cannot be better than H_n-EFF even in the case of identical jobs and identical machines. For the general problem, the lower bound d for approximate budget-balance even implies that in general, no Moulin mechanism can be better than $\max\{d, H_n\}$-EFF.

Lemma 4.10. *There is a* $(\mathrm{P}|p_i{=}1|\mathrm{C_{max}})$ *cost-sharing problem for which there is no Moulin mechanism that is* γ*-EFF for* $\gamma < H_n$.

Proof. Consider a $(\mathrm{P}|p_i{=}1|\mathrm{C_{max}})$ cost-sharing problem with $m = n$ machines. With $\mathsf{MSP}(S) = 1$ for all $S \subseteq [n]$ and Lemma 4.5, the claim follows.

□

Lemma 4.11. *There is a* $(\mathrm{Q}||\mathrm{C_{max}})$ *cost-sharing problem with* d *different processing times for which there is no Moulin mechanisms that is* γ*-EFF for* $\gamma < d$.

Proof. The lemma simply follows from Theorem 4.4 and Theorem 4.9.

□

4.6.2 Moulin Mechanisms for Identical Jobs

Theorem 4.12 is the main theorem of this section and follows directly from Theorem 4.13, Theorem 4.1, Theorem 4.4, and Lemma 4.2.

Theorem 4.12. *For each* $(\mathrm{Q}|p_i{=}1|\mathrm{C_{max}})$ *cost-sharing problem there is a cost-sharing method* ξ *such that* Moulin_ξ *is GSP,* $\frac{2m}{m+1}$*-BB and* $\frac{2m}{m+1} \cdot (1 + H_n)$*-EFF for* C_{LPT}. *Furthermore,* Moulin_ξ *is computable in time* $O(n^2 \cdot \log m)$.

Theorem 4.13. *For each* $(\mathrm{Q}|p_i{=}1|\mathrm{C_{max}})$ *cost-sharing problem there is a cost-sharing method* ξ *that is cross-monotonic,* $\frac{2m}{m+1}$*-BB for* C_{LPT} *and* $\frac{2m}{m+1} \cdot H_n$*-SUM. For each* $S \subseteq [n]$*, the vector of cost shares* $\xi(S)$ *is computable in time* $O(n \cdot \log m)$.

Proof. Fix $\boldsymbol{s} \in \mathbb{N}^m$ and consider the cost-sharing problem $(\mathbf{1}, \boldsymbol{s})$.

For each $S \subseteq [n]$, define method $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ by

$$\xi_i(S) := \begin{cases} \frac{2m}{m+1} \cdot \min_{j \in [|S|]} \frac{\mathsf{msp}(j)}{j} & \text{if } i \in S \\ 0 & \text{otherwise.} \end{cases}$$

Cross-monotonicity is obvious. Fix $S \subseteq [n]$. We start proving $\frac{2m}{m+1}$-BB. Directly from the definition, it follows that $\sum_{i \in S} \xi_i(S) \leq \frac{2m}{m+1} \cdot \mathsf{msp}(|S|) = \frac{2m}{m+1} \cdot \mathsf{MSP}(S)$.

- If $|S| \leq m$, let $k \leq |S|$ be maximum with $\frac{\mathsf{msp}(k)}{k} = min_{j \in [|S|]} \frac{\mathsf{msp}(j)}{j}$. If $k = |S|$ then $\sum_{i \in S} \xi_i(S) = \frac{2m}{m+1} \cdot \mathsf{MSP}(|S|)$. If $k < |S|$ we get by (3.8) and $k+1 \leq |S| \leq m$ that

$$\begin{aligned}
\sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{|S|}{k} \cdot \mathsf{msp}(k) \\
&\geq \frac{2m}{m+1} \cdot \frac{k+1}{2k} \cdot \mathsf{msp}(|S|) \\
&\geq \frac{2m}{m+1} \cdot \frac{m}{2m-2} \cdot \mathsf{msp}(|S|) \\
&\geq \mathsf{msp}(|S|) \\
&= \mathsf{MSP}(S) \\
&= C_{\mathsf{LPT}}(S) \ .
\end{aligned}$$

- If $|S| \geq m+1$, we know by (3.6) that $\mathsf{msp}(|S|) \leq \frac{|S|}{s([m])} \cdot \frac{2m}{m+1}$. With (3.7), we get

$$\begin{aligned}
\sum_{i \in S} \xi_i(S) &\geq \frac{2m}{m+1} \cdot \frac{|S|}{s([m])} \\
&\geq \mathsf{msp}(|S|) \\
&= \mathsf{MSP}(S) \\
&= C_{\mathsf{LPT}}(S) \ .
\end{aligned}$$

For summability, let $i_1, \ldots, i_{|S|}$ be an arbitrary order of S, and let $S_j := \{i_1, \ldots, i_j\}$ denote the set of the first j elements. Then,

$$\begin{aligned}
\sum_{j=1}^{|S|} \xi_{i_j}(S_j) &\leq \frac{2m}{m+1} \cdot \sum_{i=1}^{|S|} \frac{\mathsf{msp}(i)}{i} \\
&\leq \frac{2m}{m+1} \cdot H_n \cdot \mathsf{msp}(|S|) \\
&= \frac{2m}{m+1} \cdot H_n \cdot \mathsf{MSP}(S) \ .
\end{aligned}$$

Since LPT on $|S|$ identical jobs simultaneously computes all values for $\mathsf{msp}(i)$ for all $i \in [|S|]$, the time to compute $\min_{j \in [|S|]} \frac{\mathsf{msp}(j)}{j} = \xi_i(S)$ for all $i \in S$ is $O(n \cdot \log m)$. $\square$

4.6.3 Moulin Mechanisms for the General Setting

We generalize the idea for identical jobs from Section 4.6.2 to the general scheduling setting to obtain Theorem 4.14 which follows from Theorem 4.15, Theorem 4.1, Theorem 4.4, and Lemma 4.2.

Theorem 4.14. *For each* $(\mathrm{Q}||\mathrm{C}_{\max})$ *cost-sharing problem with* d *different processing times there is a cost-sharing method* ξ *such that Moulin*$_\xi$ *is GSP,* $2d$*-BB, and* $2d \cdot (1 + H_n)$*-EFF for* C_{LPT}*. Furthermore, Moulin*$_\xi$ *is computable in time* $O(n^2 \cdot \log m)$.

Theorem 4.15. *For each* $(\mathrm{Q}||\mathrm{C}_{\max})$ *cost-sharing problem with* d *different processing times there is a cost-sharing method* ξ *that is cross-monotonic,* $2d$*-BB, and* $(2d \cdot H_n)$*-SUM. For each* $S \subseteq [n]$*, the vector of cost shares* $\xi(S)$ *is computable in time* $O(n \cdot \log m)$.

Proof. Fix $\boldsymbol{p} \in \mathbb{N}^n$, $\boldsymbol{s} \in \mathbb{N}^m$ and consider the cost-sharing problem $(\boldsymbol{p}, \boldsymbol{s})$.

For $S \subseteq [n]$, we use the subsequent notations within this proof:

- $\mathcal{P}(S) := \{p_i \mid i \in S\}$ is the set of different processing times of the jobs in S. We let $\mathcal{P}(S) := \{t_1, \dots, t_{|\mathcal{P}(S)|}\}$.
- $S(t) := \{i \in S \mid p_i = t\}$ is the set of players in S with processing time t.

For each $S \subseteq [n]$ define the method $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ by

$$\xi_i(S) := \begin{cases} 2 \cdot p_i \cdot \min_{j \in [|S(p_i)|]} \frac{\mathsf{msp}(j)}{j} & \text{if } i \in S \\ 0 & \text{otherwise.} \end{cases}$$

Cross-monotonicity is obvious. Fix $S \subseteq [n]$. To show $2d$-BB, we first bound the sum of the cost shares from above by $2 \cdot |\mathcal{P}(S)| \cdot \mathsf{MSP}(S)$. We have that

$$\begin{aligned} \sum_{i \in S} \xi_i(S) &= 2 \cdot \sum_{t \in \mathcal{P}(S)} t \cdot |S(t)| \cdot \min_{j \in [|S(t)|]} \frac{\mathsf{msp}(j)}{j} \\ &\leq 2 \sum_{t \in \mathcal{P}(S)} t \cdot \mathsf{msp}(|S(t)|) \\ &= 2 \sum_{t \in \mathcal{P}(S)} \mathsf{MSP}(S(t)) \\ &\leq 2 \cdot |\mathcal{P}(S)| \cdot \mathsf{MSP}(S) \ . \end{aligned}$$

For the lower bound, let $k(t) \in [|S(t)|]$ such that $\frac{\mathsf{msp}(k(t))}{k(t)} = \min_{j \in [S(t)]} \frac{\mathsf{msp}(j)}{j}$ for all $t \in \mathcal{P}(S)$. Applying $2 \cdot \mathsf{msp}(k(t)) \geq \mathsf{msp}(|S(t)|)$ by (3.8) we get

$$\begin{aligned} \sum_{i \in S} \xi_i(S) &= 2 \sum_{t \in \mathcal{P}(S)} t \cdot |S(t)| \cdot \min_{j \in [S(t)]} \frac{\mathsf{msp}(j)}{j} \\ &= 2 \sum_{t \in \mathcal{P}(S)} t \cdot |S(t)| \cdot \frac{\mathsf{msp}(k(t))}{k(t)} \\ &\geq 2 \sum_{t \in \mathcal{P}(S)} t \cdot \mathsf{msp}(k(t)) \\ &\geq \sum_{t \in \mathcal{P}(S)} t \cdot \mathsf{msp}(|S(t)|) \ , \end{aligned}$$

and thus $\sum_{i \in S} \xi_i(S) \geq \sum_{t \in \mathcal{P}(S)} \mathsf{MSP}(S(t)) = \sum_{t \in \mathcal{P}(S)} C_{\mathsf{LPT}}(S(t)) \geq C_{\mathsf{LPT}}(S)$. The last inequality is due to Lemma 3.4.

For summability, let $i_1, \ldots, i_{|S|}$ be an arbitrary order of S, and let $S_j := \{i_1, \ldots, i_j\}$ denote the first j elements. Then,

$$\begin{aligned}
\sum_{j=1}^{|S|} \xi_{i_j}(S_j) &= 2 \cdot \sum_{j=1}^{|S|} \left(p_{i_j} \cdot \min_{j \in [|S_j(p_{i_j})|]} \frac{\mathsf{msp}(j)}{j} \right) \\
&= 2 \cdot \sum_{k=1}^{|\mathcal{P}(S)|} t_k \sum_{l=1}^{|S(t_k)|} \min_{j \in [l]} \frac{\mathsf{msp}(j)}{j} \\
&\leq 2 \cdot \sum_{k=1}^{|\mathcal{P}(S)|} \sum_{l=1}^{|S(t_k)|} \frac{t_k \cdot \mathsf{msp}(l)}{l} \\
&\leq 2 \cdot \sum_{k=1}^{|\mathcal{P}(S)|} \sum_{l=1}^{|S(t_k)|} \frac{t_k \cdot \mathsf{msp}(|S(t_k)|)}{l} \\
&= 2 \cdot \sum_{k=1}^{|\mathcal{P}(S)|} \sum_{l=1}^{|S(t_k)|} \frac{\mathsf{MSP}(S(t_k))}{l} \\
&\leq 2 \cdot \sum_{k=1}^{|\mathcal{P}(S)|} \sum_{l=1}^{|S|} \frac{\mathsf{MSP}(S)}{l} \\
&\leq 2 \cdot |\mathcal{P}(S)| \cdot H_{|S|} \cdot \mathsf{MSP}(S) \ .
\end{aligned}$$

The time to compute $\xi(S)$ is determined by the computation time of $\mathsf{msp}(i)$ for all $i \in [|S|]$ (for *identical* jobs) which is accomplished by one run of LPT for $|S|$ identical jobs that takes time $O(n \cdot \log m)$.

□

4.6.4 Efficiency Considerations for Identical Jobs

The results in this section were originally presented in [14] with the alternative definition of β-BB. Whereas [14] performs a tight (but rather technical) analysis for a variety of cases, we settle here for showing that in many cases, $\frac{2m}{m+1}$ is an upper bound on the social cost approximation achieved by the Moulin mechanisms employing our cost-sharing methods for (Q$|p_i{=}1|$C$_{\max}$) from Section 4.6.2.

Theorem 4.16. *For each* (Q$|p_i{=}1|$C$_{\max}$) *cost-sharing problem, let ξ be the cost-sharing method as defined in the proof of Theorem 4.13. Fix $\boldsymbol{v} \in \mathbb{R}^n_{\geq 0}$. Let $Q(\boldsymbol{v})$ be the players selected by Moulin$_\xi$, $\mu := |Q(\boldsymbol{v})|$ and λ be the largest cardinality of all sets that minimize the social cost $SC_{\mathsf{MSP}}(\cdot, \boldsymbol{v})$. Then for all $S \subseteq [n]$ it holds that $SC_{\mathsf{MSP}}(Q(\boldsymbol{v}), \boldsymbol{v}) \leq \gamma \cdot SC_{\mathsf{MSP}}(S, \boldsymbol{v})$, with*

$$\gamma = \begin{cases} 1 & \textit{if } \mu = \lambda \\ 1 + \frac{2m}{m+1} \cdot H_n & \textit{if } \lambda > \mu \textit{ and } \mu \leq s([m]) \\ \frac{2m}{m+1} & \textit{otherwise.} \end{cases}$$

Proof. Fix $\boldsymbol{s} \in \mathbb{N}^m$ and consider the cost-sharing problem $(\boldsymbol{1}, \boldsymbol{s})$.

Since negative bids have no impact on the social cost, we assume that $\boldsymbol{v} \in \mathbb{R}^n_{\geq 0}$. The main idea of the proof is to order the players from 1 to n such that $v_1 \geq \ldots \geq v_n$.

Then, $Q(\boldsymbol{v}) = [\mu]$, and $[\lambda]$ is the maximum set of players that minimizes $SC_{\mathsf{MSP}}(\cdot, \boldsymbol{v})$. $Moulin_\xi$ is obviously 1-EFF for $\mu = \lambda$.

Let $\sigma := s([m])$ and define $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$ by $x_k := \min_{j \in [k]} \frac{\mathsf{msp}(j)}{j}$. Then, for all $S \subseteq [n]$ with $|S| = k$ and all $i \in S$, it is $\xi_i(S) = \frac{2m}{m+1} \cdot x_k$. We frequently use that for all $i \in [\mu]$ it is $v_i \geq \frac{2m}{m+1} \cdot x_\mu$, as $Moulin_\xi$ serves $[\mu]$. Furthermore, for all $i > \mu$ it is $v_i < \frac{2m}{m+1} \cdot x_i \leq \frac{2m}{m+1} \cdot \frac{\mathsf{msp}(i)}{i}$, as $Moulin_\xi$ does not serve players $\{\mu+1, \ldots, n\}$.

- Case ($\lambda > \mu$ and $\mu \leq \sigma$):
  By non-increasing makespan costs,

$$\begin{aligned}
SC_{\mathsf{MSP}}([\mu], \boldsymbol{v}) &= \mathsf{msp}(\mu) + \sum_{i=\mu+1}^{\lambda} v_i + \sum_{i=\lambda+1}^{n} v_i \\
&< \mathsf{msp}(\lambda) + \frac{2m}{m+1} \cdot \sum_{i=\mu+1}^{\lambda} \frac{\mathsf{msp}(i)}{i} + \sum_{i=\lambda+1}^{n} v_i \\
&\leq \mathsf{msp}(\lambda) + \frac{2m}{m+1} \cdot H_n \cdot \mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{n} v_i \\
&\leq \left(1 + \frac{2m}{m+1} \cdot H_n\right) \cdot \left(\mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{n} v_i\right) \\
&= \left(1 + \frac{2m}{m+1} \cdot H_n\right) \cdot SC_{\mathsf{MSP}}([\lambda], \boldsymbol{v}) \ .
\end{aligned}$$

In order to show $\gamma = \frac{2m}{m+1}$ we split the remaining case into three subcases:

- Case ($\lambda > \mu$ and $\mu > \sigma$):
  By definition and (3.5) it holds that $x_\sigma \leq \frac{\mathsf{msp}(\sigma)}{\sigma} = \frac{1}{\sigma}$. From (3.7) we conclude that for all $i \geq \sigma$ it holds that $x_i = \frac{1}{\sigma}$. As $\mu > \sigma \geq m$ we know by (3.6) that $\mathsf{msp}(\mu) \leq \frac{\mu}{\sigma} \cdot \frac{2m}{m+1}$. Then together with $\frac{\lambda}{\sigma} \leq \mathsf{msp}(\lambda)$ by (3.4),

$$\begin{aligned}
SC_{\mathsf{MSP}}([\mu], \boldsymbol{v}) &= \mathsf{msp}(\mu) + \sum_{i=\mu+1}^{\lambda} v_i + \sum_{i=\lambda+1}^{n} v_i \\
&< \mathsf{msp}(\mu) + \frac{2m}{m+1} \cdot \frac{\lambda - \mu}{\sigma} + \sum_{i=\lambda+1}^{n} v_i \\
&\leq \frac{2m}{m+1} \cdot \mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{n} v_i \\
&\leq \frac{2m}{m+1} \cdot \left(\mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{n} v_i\right) \\
&= \frac{2m}{m+1} \cdot SC_{\mathsf{MSP}}([\lambda], \boldsymbol{v}) \ .
\end{aligned}$$

- Case ($\lambda < \mu$ and $\mu \geq m+1$):
  (3.6),(3.4), and (3.7) imply $\mathsf{msp}(\mu) \leq \frac{\mu}{\sigma} \cdot \frac{2m}{m+1}$, $\mathsf{msp}(\lambda) \geq \frac{\lambda}{\sigma}$, and $x_\mu \geq \frac{1}{\sigma}$, thus

$$\begin{aligned}
SC_{\mathsf{MSP}}([\lambda], \boldsymbol{v}) &= \mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{\mu} v_i + \sum_{i=\mu+1}^{n} v_i \\
&\geq \frac{\lambda}{\sigma} + \frac{2m}{m+1} \cdot (\mu - \lambda) \cdot x_\mu + \sum_{i=\mu+1}^{n} v_i \\
&\geq \frac{\lambda}{\sigma} + \frac{2m}{m+1} \cdot \frac{\mu - \lambda}{\sigma} + \sum_{i=\mu+1}^{n} v_i \\
&\geq \frac{\mu}{\sigma} + \sum_{i=\mu+1}^{n} v_i \\
&\geq \frac{m+1}{2m} \cdot \left(\mathsf{msp}(\mu) + \sum_{i=\mu+1}^{n} v_i \right) \\
&= \frac{m+1}{2m} \cdot SC_{\mathsf{MSP}}([\mu], \boldsymbol{v}) \ .
\end{aligned}$$

- Case ($\lambda < \mu$ and $\mu < m+1$):
  Let $k \in [\mu]$ be maximum with $x_\mu = x_k = \frac{\mathsf{msp}(k)}{k}$.
  - It holds that $\lambda > 0$: If $\lambda = 0$ and thus $[\lambda] = \emptyset$, it holds that

$$\begin{aligned}
SC_{\mathsf{MSP}}([k], \boldsymbol{v}) &= \mathsf{msp}(k) + \sum_{i=k+1}^{n} v_i \\
&= k \cdot x_\mu + \sum_{i=k+1}^{n} v_i \\
&\leq k \cdot \frac{2m}{m+1} \cdot x_\mu + \sum_{i=k+1}^{n} v_i \\
&\leq \sum_{i=1}^{n} v_i \\
&= SC_{\mathsf{MSP}}(\emptyset, \boldsymbol{v}) \ ,
\end{aligned}$$

    contradicting the maximality of λ.
  - It holds that $\lambda \geq k$: If $\lambda < k$, by $\frac{\mathsf{msp}(k)}{k} = x_k \leq x_\lambda \leq \frac{\mathsf{msp}(\lambda)}{\lambda}$, we get

$$\begin{aligned}
SC_{\mathsf{MSP}}([\lambda], \boldsymbol{v}) &= \mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{n} v_i \\
&\geq \mathsf{msp}(\lambda) + \frac{2m}{m+1} \sum_{i=\lambda+1}^{k} x_k + \sum_{i=k+1}^{n} v_i \\
&\geq \frac{\lambda}{k} \cdot \mathsf{msp}(k) + \frac{2m}{m+1} \cdot \frac{k - \lambda}{k} \cdot \mathsf{msp}(k) + \sum_{k+1}^{n} v_i \\
&\geq \mathsf{msp}(k) + \sum_{k+1}^{n} v_i \\
&= SC_{\mathsf{MSP}}([k], \boldsymbol{v}) \ ,
\end{aligned}$$

    a contradiction to λ being maximum.

- It holds that $2\lambda > \mu$: Otherwise, $\mu \geq 2k$ and $\frac{\mathsf{msp}(2k)}{2k} \leq \frac{\mathsf{msp}(k)}{k}$ by (3.3) contradicts the maximality of k.
- It holds that

$$\mu x_\mu = \frac{\mu}{k} \cdot \mathsf{msp}(k) \geq \frac{\mu}{2k} \cdot \mathsf{msp}(\mu) \geq \frac{k+1}{2k} \cdot \mathsf{msp}(\mu) \geq \frac{m+1}{2m} \cdot \mathsf{msp}(\mu) \ ,$$

which follows from $\mathsf{msp}(\mu) \leq 2 \cdot \mathsf{msp}(k)$ by (3.8), $\mu \geq \lambda + 1 \geq k+1$, and $m \geq \mu$.

- It holds that $\sum_{i=\lambda+1}^{\mu} v_i \geq \frac{1}{m} \cdot \mathsf{msp}(\mu)$: This follows from $\mu - \lambda \geq 1$, $m \geq \mu$, and thus

$$\begin{aligned}\sum_{i=\lambda+1}^{\mu} v_i &\geq (\mu - \lambda) \cdot \frac{2m}{m+1} \cdot x_\mu \\ &\geq (\mu - \lambda) \cdot \frac{\mathsf{msp}(\mu)}{\mu} \\ &\geq \frac{1}{m} \cdot \mathsf{msp}(\mu) \ .\end{aligned}$$

Finally, applying $2 \cdot \mathsf{msp}(\lambda) \geq \mathsf{msp}(2\lambda) \geq \mathsf{msp}(\mu)$ (confer (3.3)), we get

$$\begin{aligned}SC_{\mathsf{MSP}}([\lambda], \boldsymbol{v}) &= \mathsf{msp}(\lambda) + \sum_{i=\lambda+1}^{\mu} v_i + \sum_{i=\mu+1}^{n} v_i \\ &\geq \frac{1}{2} \cdot \mathsf{msp}(\mu) + \frac{1}{m} \cdot \mathsf{msp}(\mu) + \sum_{i=\mu+1}^{n} v_i \\ &= \frac{m+2}{2m} \cdot \mathsf{msp}(\mu) + \sum_{i=\mu+1}^{n} v_i \\ &\geq \frac{m+1}{2m} \cdot \left(\mathsf{msp}(\mu) + \sum_{i=\mu+1}^{n} v_i \right) \\ &= \frac{m+1}{2m} \cdot SC_{\mathsf{MSP}}([\mu], \boldsymbol{v}) \ .\end{aligned}$$

□

4.6.5 Cost-Sharing Methods in the Approximate Core

Theorem 4.17. *For each* (Q||$\mathrm{C_{max}}$) *cost-sharing problem, there is a cost-sharing method in the* $\frac{2m}{m+1}$*-core for* C_{LPT}*. For each* $S \subseteq [n]$*, the vector* $\xi(S)$ *of cost shares can be computed in time* $O(n \log n + n \cdot m)$*.*

Proof. Fix $\boldsymbol{p} \in \mathbb{N}^n$ and $\boldsymbol{s} \in \mathbb{N}^m$ and consider the cost-sharing problem $(\boldsymbol{p}, \boldsymbol{s})$. Furthermore, fix a set $S \subseteq [n]$ and consider running LPT on S. Let $\hat{S}$ be the set of jobs that LPT assigns until there is exactly one machine that has a completion time equal to $C_{\mathsf{LPT}}(S) = C_{\mathsf{LPT}}(\hat{S})$. We let ϕ' denote the assignment that LPT computes for S. For simplicity, let $\tau := |m_{\phi'}(\hat{S})|$. To define the cost-sharing method ξ, we look at the following cases:

- If $\tau < |\hat{S}|$, we define

$$\xi_i(S) := \begin{cases} \frac{2m}{m+1} \cdot \frac{p_i}{s(m_{\phi'}(\hat{S}))} & \text{if } i \in \hat{S} \\ 0 & \text{otherwise.} \end{cases}$$

- If $\tau = |\hat{S}|$, we distinguish three cases:
  - If $\tau \geq 3$, let $A := C_{\mathsf{LPT}}(S) \cdot s(m_{\phi'}(\hat{S})) - p(\hat{S})$ and define

$$\xi_i(S) := \begin{cases} \frac{2m}{m+1} \cdot \frac{p_i}{s(m_{\phi'}(\hat{S}))} & \text{if } i \in \hat{S} \text{ and } A < \frac{\tau-1}{\tau+1} \cdot p(\hat{S}) \\ \frac{2m}{m+1} \cdot \frac{p_i}{s(m_{\phi'}(\hat{S}))-s_\tau} & \text{if } i \in \hat{S} \text{ and } A \geq \frac{\tau-1}{\tau+1} \cdot p(\hat{S}) \\ 0 & \text{otherwise.} \end{cases}$$

  - If $\tau = 2$, assume without loss of generality that $\hat{S} = \{1, 2\}$. Let $z := \frac{p_1+p_2}{s_1} - \frac{p_2}{s_2}$ and define

$$\xi_i(S) := \begin{cases} \frac{p_1}{s_1} - \frac{z}{2} & \text{if } i = 1 \\ \frac{p_2}{s_1} - \frac{z}{2} & \text{if } i = 2 \\ 0 & \text{otherwise.} \end{cases}$$

  - If $\tau = 1$, assume without loss of generality that $\hat{S} = \{1\}$. Define $\xi_1(S) := \frac{p_1}{s_1}$ and $\xi_i(S) := 0$ for all $i \in S \setminus \{1\}$. It holds that $\xi_1(S) = \mathsf{MSP}(S) = C_{\mathsf{LPT}}(S)$.

We continue to show that ξ as defined above is in the $\frac{2m}{m+1}$-core for C_{LPT}.

- If $\tau < |\hat{S}|$, there is at least one machine that is assigned more than one job at the time when $\hat{S}$ is assigned. The overall cost share is larger than the actual cost $C_{\mathsf{LPT}}(S)$, since by Equation (3.10) and, obviously, $\tau \leq m$,

$$\begin{aligned} \sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{s(m_{\phi'}(\hat{S}))} \\ &\geq \frac{2m}{m+1} \cdot \frac{\tau+1}{2\tau} \cdot C_{\mathsf{LPT}}(S) \\ &\geq C_{\mathsf{LPT}}(S) \ . \end{aligned}$$

To bound the cost shares for a subset $T \subseteq S$ from above, consider an optimal assignment ϕ for $T \cap \hat{S}$ with $m_\phi(T \cap \hat{S}) \subseteq m_{\phi'}(\hat{S})$. Such an assignment exists due to (3.9). If we further utilize (3.2), we get that

$$\begin{aligned} \sum_{i \in T} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(T \cap \hat{S})}{s(m_{\phi'}(\hat{S}))} \\ &\leq \frac{2m}{m+1} \cdot \frac{p(T \cap \hat{S})}{s(m_\phi(T \cap \hat{S}))} \\ &\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T \cap \hat{S}) \\ &\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T) \ . \end{aligned}$$

- If $\tau = |\hat{S}|$, LPT has assigned at most one job to each machine at the time when $\hat{S}$ is assigned. In this case, it is $C_{\mathsf{LPT}}(S) = \frac{p_\tau}{s_\tau}$.
  - If $\tau \geq 3$, consider two cases:
    - If $A < \frac{\tau-1}{\tau+1} \cdot p(\hat{S})$, the overall share is larger than $C_{\mathsf{LPT}}(S)$, since

$$\begin{aligned}\sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{s(m_{\phi'}(\hat{S}))} \\ &> \frac{2m}{m+1} \cdot \frac{\tau+1}{2\tau} \cdot C_{\mathsf{LPT}}(S) \\ &\geq C_{\mathsf{LPT}}(S) \ .\end{aligned}$$

Furthermore, $\sum_{i \in T} \xi_i(S) \leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T)$ can be show just like for case $\tau < |\hat{S}|$.

    - If $A \geq \frac{\tau-1}{\tau+1} \cdot p(\hat{S})$, we first observe that $p_g + p_\tau \geq C_{\mathsf{LPT}}(S) \cdot s_g$ for all $g \in [\tau-1]$. This is due to the fact that the assignment computed by LPT is a Nash equilibrium. Summing up over all $g \in [\tau-1]$, we conclude $p(\hat{S}) + (\tau-2) \cdot p_\tau \geq C_{\mathsf{LPT}}(S) \cdot (s(m_{\phi'}(\hat{S})) - s_\tau)$. Further applying that $p(\hat{S}) \geq \tau p_\tau$ (confer Assumption 3.1), we get

$$\frac{2\tau-2}{\tau} \cdot p(\hat{S}) \geq C_{\mathsf{LPT}}(S) \cdot (s(m_{\phi'}(\hat{S})) - s_\tau)$$

and thus

$$\begin{aligned}\sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{s(m_{\phi'}(\hat{S})) - s_\tau} \\ &\geq \frac{2m}{m+1} \cdot \frac{\tau}{2\tau-2} \cdot C_{\mathsf{LPT}}(S) \\ &\geq C_{\mathsf{LPT}}(S) \ .\end{aligned}$$

To bound the cost share for a set $T \subseteq S$ from above, we first show that $\sum_{i \in S} \xi_i(S) \leq \frac{2m}{m+1} \cdot \mathsf{MSP}(S)$ and second we show that for all $T \subsetneq S$, $\sum_{i \in T} \xi_i(S) \leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T)$. If $\mathsf{MSP}(S) = C_{\mathsf{LPT}}(S)$, A can be written as

$$A = \mathsf{MSP}(S) \cdot \Big(s(m_{\phi'}(\hat{S})) - s_\tau\Big) - \Big(p(\hat{S}) - p_\tau\Big) \ .$$

Now, $A \geq \frac{\tau-1}{\tau+1} \cdot p(\hat{S})$ leads to

$$p(\hat{S}) \leq \frac{\tau+1}{2\tau} \cdot \Big(\mathsf{MSP}(S) \cdot \Big(s(m_{\phi'}(\hat{S})) - s_\tau\Big) + p_\tau\Big) \ .$$

Additionally, $\tau \geq 3$ implies that $s(m_{\phi'}(\hat{S})) - s_\tau \geq 2 s_\tau$ and $\frac{\tau+1}{2\tau} \leq \frac{2}{3}$. Thus,

$$\begin{aligned}\sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{(s(m_{\phi'}(\hat{S})) - s_\tau)} \\ &\leq \frac{2m}{m+1} \cdot \frac{\tau+1}{2\tau} \cdot \left(\mathsf{MSP}(S) + \frac{p_\tau}{s(m_{\phi'}(\hat{S})) - s_\tau}\right) \\ &\leq \frac{2m}{m+1} \cdot \frac{\tau+1}{2\tau} \cdot \frac{3}{2} \cdot \mathsf{MSP}(S) \\ &\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(S) \ .\end{aligned}$$

On the other hand, if $\mathsf{MSP}(S) < C_{\mathsf{LPT}}(S)$, an optimal assignment ϕ for S cannot use machines $\{\tau, \ldots, m\}$ (confer Assumption 3.1). Hence we get $m_\phi(\hat{S}) \subseteq m_\phi(S) \subseteq m_{\phi'}(\hat{S}) \setminus \{\tau\}$. Together with (3.2), this leads to

$$\begin{aligned}
\sum_{i \in S} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{(s(m_{\phi'}(\hat{S})) - s_\tau)} \\
&\leq \frac{2m}{m+1} \cdot \frac{p(\hat{S})}{s(m_\phi(\hat{S}))} \\
&\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(\hat{S}) \\
&\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(S) \ .
\end{aligned}$$

Now consider a proper subset $T \subsetneq S$. Note that there is an optimal assignment ϕ for $T \cap \hat{S}$ with $m_\phi(T \cap \hat{S}) \subseteq m_{\phi'}(\hat{S}) \setminus \{\tau\}$, since an optimal assignment for a proper subset of $\hat{S}$ does not have to use machine τ anymore. Again applying (3.2), we get

$$\begin{aligned}
\sum_{i \in T} \xi_i(S) &= \frac{2m}{m+1} \cdot \frac{p(T \cap \hat{S})}{(s(m_{\phi'}(\hat{S})) - s_\tau)} \\
&\leq \frac{2m}{m+1} \cdot \frac{p(T \cap \hat{S})}{s(m_\phi(T \cap \hat{S})} \\
&\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T \cap \hat{S}) \\
&\leq \frac{2m}{m+1} \cdot \mathsf{MSP}(T) \ .
\end{aligned}$$

– If $\tau = 2$, then

$$\sum_{i \in S} \xi_i(S) = \frac{p_2}{s_2} = \mathsf{MSP}(S) = C_{\mathsf{LPT}}(S) \ .$$

For $T \subseteq S \setminus \{1, 2\}$, $\sum_{i \in T} \xi_i(S) = 0 \leq \mathsf{MSP}(T)$. Consider $T \subseteq S \setminus \{1\}$ with $2 \in T$. We have that $\sum_{i \in T} \xi_i(S) < \frac{p_2}{s_1} \leq \mathsf{MSP}(T)$. Analogously, for $T \subseteq S \setminus \{2\}$ with $1 \in T$ it holds that $\sum_{i \in T} \xi_i(S) < \frac{p_1}{s_1} \leq \mathsf{MSP}(T)$.

– If $\tau = 1$, the core conditions are trivially satisfied.

The time to compute the cost shares $\xi(S)$ is in all cases determined by the running time of LPT with input S.

□

4.7 Conclusion and Open Problems

- We first argue that the good cases of $O(1)$-EFF in the fine-grained analysis for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems might be likely to occur. Note that only if $Moulin_\xi$ gives the service to less or equal than $\sigma := s([m])$ players, the worst case performance may arise. Now, if the provider announces that only bidding at least $\frac{1}{\sigma}$ may result in receiving the service and furthermore only maintains a certain set of machines such that his cluster may operate at full capacity, we think it likely that there are more than σ players who each bid at least $\frac{1}{\sigma}$. In our opinion, our fine-grained analysis pursues an interesting direction. It is an open question if such an analysis is promising for the problem of scheduling arbitrary jobs on parallel machines and/or for other cost-sharing problems.
- With respect to social cost efficiency of our mechanisms for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems, it is still an open issue to improve the $O(d \cdot \log n)$-EFF or our methods or to increase the lower bound of $\max\{d, H_n\}$.
- Certainly, our scheduling model may not very well reflect real-world applications. Players might not only be interested in their job being processed, but also in its completion time. Furthermore, the makespan as the completion time of the whole system may not reflect the provider's cost properly. On the other hand, the result by Brenner and Schäfer [16] rules out the existence of reasonable cost-sharing methods for minimizing the sum of completion time. We thus consider our work to be a first basic step for cost-sharing scheduling scenarios. We obtain first results for more realistic scenarios in Chapter 6.
- Finally, one has to go beyond cross-monotonicity and develop alternatives to Moulin mechanisms with better approximate budget-balance and/or social cost efficiency. We take this action in Chapters 5 and 6, where we are able to improve on some of the results in this chapter.

5

Group-Strategyproof Non-Moulin Mechanisms

5.1 Contribution

We consider the problem of computing GSP non-Moulin mechanisms with good approximate budget-balance for *symmetric* cost functions. As symmetric costs only depend on the cardinality of the served set, we approach this problem by defining exactly one vector of cost shares for each cardinality and making the cost share of a specific player solely dependent on his rank in the served set. This is realized by applying *preference-ordered* cost-sharing methods as introduced in Section 5.3. We generalize this approach for specific non-symmetric cost functions. In particular, our results are as follows:

- For symmetric and subadditive costs, we investigate what can be achieved by charging all players equally. We present GSP mechanisms that are 2-BB and show that in general, this is the best possible. If costs are additionally submodular, we even obtain 1-BB.
- We consider the case that for any set of served players, there are at most *two* different cost shares and introduce *two-price cost-sharing forms* (2P-CSFs) that define these cost shares. Furthermore, we present a mechanism $\mathit{MechCSF}_F$ and identify a validity requirement such that for any valid 2P-CSF F, $\mathit{MechCSF}_F$ is GSP. This is analogous to Moulin_ξ which is GSP if the cost-sharing method ξ is cross-monotonic. The usefulness of our new technique lies in the fact that 2P-CSFs do not necessarily represent cross-monotonic cost-sharing methods. Hence, for certain classes of cost functions, 2P-CSFs allow for better budget-balance approximations than cross-monotonic cost-sharing methods:
  - For symmetric and subadditive costs, we give an algorithm to compute valid 2P-CSFs that are $\frac{\sqrt{17}+1}{4}$-BB, where $\frac{\sqrt{17}+1}{4} \approx 1.28$. We show that in general, this is the best valid 2P-CSFs can yield. Yet, this significantly improves over 2-BB, which is generally the best possible for cross-monotonic cost shares (confer Section 4.6.5). Any such 2P-CSF F and the corresponding mechanism $\mathit{MechCSF}_F$ are computable in polynomial time, given that the symmetric costs can be evaluated in polynomial time.
  - We apply this technique to $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems obtaining $\frac{\sqrt{17}+1}{4}$-BB and to $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ cost-sharing problems where we even achieve 1-BB. Interestingly, the bound of 2 for cross-monotonic cost-shares is derived from $(\mathrm{P}|p_i{=}1|\mathrm{C}_{\max})$ costs (confer Theorem 4.7, p. 35).

  - Unfortunately, we show that the corresponding mechanisms can generally not be better than $\Omega(n)$-EFF.

- For the non-symmetric (Q||$C_{\max}$) cost-sharing problems, we use several 2P-CSFs to obtain GSP and $(\frac{\sqrt{17}+1}{4} \cdot d)$-BB mechanisms, where d is the number of different processing times. This beats the previously best-known $(2d)$-BB (confer Section 4.6). Our mechanisms are computable in polynomial time.
- For the specific non-symmetric (P3|$p_i \in \{1,2\}$|$C_{\max}$) cost-sharing problems, we extend our techniques to guarantee GSP and 1-BB by generalizing the notion of a preference order and making cost shares dependent on the rank as well as the cardinalities of *both* classes of served players. Unfortunately, the same approach fails for 4 identical machines (this result is not part of this thesis; we refer to [12]).

  The results we introduce in this chapter are published in [12].

5.2 Organization

Section 5.3 formally defines *preference-ordered* cost-sharing methods. Our results for charging all players equally are presented in Section 5.4. Section 5.5 introduces 2P-CSFs together with our novel mechanisms and give the applications for the symmetric (Q|p_i=1|$C_{\max}$) and especially (P|p_i=1|$C_{\max}$) cost-sharing problems. The applications for our techniques extended to non-symmetric cost-sharing problems are presented in Section 5.6.

5.3 Preference-Ordered Cost-Sharing Methods

We introduce a property of cost-sharing methods that ensures that the cost share of a player i in a set of served players $S \subseteq [n]$ only depends on the rank of i in S and the cardinality of S:

Definition 5.1. *A cost-sharing method* $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ *is* preference-ordered *if there are vectors* $\boldsymbol{\xi}^k \in \mathbb{R}^k_{\geq 0}$ *for all* $k \in [n]$ *such that for all* $S \subseteq [n]$

$$\xi_i(S) := \begin{cases} \xi^{|S|}_{\mathsf{rank}(i,S)} & \text{if } i \in S \\ 0 & \text{otherwise.} \end{cases}$$

Example 5.2. Let $n = 5$, $\boldsymbol{\xi}^2 = (2,1)$, and consider sets $S = \{2,4\}$ and $T = \{1,5\}$. Then $\xi(S) = (0,2,0,1,0)$ and $\xi(T) = (2,0,0,0,1)$.

5.4 Symmetric and Subadditive Costs and One Price

This section discusses the (im)possibilities of charging all players equally. Lemma 5.3 gives the positive result that GSP and 2-BB can be achieved for any symmetric and subadditive cost function whereas Lemma 5.4 establishes tightness of this result.

Lemma 5.3. *For any symmetric and subadditive cost function C, there is a GSP mechanism that is 2-BB for C and always charges players equally. If C is also submodular, this mechanism is even 1-BB for C.*

Proof. Let C be a symmetric and subadditive cost function and $c : [n] \to \mathbb{R}_{\geq 0}$ be as in Definition 2.11.

- We define

$$l_k := 2 \cdot \min_{j \in [k]} \frac{c(j)}{j} \text{ for all } k \in [n] \ .$$

For all $k \in [n]$ it holds that $k \cdot l_k \leq 2 \cdot c(k)$. For the lower bound, fix an arbitrary $k \in [n]$. Subadditivity implies for all $j \in [n]$ that $c(j) \leq c(\lceil \frac{j}{2} \rceil) + c(\lfloor \frac{j}{2} \rfloor) \leq 2 \cdot c(\lceil \frac{j}{2} \rceil)$ and especially $\frac{c(2j)}{2j} \leq \frac{c(j)}{j}$. Thus, $\min_{j \in [k]} \frac{c(j)}{j} = \frac{c(j')}{j'}$ for a $j' \in \{\lceil \frac{k}{2} \rceil, \ldots, k\}$ and $k \cdot l_k = k \cdot 2 \cdot \frac{c(j')}{j'} \geq 2 \cdot c(\lceil \frac{k}{2} \rceil) \geq c(k)$.

Define a cost-sharing method ξ as follows: for all $S \subseteq [n]$ let $\xi_i(S) := l_{|S|}$ for all $i \in S$ and $\xi_i(S) := 0$ otherwise. Clearly, ξ is cross-monotonic. Furthermore, we have that for all $S \subseteq [n]$ it holds that

$$C(S) = c(|S|) \leq |S| \cdot l_{|S|} = \sum_{i \in S} \xi_i(S) \leq 2 \cdot c(|S|) = 2 \cdot C(S) \ .$$

By Theorem 4.1, *Moulin*$_\xi$ is GSP and 2-BB for C .

- If C is also submodular, it holds that $\frac{c(j+1)}{j+1} \leq \frac{c(j)}{j}$ for all $j \in [n-1]$, which can be shown by an easy induction: Consider Definition 2.8 and without loss of generality, let $S := \{1\}$ and $T := \{2\}$. Then $c(2) - c(1) \leq c(1)$ and thus $c(1) \geq \frac{c(2)}{2}$. Now assume that $\frac{c(k+1)}{k+1} \leq \frac{c(k)}{k}$ for all $k \in [j-1]$ and without loss of generality, consider Definition 2.8 with $S := [j]$ and $T := [j+1] \setminus \{1\}$. Then, utilizing the induction hypothesis,

$$c(j+1) - c(j) \leq c(j) - c(j-1) \leq c(j) - \frac{j-1}{j} \cdot c(j) \iff \frac{c(j+1)}{j+1} \leq \frac{c(j)}{j} \ .$$

It follows that $k \cdot l_k = 2 \cdot c(k)$ for all $k \in [n]$. For $l'_k := \frac{1}{2} \cdot l_k$, the corresponding (cross-monotonic) cost-sharing method ξ' where for all $S \subseteq [n]$, $\xi'_i(S) := l'_{|S|}$ if $i \in S$ and $\xi'_i(S) := 0$ otherwise, is even 1-BB for C.

$\square$

Lemma 5.4. *For any $\varepsilon > 0$, there is a symmetric and subadditive cost function C for which no GSP mechanism that charges all players equally is $(2-\varepsilon)$-BB for C.*

Proof. Fix $0 < \varepsilon \leq 1$ and let n be large enough such that $n - 1 > \frac{2-\varepsilon}{\varepsilon}$. Define C by $C(S) := 1$ for all $S \subsetneq [n]$ and $C([n]) := 2$. Assume that there is a GSP mechanism $M = (Q, x)$ that is $(2-\varepsilon)$-BB for C and charges all players equally. As M is GSP, it induces a unique cost-sharing method ξ (confer Theorem C.1). For each set $S \subseteq [n]$, let $l(S) := \xi_1(S)$ (by assumption, $\xi_i(S) = l(S)$ for all $i \in S$).

We first observe that for each $j \in [n]$, it is $l([n] \setminus \{j\}) < l([n])$. Assume that there is a $j \in [n]$ with $l([n] \setminus \{j\}) \geq l([n])$. Then

$$(n-1) \cdot l([n] \setminus \{j\}) = \sum_{i \in [n] \setminus \{j\}} \xi_i([n] \setminus \{j\}) \leq (2-\varepsilon) \cdot C([n] \setminus \{j\}) = 2 - \varepsilon \ .$$

It follows that $n \cdot l([n] \setminus \{j\}) \leq 2 - \varepsilon + l([n] \setminus \{j\}) \leq 2 - \varepsilon + \frac{2-\varepsilon}{n-1} < 2$. However, $2 = C([n]) \leq \sum_{i \in [n]} \xi_i([n]) = n \cdot l([n]) \leq n \cdot l([n] \setminus \{j\}) < 2$ yields a contradiction.

In particular, it holds that $l([n] \setminus \{1\}) < l([n])$ and $l([n] \setminus \{2\}) < l([n])$. Define the vector of true valuations $\boldsymbol{v}$ by $v_1 := v_2 := l([n])$ and $v_j := b_j^+$ for all $j \in [n] \setminus \{1, 2\}$. It has to hold that $Q(b_1^+, \boldsymbol{v}_{-1}) = [n] \setminus \{2\}$. Otherwise, if $Q(b_1^+, \boldsymbol{v}_{-1}) = [n]$, player 2 may submit $b_2 < 0$ in order to decrease the cost-share of all other players while maintaining zero utility. Analogously, $Q(b_2^+, \boldsymbol{v}_{-2}) = [n] \setminus \{1\}$.

We finish this proof by looking at the possible values of $Q(\boldsymbol{v})$ and showing a contradiction to GSP for all of them. If $Q(\boldsymbol{v}) = [n]$, player 1 may bid b_1^+ in order to receive the service for a strictly lower cost share. If $Q(\boldsymbol{v}) = [n] \setminus \{1\}$ or $Q(\boldsymbol{v}) = [n] \setminus \{1, 2\}$, player 1 may bid b_1^+ in order to now receive the service for a strictly positive utility. If $Q(\boldsymbol{v}) = [n] \setminus \{2\}$, the same holds for player 2.

□

5.5 Symmetric Costs and Two Prices

5.5.1 Two-Price Cost-Sharing Forms

In this section, we apply specific preference-ordered cost-sharing methods (confer Definition 5.1) with the property that they charge each set of players at most two different cost shares. The corresponding vectors $\boldsymbol{\xi}^1, \ldots \boldsymbol{\xi}^n$ are such that each $\boldsymbol{\xi}^k$ for each cardinality $k \in [n]$ is of the form $\boldsymbol{\xi}^k := (h_k, \ldots, h_k, l_k, \ldots, l_k)$ with $h_k > l_k$. Players who are first in the order (the *disadvantaged* players) pay the higher cost share h_k and the remaining players (the *advantaged* players) pay the lower cost share l_k. We require that there is always at least one advantaged player and define the number of disadvantaged players in vector $\boldsymbol{\xi}^k$ by $d_k := |\max\{i \in [k-1] \mid \xi_i^k > \xi_k^k\}|$. We call a contiguous range $\{s, s+1, \ldots, t\} \subseteq [n]$ of cardinalities with $d_s = d_{t+1} = 0$ (let $d_{t+1} := 0$ if $t = n$), and $d_k > 0$ for $k \in \{s+1, \ldots, t\}$ a *segment*. That is, only at the beginning of a segment there is no disadvantaged player paying the higher cost share.

In order to succinctly represent the vectors $\{\boldsymbol{\xi}^k\}_{k \in [n]}$, we use *two-price cost-sharing forms* (2P-CSFs):

Definition 5.5. *A* two-price cost-sharing form *(2P-CSF) is a tuple*

$$F = (n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d}) \text{ with } n \in \mathbb{N},\ \boldsymbol{h}, \boldsymbol{l} \in \mathbb{R}^n_{\geq 0},\ \text{and } \boldsymbol{d} \in \mathbb{N}^n_0 \ .$$

We furthermore define β-BB of 2P-CSFs analogously to the β-BB of the cost-sharing methods they induce:

Definition 5.6. *Let $F = (n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ be a 2P-CSF, C be a symmetric cost function and $c : [n] \to \mathbb{R}_{\geq 0}$ as in Definition 2.11. F is β-BB for C if for all $k \in [n]$,*

$$c(k) \leq d_k \cdot h_k + (k - d_k) \cdot l_k \leq \beta \cdot c(k) \ .$$

5.5.2 Validity of Two-Price Cost-Sharing Forms

We give a validity requirement of 2P-CSFs that allows for defining GSP mechanisms, as we show in Section 5.5.3.

Definition 5.7. *A 2P-CSF* $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ *is* valid *if* (5.1)– (5.7) *hold:*

- *If there is only one player, he pays the lower cost share:*

$$d_1 = 0 \tag{5.1}$$

- *Higher cost shares are strictly larger than lower cost shares:*

$$\text{For all } k \in [n] : h_k > l_k \tag{5.2}$$

- *Lower cost shares are non-increasing:*

$$\text{For all } k \in [n] \setminus \{1\} : l_k \leq l_{k-1} \tag{5.3}$$

- *Lower cost shares stay the same within a segment:*

$$\text{For all } k \in [n] \setminus \{1\} : (l_k < l_{k-1}) \Longrightarrow d_k = 0 \tag{5.4}$$

- *The number of disadvantaged players increases by at most one:*

$$\text{For all } k \in [n] \setminus \{1\} : d_k \leq d_{k-1} + 1 \tag{5.5}$$

- *Higher cost shares may only increase at the beginning of a segment:*

$$\text{For all } k \in [n] \setminus \{1\} : (h_k > h_{k-1}) \Longrightarrow d_k = 0 \tag{5.6}$$

- *Higher cost shares may only decrease at the beginning of a segment or if there is only one disadvantaged player:*

$$\text{For all } k \in [n] \setminus \{1\} : (h_k < h_{k-1}) \Longrightarrow d_k \leq 1 \tag{5.7}$$

Note that (5.1) and (5.5) imply that there is always at least one advantaged player. Example 5.8 gives a valid 2P-CSF. The segments are $\{1, 2\}, \{3, \ldots, 11\}$ and $\{12, 13\}$.

Example 5.8. A valid 2P-CSF with resulting vectors $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^n$ for $n = 13$:

k	h_k	l_k	d_k	$\boldsymbol{\xi}^k$
1	9	3	0	(3)
2	9	3	1	$(9, 3)$
3	6	2	0	$(2, 2, 2)$
4	6	2	1	$(6, 2, 2, 2)$
5	6	2	2	$(6, 6, 2, 2, 2)$
6	6	2	3	$(6, 6, 6, 2, 2, 2)$
7	6	2	4	$(6, 6, 6, 6, 2, 2, 2)$
8	6	2	2	$(6, 6, 2, 2, 2, 2, 2, 2)$
9	5	2	1	$(5, 2, 2, 2, 2, 2, 2, 2, 2)$
10	4	2	1	$(4, 2, 2, 2, 2, 2, 2, 2, 2, 2)$
11	4	2	2	$(4, 4, 2, 2, 2, 2, 2, 2, 2, 2, 2)$
12	7	1	0	$(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$
13	7	1	1	$(7, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$

5.5.3 GSP Mechanisms for Two-Price Cost-Sharing Forms

Before we present our mechanisms that are GSP for any valid 2P-CSF, we give an intuition how the mechanism for the 2P-CSF in Example 5.8 works. For $n = 13$, consider $\boldsymbol{v} = (-1, -1, 3, 2, 2, 6, 6, 2, 2, 2, 2, 5, 5)$. As there are less than 12 players with $v_i \geq 1$, we can at most serve 11 players. On the other hand, we should serve at least 3 players, as there are more than 3 players with $v_i \geq 2$ and serving less players is not consistent with GSP:

- If we only serve 1 or 2 players with $v_i \geq 3$, a group of 3 players, composed of the served players and a sufficient number of players with $v_i \geq 2$, may bid b_i^+ and all other players may bid -1. As a result, the players originally served for a cost share of at least 3 now pay a cost share of 2.
- If we serve no player, players $3, 4$ and 5 may bid b_i^+ and all others may bid -1 in order to provide player 3 with a strictly positive utility.

Thus, the number of served players is somewhere in the segment $\{3, 11\}$. Serving all remaining players who exhibit positive valuations and charging cost shares according to $\boldsymbol{\xi}^{11}$ would imply a cost share of 4 for the disadvantaged players 3 and 4. We first try to reject a suitable subset of *indifferent* players with $v_i = 2$, such that all players with $v_i > 2$ are served for the low cost share of 2. However, at this point, we need 8 indifferent players while there are only 6. Now, as the bid of the least preferred player 3 is smaller than $h_{11} = 4$, he is rejected. In the same fashion, we reject players 4 and 5 as $v_4 < h_{10}$ and $v_5 < h_9$, and there are not enough indifferent players. Thereby, the number of indifferent players reduces to 4.

In order to charge especially player 6 the low cost share, we need 5 indifferent players. Nevertheless, as player 6 can pay for the high cost share, he receives the service for $h_8 = 6$. But now, the number of indifferent players we need in order to serve all remaining players for the low cost share of 2 is only 4. Thus, we reject players 8 to 11 and serve $S = \{6, 7, 12, 13\}$, where $\xi(S) = (0, 0, 0, 0, 0, 6, 2, 0, 0, 0, 0, 2, 2)$ as induced by $\boldsymbol{\xi}^4$.

The intuition is that including the least preferred player for the current higher cost share never harms the other players. Instead, it may even benefit in that more players can be served for the low cost share afterwards. Once a player is included for a higher cost share, this cost share remains fixed during the further execution.

We remark that sequential stand alone mechanisms for submodular costs use the same approach (confer Appendix B.1). However, whereas subsequent players are irrelevant to a specific player for sequential stand alone mechanisms, they are highly relevant for our mechanism. On the one hand, we account for rejecting indifferent players, and on the other hand, it is not obvious that subsequent non-indifferent player are incapable of helping a preceding player.

Before formally defining our mechanism, we give the auxiliary values $\rho(k, j)$ for all $k \in [n]$ and all $j \in [k]$ indicating how many players i with $\mathsf{rank}(i, S) \geq j$ in any set S with $|S| = k$ are required to not receive the service such that at most the first $j - 1$ players in S are disadvantaged:

Definition 5.9. *Let $F = (n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ be a valid 2P-CSF. For all $k \in [n]$ and all $j \in [k]$, define*

$$\rho(k, j) := \begin{cases} 0 & \text{if } d_k < j \\ \min\{i \in [k - j] \mid d_{k-i} \leq j - 1\} & \text{otherwise.} \end{cases} \tag{5.8}$$

Note that the values $\rho(k,j)$ are well defined. If $j \in [d_k]$, we know that especially $d_j < j$ as by validity, there is always at least one advantaged player. Note that by validity property (5.5), for $j \in [d_k]$ it holds that $\min\{i \in [k-j] \mid d_{k-i} \leq j-1\} = \min\{i \in [k-j] \mid d_{k-i} = j-1\}$. Here, $\rho(k,j)$ even gives the number of required players such that *exactly* the first $j-1$ players are disadvantaged.

Example 5.10. For the 2P-CSF from Example 5.8 it is $\rho(11,1) = 8, \rho(11,2) = 1$, and $\rho(11,j) = 0$ for all $2 < j \leq 11$.

Now we are ready to state mechanism $\mathit{MechCSF}_F$ in Algorithm 5.1:

Algorithm 5.1 (computing $\mathit{MechCSF}_F(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$ **for** $F = (n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$**).**

1: $k := \max\left\{ j \in [n]_0 \;\middle|\; |\{i \in [n] \mid b_i \geq l_j\}| \geq j \right\}$ ▷ Starting cardinality
2: **if** $k = 0$ **then** return $(\emptyset, \boldsymbol{0})$
3: $l := l_k$ ▷ Low cost-share
4: $A := \{i \in [n] \mid b_i \geq l\}$ ▷ (potential) Advantaged players
5: $I := \{i \in [n] \mid b_i = l\}$ ▷ (alleged) Indifferent players
6: $D := \emptyset$ ▷ Disadvantaged players
7: **loop**
8: $\quad \ell := \min A$ ▷ Least preferred player
9: $\quad r := \mathsf{rank}(\ell, A \cup D)$
10: $\quad$ **if** $|I| \geq \rho(|A| + |D|, r)$ **then** ▷ Sufficient indifferent players?
11: $\qquad A := A \setminus \{\rho(|A| + |D|, r)$ largest elements of $I\}$
12: $\qquad$ **break**
13: $\quad$ **if** $b_\ell \geq h_{|A|+|D|}$ **then** $D := D \cup \{\ell\}$ ▷ Make disadvantaged
14: $\quad A := A \setminus \{\ell\}$; $I := I \setminus \{\ell\}$
15: $x_i := h_{|A|+|D|}$ for $i \in D$; $x_i := l$ for $i \in A$; $x_i := 0$ otherwise
16: return $(A \cup D, \boldsymbol{x})$

It is easy to verify that for every β-BB 2P-CSFF, $\mathit{MechCSF}_F$ is β-BB and meets NPT, VP, and strict CS. Lemma 5.11 will be helpful for finally proving GSP of $\mathit{MechCSF}_F$ in Theorem 5.12:

Lemma 5.11. *For any valid 2P-CSF* $F = (n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ *it holds for all* $k \in [n]$ *and all* $j \in [d_k]$ *that*

$$\rho(k, j+1) \leq \rho(k,j) - 1 \text{ and} \tag{5.9}$$

$$\rho(k-1, j) = \rho(k,j) - 1 \ . \tag{5.10}$$

Proof. To see (5.9), let $k' < k$ be maximum with $d_{k'} = j-1$, i.e., $\rho(k,j) = k - k'$. Furthermore, let $k'' \leq k$ be maximum with $d_{k''} = j$, i.e., $\rho(k, j+1) = k - k''$. Either $k'' = k$ and consequently $\rho(k, j+1) = 0 \leq k - k' - 1 = \rho(k,j) - 1$, or $k' < k'' < k$ (by validity property (5.5)) implying that $\rho(k, j+1) = k - k'' \leq k - k' - 1 = \rho(k,j) - 1$. The second property (5.10) is straightforward to show as for $k' < k$ being maximum with $d_{k'} = j - 1$, $\rho(k,j) = k - k'$ and $\rho(k-1,j) = k - 1 - k'$.

□

Note that Lemma 5.11 especially implies that adding a player ℓ to set D in line 13 never harms subsequent players in $A \setminus \{\ell\}$. To see this, look at the number of indifferent players needed to evaluate line 10 to true. If ℓ is added to D, by (5.9), this number decreases by *at least* one for the next iteration. On the other hand, if ℓ is not added, by (5.10), it only decreases by *exactly* one. For an intuition, the 2P-CSF from Example 5.8 yields $\rho(11,1) = 8$ and $\rho(11,2) = 1$, but $\rho(10,1) = 7$.

Theorem 5.12. *For any valid 2P-CSF F, $MechCSF_F$ is GSP.*

Proof. For any input $\boldsymbol{b} \in \mathbb{R}^n$, denote by $k(\boldsymbol{b})$ and $l(\boldsymbol{b})$ the values of k and l in lines 1 and 3 of Algorithm 5.1. Moreover, set $s(\boldsymbol{b}) := \max\{j \in [k(\boldsymbol{b})] \mid d_j = 0\}$ to the beginning of the segment that $k(\boldsymbol{b})$ is in (if $k(\boldsymbol{b}) = 0$, let $s(\boldsymbol{b}) = 0$). Note that $s(\boldsymbol{b}) \leq |Q(\boldsymbol{b})| \leq k(\boldsymbol{b})$.

Assume that $MechCSF_F$ is not GSP. Then, there exists a coalition $K \subseteq [n]$ with true valuations $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$ and a bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for all $i \in K$, with at least one strict inequality. Without loss of generality, let $\boldsymbol{v} := (\boldsymbol{v}_K, \boldsymbol{b}_{-K})$.

- Without loss of generality, we may assume that $s(\boldsymbol{b}) = s(\boldsymbol{v})$:
  - If $s(\boldsymbol{b}) > s(\boldsymbol{v})$, it holds that $s(\boldsymbol{b}) > k(\boldsymbol{v})$, and thus $|Q(\boldsymbol{b})| > k(\boldsymbol{v})$. By choice of $k(\boldsymbol{v})$, this means that the number of players receiving the service for $\boldsymbol{b}$ is strictly larger than the number of players who can actually afford to pay at least $l(\boldsymbol{b})$. Formally, there is a player $j \in Q(\boldsymbol{b})$ with $v_j < l(\boldsymbol{b})$ because $k(\boldsymbol{v})$ would not have been maximal in line 1 otherwise. Since $j \in Q(\boldsymbol{b})$, it holds that $b_j \geq x_j(\boldsymbol{b}) \geq l(\boldsymbol{b}) > v_j$, so we have $j \in K$ and $u_j(\boldsymbol{b}, v_j) < 0 \leq u_j(\boldsymbol{v}, v_j)$, a contradiction.
  - If $s(\boldsymbol{b}) < s(\boldsymbol{v})$, it holds that $k(\boldsymbol{b}) < s(\boldsymbol{v})$ and thus $|Q(\boldsymbol{b})| < s(\boldsymbol{v})$. Here, the idea is to change the bid vector $\boldsymbol{b}$ to $\boldsymbol{b}'$ in such a way that exactly $s(\boldsymbol{v})$ players receive the service for $\boldsymbol{b}'$ while utilities can only improve. Formally, let $L := \{j \in [n] \mid b_j \geq l(\boldsymbol{v})\}$. Clearly, $|L| < s(\boldsymbol{v})$ as $k(\boldsymbol{b})$ would not have been maximal in line 1 otherwise. Now, let M be a set of $s(\boldsymbol{v}) - |L|$ players $j \in [n] \setminus L$ with $l(\boldsymbol{v}) \leq v_j$. Such a set M exists. Define a new bid vector $\boldsymbol{b}' \in \mathbb{R}^n$ by $b'_j := l(\boldsymbol{v})$ for $j \in M$ and $b'_j := b_j$ otherwise. Then $s(\boldsymbol{b}') = |Q(\boldsymbol{b}')| = k(\boldsymbol{b}') = s(\boldsymbol{v})$ and $u_j(\boldsymbol{b}', v_j) \geq u_j(\boldsymbol{b}, v_j)$ for all players $j \in [n]$.

Let now $l := l(\boldsymbol{b}) = l(\boldsymbol{v})$ and $\mathcal{L} := \{\ell_1, \ldots, \ell_t\}$ be the players that are considered for $\boldsymbol{v}$ in Line 8, where player ℓ_k is examined in loop $k \in [t]$.

- Without loss of generality, we may assume that $K \subseteq \mathcal{L}$:
  For input $\boldsymbol{v}$, players $i \notin \mathcal{L}$ either receive the service for l or are rejected due to $v_i = l$. These players can neither strictly improve their utility for another outcome in the segment, nor change the outcome for $\boldsymbol{v}$ by bidding $b_i \neq v_i$ without strictly decreasing their utility.

Let ℓ_j be the smallest player in $\mathcal{L} \cap K$ that strictly increases his utility and define $\mathcal{L}^{<j} := \{\ell_1, \ldots, \ell_{j-1}\}$ and $\mathcal{L}^{>j} := \{\ell_{j+1}, \ldots, \ell_t\}$.

- Without loss of generality, we may assume that $K \subseteq \mathcal{L}^{>j} \cup \{\ell_j\}$:
  Players $\mathcal{L}^{<j} \cap Q(\boldsymbol{v})$ do not harm subsequent players if they continue to receive the service (confer Lemma 5.11), and players in $\mathcal{L}^{<j} \setminus Q(\boldsymbol{v})$ strictly decrease their utility if they bid in order to receive the service.

- Without loss of generality, we may assume that $v_i > l$ for all $i \in K$ and that $b_i = l$ and $i \notin Q(\boldsymbol{b})$ for all $i \in K \setminus \{\ell_j\}$:
  The only way for player $i \in K \setminus \{\ell_j\}$ to be beneficial for player ℓ_j is to bid in order to not receive the service. Players with $v_i = l$ are considered to be rejected anyway, thus $v_i > l$ for all $i \in K$. Players $i \in K \setminus \{\ell_j\}$ may bid $b_i = l$ as the mechanism then rejects these players as desired. Without loss of generality we may consider a coalition with minimum cardinality, thus $i \notin Q(\boldsymbol{b})$ for all $i \in K \setminus \{\ell_j\}$.

By the above assumptions, the runs of both mechanisms with inputs $\boldsymbol{v}$ and $\boldsymbol{b}$ are the same up to consideration of player ℓ_j. Let ρ be the number of players needed in Line 10 in loop j of $MechCSF_F$ for inputs $\boldsymbol{v}$ and $\boldsymbol{b}$, respectively. Define

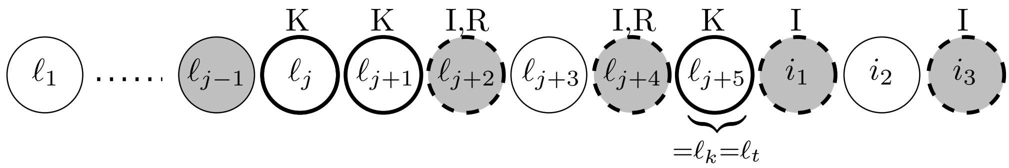
$$I := \{i \in [n] \mid v_i = l\} \setminus \{i \in \mathcal{L}^{<j} \mid v_i = l\} .$$

Figure 5.1 particularly illustrates the sets K and I and is helpful for understanding the remaining part of the proof.

As there were not enough indifferent players for $\boldsymbol{v}$ to be rejected, it holds that $|I| < \rho$. For input $\boldsymbol{b}$ however, there are just enough (alleged) indifferent players, thus $|K \setminus \{\ell_j\}| + |I| = \rho$. Define ℓ_k to be the player in $K \setminus \{\ell_j\}$ that is highest in the order. Let $R := I \cap \{\ell_{j+1}, \ldots, \ell_{k-1}\}$. By definition, $|K \setminus \{\ell_j\}| + |R| \leq k - j$. Let ρ' be the number of indifferent players needed for ℓ_k for input $\boldsymbol{v}$. It is $\rho' \leq \rho - (k - j)$ by iteratively applying Theorem (5.11.) and thus $\rho' \leq \rho - |K \setminus \{\ell_j\}| - |R| = |I| - |R|$. It follows that ℓ_k receives the service for $\boldsymbol{v}$ and pays l. A contradiction, as $\ell_k \notin Q(\boldsymbol{b})$ and $v_k > l$, and thus $u_{\ell_k}(\boldsymbol{b}, v_{\ell_k}) = 0 < v_{\ell_k} - l = u_{\ell_k}(\boldsymbol{v}, v_{\ell_k})$.

$\square$

Fig. 5.1. Sketch for the proof of Theorem 5.12. Shows sequence of player $\ell_1, \ldots, \ell_t$ considered in the run for $\boldsymbol{v}$, and players i_1, i_2, i_3 still in the game for $\boldsymbol{v}$ after deleting players i with $v_i < l$. Shaded players i are those with $v_i = l$.



Lemma 5.13. *For each 2P-CSF F that is computable in time $O(t)$, $MechCSF_F$ is computable in time $O(t + n^2)$.*

Proof. For the running time, consider Algorithm 5.1. In the worst case, it has to compute the whole 2P-CSF F in time $O(t)$. Given these values, observe that every operation outside the loop takes time at most $O(n^2)$. The loop is executed at most n times and each operation in the loop takes at most time $O(n)$. An overall running time of $O(t + n^2)$ results.

$\square$

5.5.4 $\frac{\sqrt{17}+1}{4}$-BB Two-Price Cost-Sharing Forms for Subadditive Costs

Theorem 5.14 is the main theorem of this section. It is an immediate corollary of Theorem 5.15, Theorem 5.16, Theorem 5.12, and Lemma 5.13.

Theorem 5.14. *For any symmetric and subadditive cost function C, there is a 2P-CSF F such that $MechCSF_F$ is GSP and $\frac{\sqrt{17}+1}{4}$-BB for C. Moreover, if C is given as an array of n function values, it can be computed in time $O(n^2)$.*

Algorithm 5.2 computes a 2P-CSF for increasing cardinalities.

Algorithm 5.2 (computing a 2P-CSF for subadditive costs).
Input: symmetric and subadditive cost function C (c as in Definition 2.11)
Output: valid 2P-CSF $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$

1: $\boldsymbol{h} := \boldsymbol{l} := \boldsymbol{d} := \boldsymbol{0}$; $\beta := \frac{\sqrt{17}+1}{4}$
2: $l_1 := \beta \cdot c(1)$; $h_1 := \infty$; $d_1 := 0$; $f := 1$
3: **for** $k := 2, \ldots, n$ **do**
4:     **if** $\beta \cdot \frac{c(k)}{k} \leq l_f$ **then** $l_k := \beta \cdot \frac{c(k)}{k}$; $h_k := \infty$; $d_k := 0$; $f := k$
5:     **else**
6:         $l_k := l_{k-1}$; $h_k := \min\{\beta \cdot c(k) - (k-1) \cdot l_k, h_{k-1}\}$
7:         **if** $h_k + (k-1) \cdot l_k < c(k)$ **then**
8:             $d_k := 2$
9:         **else if** $h_k + (k-1) \cdot l_k \geq 2 \cdot c(f)$ **then**
10:             $d_k := 1$
11:         **else if** $h_k \geq (\beta^2 - \beta) \cdot c(f)$ **then**
12:             $d_k := 1$
13:             **if** $(\beta^2 - \beta) \cdot c(f) + (k-1) \cdot l_k \geq c(k)$ and $f \geq 4$ **then**
14:                 $h_k := (\beta^2 - \beta) \cdot c(f)$
15:         **else**
16:             $d_k := 0$; $h_k := \infty$
17: **return** $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$

Here, "∞" is a placeholder for any "sufficiently large" value (a value strictly larger than $\beta \cdot c(f)$ is sufficient) to simplify the presentation.

Theorem 5.15. *For any symmetric cost function C, given as an array of n function values, the 2P-CSF computed by Algorithm 5.2 is valid and computable in time $O(n)$.*

Proof. Fix a symmetric cost function C and define $c : [n] \to \mathbb{R}_{\geq 0}$ according to Definition 2.11. Let $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ be the output 2P-CSF. Consider an arbitrary cardinality $k \in [n]$. Clearly, $d_i \in \{0, 1, 2\}$.

- Condition (5.1) ($d_1 = 0$) trivially holds by line 2.
- Condition (5.2) ($h_k > l_k$) is shown by induction: Obviously, $h_1 = \infty > \beta \cdot c(1) = l_1$. Assume that $h_j > l_j$ for all $j \in [k-1]$. In the case $h_k = \infty$ there is nothing to show. If h_k is set to some different value, then especially line 4 evaluated to false. We define $f := \max\{j \in [k-1] \mid \beta \cdot \frac{c(j)}{j} = l_k\}$ to be the last cardinality previous to k for which the lower cost share was set in lines 2 or 4. Now, $\frac{c(k)}{k} > \frac{c(f)}{f} = \frac{l_f}{\beta}$ and $l_k = \ldots = l_f$. The value h_k is set for the first time in line 6, and either $h_k = h_{k-1} > l_{k-1} = l_k$ or $h_k = \beta \cdot c(k) - (k-1) \cdot l_k > l_k$. If h_k is set in line 14, $h_k = (\beta^2 - \beta) \cdot c(f) = (\beta - 1) \cdot f \cdot l_k > 0.28 \cdot 4 \cdot l_k > l_k$.
- Condition (5.3) ($l_k \leq l_{k-1}$) and Condition (5.4.) ($l_k < l_{k-1} \implies d_k = 0$) hold since $l_k \neq l_{k-1}$ only if l_k was set in line 4.
- Condition (5.5) ($d_k \leq d_{k-1} + 1$) has only be checked for $d_k = 2$. Then $h_k < c(k) - (k-1) \cdot l_k$ because line 7 evaluated to true and thus $h_k = \min\{\beta \cdot c(k) - (k-1) \cdot l_k, h_{k-1}\} = h_{k-1}$. Now assume $d_k > d_{k-1} + 1$, i.e., $d_{k-1} = 0$. Then, $h_k = h_{k-1} = \infty$, a contradiction to $h_k < c(k) - (k-1) \cdot l_k$.
- Condition (5.6) ($h_k > h_{k-1} \implies d_k = 0$) holds since line 6 ensures that from $d_k > 0$ it follows that $h_k \leq h_{k-1}$.

- Condition (5.7) ($h_k < h_{k-1} \Longrightarrow d_k \leq 1$) holds as for $d_k = 2$ it is $h_k = h_{k-1}$ as shown for Condition (5.5.).

Computability in $O(n)$ is obvious. □

Theorem 5.16. *For any subadditive and symmetric cost function C, the 2P-CSF computed by Algorithm 5.2 is $\frac{\sqrt{17}+1}{4}$-BB.*

Proof. Fix a subadditive and symmetric cost function C and let c be as in Definition 2.11. For all $k \in [n]$, we define $\gamma(k) := d_k \cdot h_k + (k - d_k) \cdot l_k$ and show that $c(k) \leq \gamma(k) \leq \frac{\sqrt{17}+1}{4} \cdot c(k)$ for all $k \in [n]$.

- If $d_k = 1$, h_k is either set in line 6 or in line 14. In both cases $h_k \leq \beta \cdot c(k) - (k-1) \cdot l_k$ and thus $\gamma(k) = h_k + (k-1) \cdot l_k \leq \beta \cdot c(k)$. Furthermore, as line 7 evaluated to false and h_i is only reset to $(\beta^2 - \beta) \cdot c(f)$ if line 13 evaluates to true, $\gamma(k) \geq c(k)$.

For $d_k \in \{0, 2\}$, we define $f := \max\{j \in [k] \mid \beta \cdot \frac{c(j)}{j} = l_k\}$ to be the last cardinality previous or equal to k for which the lower cost share was set in lines 2 or 4. Furthermore, let $g := \min(\{j \in \{k+1, k+2, \ldots, n\} \mid \beta \cdot \frac{c(j)}{j} \leq l_k\} \cup \{n+1\})$ be the next such cardinality after k (or $g = n + 1$ if f is the largest such cardinality). It is $f \leq k < g \leq 2f$. Otherwise, if $g > 2f$, g would not be minimum due to $\frac{c(2f)}{2f} \leq \frac{2 \cdot c(f)}{2f} = \frac{c(f)}{f}$ because of subadditivity. Since c is non-decreasing, $c(k) \leq c(2f) \leq 2 \cdot c(f)$. Set $h'_k := \min\{\beta \cdot c(k) - (k-1) \cdot l_k, h_{k-1}\}$. We will make use of the following property: for $k \geq 3$,

$$d_{k-1} = 1 \text{ and } \gamma(k-1) \geq 2 \cdot c(f) \Longrightarrow \forall j \in \{k, k+1, \ldots, g-1\} : d_j = 1 \,. \quad (5.11)$$

Proof of (5.11): If $h'_k = h_{k-1}$, then $h'_k + (k-1) \cdot l_k = \gamma(k-1) + l_k$. If $h'_k = \beta \cdot c(k) - (k-1) \cdot l_k$, then $h'_k + (k-1) \cdot l_k = \beta \cdot c(k) \geq \beta \cdot c(k-1) \geq \gamma(k-1)$. In any case, $h'_k + (k-1) \cdot l_k \geq \gamma(k-1) \geq 2 \cdot c(f) \geq c(k)$ and line 9 evaluates to true. Then $h_k = h'_k$ and $d_k = 1$. Inductively, (5.11) follows.

- Consider $d_k = 2$. Observe that $d_1 = 0$ and $d_2 = 0$ due to $\frac{c(2)}{2} \leq \frac{c(1)}{1}$. Thus, by validity, $k \geq 4$. We first show $h_k = (\beta^2 - \beta) \cdot c(f)$. Define $k' := \max\{j \in [k] \mid d_j = 1\}$. Let $s := \max\{j \in [k] \mid d_k = 0\}$ be the start of the segment that k is in. By validity, $f \leq s < k' < k$, $d_{s+1} = 1$, and $h_{s+1} \geq h_k = h_{k'}$.
  Since line 9 evaluated to false for cardinality k' because of (5.11), line 11 must have evaluated to true, implying $h_k = h'_k \geq (\beta^2 - \beta) \cdot c(f)$. Now assume '>'. From $l_{s+1} = l_s = \frac{\beta}{f} \cdot c(f) \geq \frac{\beta}{s} \cdot c(f)$ and the fact that line 13 evaluated to false for cardinality $(s+1)$ (as by assumption $h_{s+1} \geq h_k > (\beta^2 - \beta) \cdot c(f)$), we conclude that $\beta^2 \cdot c(f) \leq (\beta^2 - \beta) \cdot c(f) + s \cdot l_{s+1} < c(s+1)$.
  Then, however, $h_{s+1} + s \cdot l_{s+1} = \beta \cdot c(s+1) > \beta^3 \cdot c(f) > 2 \cdot c(f)$, a contradiction, since line 9 must have evaluated to false due to (5.11). Hence, $h_k = (\beta^2 - \beta) \cdot c(f)$ and thus

$$\begin{aligned}
\gamma(k) &= 2h_k + (k-2) \cdot l_k \\
&= 2h_k + (k-2) \cdot \beta \cdot \frac{c(f)}{f} \\
&\geq (2\beta^2 - \beta) \cdot c(f) \\
&= 2 \cdot c(f) \\
&\geq c(k) \ .
\end{aligned}$$

Also, $\beta \cdot c(f) < \beta^2 \cdot c(f) = h_k + \beta \cdot c(f) \leq h_k + s \cdot l_{s+1} < h_k + (k-1) \cdot l_k < c(k)$, where the last inequality holds since line 7 evaluated to true. Applying the inequalities above, we get

$$\begin{aligned}\gamma(k) &= 2h_k + (k-2) \cdot l_k \\ &< h_k + c(k) \\ &= (\beta^2 - \beta) \cdot c(f) + c(k) \\ &< (\beta - 1) \cdot c(k) + c(k) \\ &= \beta \cdot c(k) \ .\end{aligned}$$

- Now let $d_k = 0$. If $k = f$, then $\gamma(k) = \beta \cdot c(k)$. In the following, we consider $k > f$. Since line 11 evaluated to false, $h'_k < (\beta^2 - \beta) \cdot c(f)$. We first show that $h'_k = \beta \cdot c(k) - (k-1) \cdot l_k$. Assume otherwise. Then, $h'_k = h_{k-1}$ and $d_{k-1} = 1$ since $h_{k-1} \notin \{\infty, (\beta^2 - \beta) \cdot c(f)\}$. Yet, line 9 evaluated to false for $(k-1)$ because of (5.11). Thus, $h'_k = h_{k-1} \geq (\beta^2 - \beta) \cdot c(f)$ by line 11 which contradicts $h'_k < (\beta^2 - \beta) \cdot c(f)$.
  Now $\beta \cdot c(k) = h'_k + (k-1) \cdot l_k < (\beta - 1) \cdot \beta \cdot c(f) + (k-1) \cdot l_k$. Furthermore, $\gamma(k) = k \cdot l_k = k \cdot l_f = k \cdot \frac{\beta}{f} \cdot c(f) > \beta \cdot c(f)$ and $\gamma(k) > (k-1) \cdot l_k$. Putting everything together gives
  $$c(k) = \frac{\beta \cdot c(k)}{\beta} \leq \frac{(\beta - 1) \cdot \gamma(k) + \gamma(k)}{\beta} = \gamma(k) \,.$$
  Moreover, as line 4 evaluated to false, $\gamma(k) = k \cdot l_f < k \cdot \frac{\beta}{k} \cdot c(k) = \beta \cdot c(k)$.

□

Theorem 5.17 shows that $\frac{\sqrt{17}+1}{4}$-BB is in general the best that can be achieved with our technique.

Theorem 5.17. *For all $\varepsilon > 0$, there is a symmetric and subadditive cost function C for which no valid $\left(\frac{\sqrt{17}+1}{4} - \varepsilon\right)$-BB 2P-CSF exists.*

Proof. Fix $\beta := \frac{\sqrt{17}+1}{4}$, let $0 < \varepsilon \leq \beta - 1$, and set $\alpha := \beta - \varepsilon$. Additionally, let $j, l \in \mathbb{N}$ with $l > \ln_\beta \frac{\beta-1}{\varepsilon}$ and $j > \frac{(l+1) \cdot \alpha}{\varepsilon} = \frac{(l+1) \cdot \beta}{\varepsilon} - (l+1)$. Set $m := j + l + 1$ and $n := m + 1$ and consider a symmetric cost function C that induces c defined below:

k	$1 \cdots j$	$j+1$	$j+2$	$\cdots$	$j+l$	m	n
$c(k)$	$1 \cdots 1$	$\beta - \frac{\beta-1}{\beta^1}$	$\beta - \frac{\beta-1}{\beta^2}$	$\cdots$	$\beta - \frac{\beta-1}{\beta^l}$	β	2

Clearly, C is subadditive. Now assume there is a valid 2P-CSF $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ which is α-BB. For all $k \in [n]$, we again define $\gamma(k) := d_k \cdot h_k + (k - d_k) \cdot l_k$.

The rough idea is the following: It can be shown that $d_m \geq 1$ and $d_n = d_m + 1$. Let $s := \max\{k \in [m-1] \mid d_k = 0\}$ be the start of the segment that cardinality n is in. Clearly, $s < m$ and $h_n \leq h_{s+1}$. By case analysis, $h_{s+1} \leq \alpha \cdot c(s+1) - c(s) < \beta^2 - \beta$. Thus $\gamma(n) \leq h_n + \alpha \cdot \beta < 2 \cdot \beta^2 - \beta = 2 = c(n)$, a contradiction to α-BB. In detail,

- $d_m \geq 1$: Otherwise we obtain a contradiction to α-BB:
  $$\gamma(m) = m \cdot l_m \leq m \cdot l_j \leq m \cdot \frac{\alpha}{j} = \alpha \cdot \left(1 + \frac{l+1}{j}\right) < \alpha \cdot \left(1 + \frac{\varepsilon}{\alpha}\right) = \beta = c(m) \,.$$

- $d_n = d_{m+1} = d_m + 1$: Otherwise we again obtain a contradiction to α-BB:

$$\gamma(m+1) \le \alpha \cdot \beta + l_{m+1} < \beta^2 + \frac{\alpha}{j} < \beta^2 + \frac{\varepsilon}{l+1} \le \beta^2 + \frac{\varepsilon}{2} \le \beta^2 + \frac{\beta - 1}{2} < 2\,.$$

- Bounds on h_{s+1}: Due to $\alpha \cdot c(s+1) \ge \gamma(s+1) = h_{s+1} + \gamma(s) \ge h_{s+1} + c(s)$, it is $h_{s+1} \le \alpha \cdot c(s+1) - c(s)$ for all $s \in [n-1]$.
  - If $s \in [j-1]$ then

$$h_{s+1} \le \alpha \cdot c(s+1) - c(s) = \alpha - 1 < \beta^2 - \beta\ .$$

  - If $s \in \{j, j+1, \ldots, j+l-1\}$, let $k := s+1-j$. Then

$$\begin{aligned} h_{s+1} &\le \alpha \cdot c(s+1) - c(s) \\ &= \alpha \cdot c(j+k) - c(j+k-1) \\ &= \alpha \cdot \left(\beta - \frac{\beta-1}{\beta^k}\right) - \left(\beta - \frac{\beta-1}{\beta^{k-1}}\right) \\ &< \beta^2 - \frac{\beta-1}{\beta^{k-1}} - \left(\beta - \frac{\beta-1}{\beta^{k-1}}\right) \\ &= \beta^2 - \beta\ . \end{aligned}$$

  - If $s = j + l = m - 1$ then

$$\begin{aligned} h_{s+1} &\le \alpha \cdot c(m) - c(m-1) = \alpha \cdot \beta - \left(\beta - \frac{\beta-1}{\beta^l}\right) \\ &< \alpha \cdot \beta - (\beta - \varepsilon) = \alpha \cdot (\beta - 1) < \beta^2 - \beta\ . \end{aligned}$$

□

5.5.5 Applications To Scheduling Identical Jobs

We like to remind the reader that for (Q|p_i=1|C$_{\max}$) cost-sharing problems, an optimal solution can be computed in polynomial time, e.g., by the LPT algorithm in time $O(n \cdot \log m)$. Theorem 5.18 and Lemma 5.19 give our positive results with respect to approximate budget-balance, and Lemma 5.20 gives the lower bound on the approximation of social cost efficiency.

Theorem 5.18. *For each* (Q|p_i=*1*|C$_{\max}$) *cost-sharing problem there is a 2P-CSF F such that mechanism MechCSF$_F$ is GSP,* $\frac{\sqrt{17}+1}{4}$*-BB for MSP, and computable in* $O(n^2)$.

Proof. As the cost function MSP for identical jobs is symmetric and subadditive, the theorem follows by Theorem 5.14. The evaluation of msp : $[n] \to \mathbb{R}_{\ge 0}$ has no influence on the asymptotic running time, as computing msp(i) for all $i \in [n]$ is accomplished by one run of LPT in time $O(n \cdot \log m)$.

□

Lemma 5.19. *For each* (P|p_i=*1*|C$_{\max}$) *cost-sharing problem there is a 2P-CSF F such that mechanism MechCSF$_F$ is GSP,* 1*-BB, and computable in* $O(n^2)$.

Proof. Consider a (P|p_i=1|C$_{\max}$) cost-sharing problem with n players and m machines. We define a valid 1-BB 2P-CSF $(n, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ that is computable in time $O(n)$. The lemma then follows by Theorem 5.12 and Lemma 5.13.

Consider an arbitrary $k \in [n]$ and let $k = p \cdot m + q$ for $p, q \in \mathbb{N}_0$ and $q < m$.

- If $k \in [m]$, define $h_k := \infty$, $l_k := \frac{1}{k}$, and $d_k := 0$. It holds that $\mathsf{msp}(k) = 1$.
- If $k > m$ and $q = 0$, define $h_k := \infty$, $l_k := \frac{1}{m}$, and $d_k := 0$. It holds that $\mathsf{msp}(k) = p$.
- Otherwise, define $h_k := \mathsf{msp}(k) - (k-1) \cdot \frac{1}{m}$, $l_k := \frac{1}{m}$, and $d_k := 1$. It holds that $\mathsf{msp}(k) = p + 1$. In this case, $h_k = p + 1 - \frac{(k-1)}{m} > \frac{1}{m} = l_k$.

We remark that the same 2P-CSF results by applying Algorithm 5.2 and dividing all cost-shares by $\frac{\sqrt{17}+1}{4}$.

□

Lemma 5.20. *There is a* (P$|p_i{=}1|$C$_{\max}$) *cost-sharing problem for which MechCSF$_F$ for the 2P-CSF F computed by Algorithm 5.2 (or defined in the proof of Lemma 5.19) is no better than $\Omega(n)$-EFF.*

Proof. Consider the (P$|p_i{=}1|$C$_{\max}$) cost-sharing problem with $n = 2m - 1$. The corresponding symmetric costs are $\mathsf{msp}(k) = 1$ for all $k \in [m]$ and $\mathsf{msp}(k) = 2$ for all $k \in \{m+1, \ldots, 2m-1\}$. For the computed 2P-CSF F by Algorithm 5.2 and $\beta := \frac{\sqrt{17}+1}{4}$ it holds that for all $k \in [m]$, $d_k = 0$ and $l_k = \frac{\beta}{k}$, and for all $k \in \{m+1, \ldots, 2m-1\}$, $d_k = 1$, $l_k = \frac{\beta}{k}$ and $h_k = \frac{\beta \cdot 2m - k + 1}{m}$ (in particular, line 4 evaluates to true for all $k \in [m]$. Furthermore, for all $k \in \{m+1, \ldots, 2m-1\}$, lines 4 and 7 evaluate to false and line 9 to true, as $h_k + (k-1) \cdot l_k = 2 \cdot \beta$, and $f = m$ in any case). For $\boldsymbol{v} := (h_{2m-1} - \varepsilon, \ldots, h_{m+1} - \varepsilon, \frac{1}{m} + \varepsilon, \ldots, \frac{1}{m} + \varepsilon)$, $\varepsilon < 2m(\beta - 1)$, and $MechCSF_F = (Q, x)$ it holds that $Q(\boldsymbol{v}) = [m]$ and

$$SC_{\mathsf{MSP}}([m], \boldsymbol{v}) = 1 + \sum_{k=m+1}^{2m-1} \frac{\beta \cdot 2m - k + 1 - \varepsilon}{m} > 1 + \sum_{k=2}^{m} \frac{k}{m} = 1 + \frac{m+1}{2} - \frac{1}{m}$$

However, $SC_{\mathsf{MSP}}([2m-1], \boldsymbol{v}) = 2$. Note that we get the same lower bound for the 2P-CSF from the proof of Lemma 5.19.

□

5.6 Non-Symmetric Costs

In this section we apply our ideas for symmetric costs to settings with non-symmetric costs in two flavors. First, for (Q$||$C$_{\max}$) cost-sharing problems we show that dividing the players in d sets with equal processing times and applying *MechCSF* separately for every set yields GSP and $(\frac{\sqrt{17}+1}{4} \cdot d)$-BB mechanisms. Second, for (P3$|p_i \in \{1,2\}|$C$_{\max}$) cost-sharing problems, we generalize the notion of a preference-ordered cost-sharing method and re-use the ideas of mechanism *MechCSF* to obtain GSP and even 1-BB.

5.6.1 Applications To The General Scheduling Setting

Lemma 5.21. *For each* (Q$||$C$_{\max}$) *cost-sharing problem there is a mechanism that is GSP and $(\frac{\sqrt{17}+1}{4} \cdot d)$-BB for C_{LPT}, where d is the number of different processing times. Furthermore, it is computable in $O(n^2)$.*

Proof. Fix $\boldsymbol{p} \in \mathbb{N}^n$ and $\boldsymbol{s} \in \mathbb{N}^m$ and consider the cost-sharing problem $(\boldsymbol{p}, \boldsymbol{s})$. For each set $S \subseteq [n]$, we define $\mathcal{P}(S) := \{p_i \mid i \in S\}$ as the set of different processing times of the jobs in S and let $\mathcal{P}(S) := \{t_1, \ldots, t_{|\mathcal{P}(S)|}\}$. Define $N := [n]$ and $N^t := \{i \in N \mid p_i = t\}$ for all $t \in \mathcal{P}(N)$.

For each input vector $\boldsymbol{b} \in \mathbb{R}^n$, the mechanism $M = (Q, x)$ works as follows:

- For each $t \in \mathcal{P}(N)$:
  - Compute the 2P-CSF $(|N^t|, \boldsymbol{h}, \boldsymbol{l}, \boldsymbol{d})$ from Algorithm 5.2 for costs induced by $|N^t|$ *identical* jobs, i.e., for cost function msp.
  - Define a 2P-CSF $F^t := (|N^t|, t \cdot \boldsymbol{h}, t \cdot \boldsymbol{l}, \boldsymbol{d})$ by multiplying the cost shares by t.
  - Use $\mathit{MechCSF}_{F^t}$ with input $\boldsymbol{b}_{N^t}$ to decide on the set of served players $Q^t \subseteq N^t$.
- Let $\xi^t(Q^t)$ be the cost-shares induced by F^t and set $Q(\boldsymbol{b}) := \cup_{t \in \mathcal{P}(N)} Q^t$ and $x_i(\boldsymbol{b}) := \xi_i^{p_i}(Q^{p_i})$ for all $i \in Q(\boldsymbol{b})$ and $x_i(\boldsymbol{b}) := 0$ otherwise.

This mechanism is clearly GSP. It is $(\frac{\sqrt{17}+1}{4} \cdot d)$-BB for C_{LPT} since with

$$\sum_{i \in Q(\boldsymbol{b})} x_i(\boldsymbol{b}) = \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} \sum_{i \in Q^t} \xi_i^t(Q^t)$$

it holds that

$$\begin{aligned}
\sum_{i \in Q(\boldsymbol{b})} x_i(\boldsymbol{b}) &\leq \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} t \cdot \frac{\sqrt{17}+1}{4} \cdot \mathsf{msp}(|Q^t|) \\
&= \frac{\sqrt{17}+1}{4} \cdot \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} \mathsf{MSP}(Q^t) \\
&\leq \frac{\sqrt{17}+1}{4} \cdot |\mathcal{P}(Q(\boldsymbol{b}))| \cdot \mathsf{MSP}(Q(\boldsymbol{b})) \ ,
\end{aligned}$$

and moreover,

$$\begin{aligned}
\sum_{i \in Q(\boldsymbol{b})} x_i(\boldsymbol{b}) &\geq \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} t \cdot \mathsf{msp}(|Q^t|) \\
&= \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} \mathsf{MSP}(Q^t) \\
&= \sum_{t \in \mathcal{P}(Q(\boldsymbol{b}))} C_{\mathsf{LPT}}(Q^t) \\
&\geq C_{\mathsf{LPT}}(Q(\boldsymbol{b})) \ .
\end{aligned}$$

The last inequality is due to Lemma 3.4.

Computing $\mathsf{msp}(i)$ for all $i \in \mathbb{N}$ takes time $O(n \cdot \log m)$ by one run of LPT for n identical jobs. For each $t \in \mathcal{P}(N)$, the computation of F^t and each run of $\mathit{MechCSF}_{F^t}$ with $|N^t|$ players takes $O(|N^t|^2)$ time. Since $\sum_{t \in \mathcal{P}(N)} |N^t|^2 \leq n^2$, this algorithm is computable in $O(n^2)$.

□

5.6.2 Applications To Scheduling On Identical Machines

We consider $(\mathrm{P3}|p_i \in \{1,2\}|\mathrm{C_{max}})$ cost-sharing problems. Here, we assume that $p_1 \leq \ldots, \leq p_n$. We adjust the definition of a preference ordered cost-sharing method:

Definition 5.22. *A cost-sharing method* $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ is 2-type preference ordered*, if for every* $(s_1, s_2) \in [n]_0^2$ *with* $s_1 + s_2 \leq n$*, there is a vector* $\boldsymbol{\xi}^{(s_1,s_2)} \in \mathbb{R}^{(s_1+s_2)}_{\geq 0}$ *such that for all players* $i \in [n]$ *and all sets of players* $S \subseteq [n]$ *with* $s_1 := |\{i \in S \mid p_i = 1\}|$ *and* $s_2 := |\{i \in S \mid p_i = 2\}|$ *we have*

$$\xi_i(S) := \begin{cases} \xi^{(s_1,s_2)}_{\mathsf{rank}(i,S)} & \text{if } i \in S \\ 0 & \text{otherwise.} \end{cases}$$

Example 5.23. Let $n = 6$*,* $\xi^{1,2} = (2,3,3)$ *and consider* $S = \{1,4,5\}$ *with* $p_1 = 1$ *and* $p_4 = p_5 = 2$*. Then* $\boldsymbol{\xi}(S) = (2,0,0,3,3,0)$*.*

Theorem 5.24. *For each* $(\mathrm{P3}|p_i \in \{1,2\}|\mathrm{C_{max}})$ *cost-sharing problem, there is a mechanism that is GSP and* 1*-BB and runs in time* $O(n)$*.*

Proof. We first define a 2-type preference-ordered cost-sharing method according to a matrix $\mathcal{X}$. Subsequently, we define mechanism $M_{\mathcal{X}}$.

We first define vectors $\boldsymbol{\xi}^{s_1,s_2}$ for all $s_1, s_2 \in [n]_0$ with $s_1 + s_2 \leq n$ that induce a 2-type preference-ordered cost-sharing method. There are fixed *low cost shares* of $\frac{1}{3}$ for players with $p_i = 1$ and of $\frac{2}{3}$ for players with $p_i = 2$. However, some least preferred players have to pay *higher cost shares* which we define with help of matrix $\mathcal{X}$ (here given for $n = 9$, but intuitively extendable to arbitrary n):

$$\mathcal{X} := \begin{pmatrix}
* & (-,2) & (-,\frac{4}{3}) & * & (-,2) & (-,\frac{4}{3}) & * & (-,2) & (-,\frac{4}{3}) & * \\
(1,-) & (1,1) & (\frac{2}{3},-) & (1,-) & (1,1) & (\frac{2}{3},-) & (1,-) & (1,1) & (\frac{2}{3},-) & (1,-) \\
(\frac{2}{3},-) & (\frac{2}{3},1) & * & (\frac{2}{3},-) & (\frac{2}{3},1) & * & (\frac{2}{3},-) & (\frac{2}{3},1) & * & (\frac{2}{3},-) \\
* & (-,1) & (-,11) & * & (-,1) & (-,11) & * & (-,1) & (-,11) & * \\
(1,-) & * & (-,1) & (-,11) & * & (-,1) & (-,11) & * & (-,1) & (-,11) \\
(\frac{2}{3},-) & (1,-) & * & (-,1) & (-,11) & * & (-,1) & (-,11) & * & (-,1) \\
* & (\frac{2}{3},-) & (1,-) & * & (-,1) & (-,11) & * & (-,1) & (-,11) & * \\
(1,-) & * & (\frac{2}{3},-) & (1,-) & * & (-,1) & (-,11) & * & (-,1) & (-,11) \\
(\frac{2}{3},-) & (1,-) & * & (\frac{2}{3},-) & (1,-) & * & (-,1) & (-,11) & * & (-,1) \\
* & (\frac{2}{3},-) & (1,-) & * & (\frac{2}{3},-) & (1,-) & * & (-,1) & (-,11) & *
\end{pmatrix}$$

Note that we start indexing at 0. Each entry $\mathcal{X}(s_1, s_2) = (\mathcal{X}_1(s_1, s_2), \mathcal{X}_2(s_1, s_2))$ consists of two vectors: the vector $\mathcal{X}_1(s_1, s_2)$ of higher cost shares for players with $p_i = 1$ and the vector $\mathcal{X}_2(s_1, s_2)$ of higher cost shares for players with $p_i = 2$. All vectors except $(1,1)$ (which we abbreviate with '11') consist of only one cost share. To increase readability, we write '$-$' for the empty vector and '$*$' for $(-,-)$. The resulting vector of cost-shares is defined as

$$\boldsymbol{\xi}^{(s_1,s_2)} := \left(\underbrace{\mathcal{X}_1(s_1,s_2), \frac{1}{3}, \ldots, \frac{1}{3}}_{s_1}, \underbrace{\mathcal{X}_2(s_1,s_2), \frac{2}{3}, \ldots \frac{2}{3}}_{s_2} \right)$$

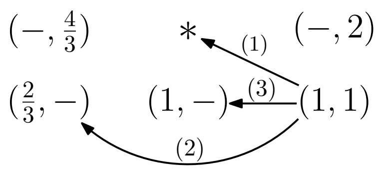
Example 5.25. $\boldsymbol{\xi}^{(2,4)} = (\frac{2}{3}, \frac{1}{3}, 1, \frac{2}{3}, \frac{2}{3}, \frac{2}{3})$, $\boldsymbol{\xi}^{(2,2)} = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$, and $\xi^{(0,2)} = (\frac{4}{3}, \frac{2}{3})$.

Algorithm 5.3 (computing $M_{\mathcal{X}}(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$ for matrix $\mathcal{X}$).

1: $A_j := \{i \in [n] \mid p_i = j \text{ and } b_i \geq \frac{j}{3}\}\ \forall j \in [2]$ ▷ (potential) Advantaged players
2: **if** $|A_1| = 1$ and $b_{\min A_1} < \frac{2}{3}$ **then** $A_1 := \emptyset$
3: $I_j := |\{i \in A_j \mid b_i = \frac{j}{3}\}|\ \forall j \in [2]$ ▷ (alleged) Indifferent players
4: $D_j := \emptyset\ \forall j \in [2]$ ▷ Disadvantaged players
5: **loop**
6: $(\rho_1, \rho_2) :=$ lex. smallest element in $\Big\{\{(t_1, t_2) \in [|I_1|]_0 \times [|I_2|]_0$ with
7: $\big|\mathcal{X}_j\big(|D_1| + |A_1| - t_1, |D_2| + |A_2| - t_2\big)\big| = |D_j|\ \forall j \in [2]\} \cup (n, n)\Big\}$
8: **if** $(\rho_1, \rho_2) = (n, n)$ and **loop** is executed for the first time **then**
9: **if** $\mathcal{X}(|A_1|, |A_2|) = (1, -)$ and $|I_2| \geq 1$ **then** $(\rho_1, \rho_2) := (0, 1)$
10: **if** $\mathcal{X}(|A_1|, |A_2|) = (1, 1)$ and $|I_2| \geq 2$ **then** $(\rho_1, \rho_2) := (0, 2)$
11: **if** $(\rho_1, \rho_2) \neq (n, n)$ **then** ▷ Sufficient indifferent players?
12: $A_j := A_j \setminus \{\rho_j \text{ largest elements of } I_j\}\ \forall j \in [2]$
13: **break**
14: $S := A_1 \cup D_1 \cup A_2 \cup D_2$; $s_j := |A_j| + |D_j|\ \forall j \in [2]$
15: Let $\ell \in A_1 \cup A_2$ be minimum with $\xi^{(s_1, s_2)}_{\mathsf{rank}(\ell, S)} > \frac{p_\ell}{3}$ ▷ Least preferred
16: **if** $b_\ell \geq \xi^{(s_1, s_2)}_{\mathsf{rank}(\ell, S)}$ **then** $D_{p_\ell} := D_{p_\ell} \cup \{\ell\}$ ▷ Make disadvantaged
17: $A_{p_\ell} := A_{p_\ell} \setminus \{\ell\}$; $I_{p_\ell} := I_{p_\ell} \setminus \{\ell\}$
18: $\boldsymbol{x} := \xi(A_1 \cup D_1 \cup A_2 \cup D_2)$, where ξ is the cost-sharing method induced by $\mathcal{X}$
19: **return** $(A_1 \cup D_1 \cup A_2 \cup D_2, \boldsymbol{x})$

Initially, players with insufficient bids are rejected. Then, we try to remove indifferent players such that all remaining players in $A_1 \cup A_2$ pay their minimum cost-share. In case of success, $(\rho_1, \rho_2) \neq (n, n)$ in line 11. Otherwise, the first player not in $D_1 \cup D_2$ who has to pay more than his minimum cost-share is accepted (for a cost-share that stays fixed during the further execution) or rejected, according to his bid. In lines 9 and 10, special cases are considered in which a player with processing time 1 improves his cost-share from 1 to $\frac{2}{3}$ (and not to the minimum cost-share). This check is only required in the first iteration; otherwise, the existence of sufficient indifferent players is already precluded by $(\rho_1, \rho_2) = (n, n)$ after line 6.

Fig. 5.2. Sketch for Example 5.26



Example 5.26. Let $n = 6$, $\boldsymbol{p} := (1, 1, 2, 2, 2, 2)$, and consider $\boldsymbol{b} := (\frac{1}{3} - \varepsilon, 1, \frac{2}{3}, 2, 2, 2)$. Initially, $A_1 = \{2\}$, $A_2 = \{3, 4, 5, 6\}$, $I_1 = \emptyset$, $I_2 = \{3\}$, and $D_1 = D_2 = \emptyset$. It is $\mathcal{X}(|A_1| + |D_1|, |A_2| + |D_2|) = (1, 1)$, and to 'move' to the next $(*)$ entry (confer arrow (1) in Figure 5.2), we need that $|I_1| \geq 1$ and $|I_2| \geq 1$. As this is not the case, $(\rho_1, \rho_2) = (n, n)$ in line 8. Even after line 10, $(\rho_1, \rho_2) = (n, n)$, as also entry $(\frac{2}{3}, -)$ cannot be reached due to $|I_2| \ngeq 2$ (confer arrow (2) in Figure 5.2). Thus, in

line 15, $\ell = 2$. As $b_2 \geq 1$, $D_1 = \{2\}$ and 2 is removed from A_1. In the next iteration, again $\mathcal{X}(|A_1| + |D_1|, |A_2| + |D_2|) = (1,1)$, and to 'move' to entry $(1,-)$, $|I_2| \geq 1$ is required (confer arrow (3) in Figure 5.2). As this is possible, $(\rho_1, \rho_2) = (0,1)$ in line 6, player 3 is removed from A_2 in line 12, and the set $\{2,4,5,6\}$ is served for cost-shares $\boldsymbol{\xi}^{(1,3)} = (1, \frac{2}{3}, \frac{2}{3}, \frac{2}{3})$.

Budget-Balance

1-BB follows by direct computation, e.g., for $S \subseteq [n]$ with $(s_1, s_2) = (2,4)$, it holds that $\mathsf{MSP}(S) = 4$. For $S \subseteq [n]$ with $(s_1, s_2) = (2,2)$ or $(s_1, s_2) = (0,2)$, $\mathsf{MSP}(S) = 2$. Example 5.25 shows that these costs are exactly recovered. The straightforward verification for the remaining cases is left to the reader.

Transitions

Showing GSP is rather intricate as there are many case differentiations. We define *transitions* that are helpful for finally proving GSP. We look at all possible runs of $M_{\mathcal{X}}$ that consider line 15 at least 2 times, i.e., there are at least two disadvantaged players who are either rejected or receive the service for a high cost share. By definition of $\mathcal{X}$, line 15 is executed at most three times. This can be observed by the '$*$' entries in $\mathcal{X}$ that act as a kind of 'barriers'.

For one specific run for bid vector $\boldsymbol{b} \in \mathbb{R}^n$ in which line 15 is executed $t \in \{2,3\}$ times, we define a *transition* to be the sequence of the t entries $\mathcal{X}(s_1, s_2)$, where s_1, s_2 are defined in line 14. We let ℓ_k be the player considered during the k-th call of line 15. All possible transitions are given in Table 5.1. We write $+$, if ℓ_k is served (i.e. accepted in line 16), and $-$ otherwise ($+/-$ if both cases may occur).

Table 5.1. All possible transitions of length 2 and 3

transition	ℓ_1	ℓ_2	ℓ_3	p_{ℓ_1}	p_{ℓ_2}	p_{ℓ_3}
$(-, \frac{4}{3}) \to (-, 2)$	$-$	$+/-$		2	2	
$(1,1) \to (1,1)$	$+$	$+/-$		1	2	
$(1,1) \to (-,2)$	$-$	$+/-$		1	2	
$(\frac{2}{3}, -) \to (1,-)$	$-$	$+/-$		1	1	
$(\frac{2}{3}, 1) \to (\frac{2}{3}, 1)$	$+$	$+/-$		1	2	
$(-, 11) \to (-, 11)$	$+$	$+/-$		2	2	
$(-, 11) \to (-, 1)$	$-$	$+/-$		2	2	
$(\frac{2}{3}, 1) \to (1,1) \to (1,1)$	$-$	$+$	$+/-$	1	1	2
$(\frac{2}{3}, 1) \to (1,1) \to (-,2)$	$-$	$-$	$+/-$	1	1	2

Proof of GSP

Assume that there is a $(\mathrm{P3}|p_i \in \{1,2\}|\mathrm{C}_{\max})$ cost-sharing problem $(\boldsymbol{p}, \mathbf{1})$ for which $M_{\mathcal{X}} = (Q, x)$ is not GSP. Then, there exists a coalition $K \subseteq [n]$ with true valuations $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$ and a bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ for all $i \in K$, with at least one strict inequality. Without loss of generality, let $\boldsymbol{v} := (\boldsymbol{v}_K, \boldsymbol{b}_{-K})$.

It can easily be verified that the mechanism is SP. This excludes the case $|K| = 1$. Define A_1, A_2 and I_1, I_2 to be the initial sets computed in lines 1 and 3 for input $\boldsymbol{v}$. If one of the special cases from lines 9 and 10 occurs, then $M_{\mathcal{X}}$ terminates. It follows that $\min A_1$ has to be the player in K who strictly improved his utility. However, then there is at least one player $i \in Q(\boldsymbol{v})$ with $v_i > \frac{p_i}{3} = x_i(\boldsymbol{v}) \geq b_i$ such that $i \notin Q(\boldsymbol{b})$ and $0 = u_i(\boldsymbol{b}, v_i) < u_i(\boldsymbol{v}, v_i)$; a contradiction. Furthermore, observe that line 15 is executed at least once, as otherwise all players i with $v_i > \frac{p_i}{3}$ receive the service and pay $\frac{p_i}{3}$, hence no player may strictly improve his utility for $\boldsymbol{b}$. Let t be the number of calls of line 15 for $M_{\mathcal{X}}$ with input $\boldsymbol{v}$ and let $\mathcal{L} := \{\ell_1, \ldots, \ell_t\}$ be the disadvantaged players, where ℓ_k is considered for the k-th call.

Without loss of generality, we may assume that $K \subseteq \mathcal{L}$: For input $\boldsymbol{v}$, players $i \notin \{\ell_1, \ldots, \ell_t\}$ either receive the service for $\frac{p_i}{3}$ or are rejected due to $v_i = \frac{p_i}{3}$. These players can neither strictly improve their utility for another outcome, nor change the outcome for $\boldsymbol{v}$ by bidding $b_i \neq v_i$ without strictly decreasing their utility.

As $K \subseteq \mathcal{L}$ and $|K| > 1$, $t \in \{2, 3\}$. Let $\ell_j \in K$ with $u_{\ell_j}(\boldsymbol{b}, v_{\ell_j}) > u_{\ell_j}(\boldsymbol{v}, v_{\ell_j})$. In particular, $v_{\ell_j} > \frac{p_{\ell_j}}{3}$. We derive contradictions for all possible transitions for $\boldsymbol{v}$.

We start with the transitions of length 2. Here, if $j = 1$, we may assume that $v_{\ell_2} > \frac{p_{\ell_2}}{3}$ and $b_{\ell_2} = -1$, as the only way for ℓ_2 to be beneficial for ℓ_1 is to bid such as not to receive the service beforehand, and players with $v_i = \frac{p_i}{3}$ are considered to be rejected anyway. On the other hand, if $j \neq 1$ and ℓ_1 does receive the service for $\boldsymbol{v}$, ℓ_1 may only change the outcome for ℓ_2 bidding such as not to receive the service; we may then assume that $b_{\ell_1} = -1$. We do not have to consider the case that $j \neq 1$ and ℓ_1 does not receive the service for $\boldsymbol{v}$, as player ℓ_1 may only change the outcome for ℓ_2 by bidding such as to receive the service, resulting in a negative utility. Thus, for all transitions where the entry for ℓ_1 is '–', it follows from $j \neq 1$ that $j \neq 2$.

- $(-, \frac{4}{3}) \to (-, 2)$:
  In particular, $|I_2 \setminus \{\ell_1\}| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (-, 2)$, it holds that $|I_2 \setminus \{\ell_1, \ell_2\}| \geq 1$, a contradiction.
- $(1, 1) \to (1, 1)$:
  In particular, $|I_2| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (1, -)$, it holds that $|I_2 \setminus \{\ell_2\}| \geq 1$, a contradiction. Thus $j = 2$, and due to $\mathcal{X}(|A_1| - 1, |A_2|) = (-, 2)$, it is $|I_2 \setminus \{\ell_2\}| \geq 1$, a contradiction.
- $(1, 1) \to (-, 2)$:
  In particular, $|I_2| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (1, -)$, it holds that $|I_2 \setminus \{\ell_2\}| \geq 1$, a contradiction.
- $(\frac{2}{3}, -) \to (1, -)$:
  In particular, $|I_2| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1| - 1, |A_2|) = (1, -)$, it is $|I_2| \geq 1$, a contradiction.
- $(\frac{2}{3}, 1) \to (\frac{2}{3}, 1)$:
  In particular, $|I_2| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (\frac{2}{3}, -)$, it holds that $|I_2 \setminus \{\ell_2\}| \geq 1$, a contradiction. Thus $j = 2$, and due to $\mathcal{X}(|A_1| - 1, |A_2|) = (1, 1)$, it is $|I_2 \setminus \{\ell_2\}| \geq 1$ (there are four cases here: either $(\rho_1, \rho_2) = (1, 1)$ or $(\rho_1, \rho_2) = (0, 2)$ in the first iteration for $\boldsymbol{b}$, or otherwise, the least preferred player with $p_i = 1$ is considered in line 15, thus accepted for a price of 1 or rejected. In any case, $(\rho_1, \rho_2) = (0, 1)$ in the next iteration, as ℓ_2 strictly improves), a contradiction.
- $(-, 11) \to (-, 11)$:
  It is $|I_1| < 1$ and $|I_2| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (-, 1)$, it holds that $|I_1| \geq 2$ or $|I_2 \setminus \{\ell_1, \ell_2\}| \geq 1$ if $|A_1| > 3$, and $|I_2 \setminus \{\ell_1, \ell_2\}| \geq 1$ if $|A_1| = 3$, a contradiction. Thus $j = 2$, where we get the same contradiction.

- $(-, 11) \to (-, 1)$:
  In particular, $|I_1| < 1$ and $|I_2 \backslash \{\ell_1\}| < 1$. If $j = 1$, due to $\mathcal{X}(|A_1|, |A_2|-1) = (-, 1)$, $|I_1| \geq 2$ or $|I_2 \setminus \{\ell_1, \ell_2\}| \geq 1$ if $|A_1| > 3$, and $|I_2 \setminus \{\ell_1, \ell_2\}| \geq 1$ if $|A_1| = 3$, a contradiction.

For the two transitions with $t = 3$, if $j = 1$, we may assume that either $v_{\ell_2} > \frac{p_{\ell_2}}{3}$ and $b_{\ell_2} = -1$, or $v_{\ell_3} > \frac{p_{\ell_3}}{3}$ and $b_{\ell_3} = -1$, or both. Having established that $j \neq 1$, as 1 does not receive the service for any of the two transitions, ℓ_1 is not capable of helping ℓ_2 or ℓ_3 without obtaining negative utility. Thus, in case $j = 2$, we only have to consider that ℓ_3 drops out, i.e., $v_{\ell_3} > \frac{p_{\ell_3}}{3}$ and $b_{\ell_3} = -1$. With the same arguments, for $j = 3$ we only have to consider the transition $(\frac{2}{3}, 1) \to (1, 1) \to (1, 1)$ where ℓ_2 bids -1.

- $(\frac{2}{3}, 1) \to (1, 1) \to (1, 1)$:
  In particular, $|I_2| < 1$. If $j = 1$ and only ℓ_2 bids -1, due to $\mathcal{X}(|A_1| - 1, |A_2|) = (1, 1)$, $|I_2| \geq 2$, a contradiction. If $j = 1$ and only ℓ_3 bids -1, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (\frac{2}{3}, -)$, $|I_2 \setminus \{\ell_3\}| \geq 1$, a contradiction. If $j = 1$ and ℓ_2 and ℓ_3 submit a negative bid, due to $\mathcal{X}(|A_1| - 1, |A_2| - 1) = (1, -)$ it holds that $|I_2 \setminus \{\ell_3\}| \geq 1$, a contradiction. If $j = 2$, due to $\mathcal{X}(|A_1|, |A_2| - 1) = (\frac{2}{3}, -)$, the fact that 1 does not receive the service and therefore $\mathcal{X}(|A_1| - 1, |A_2| - 1) = (1, -)$, it holds that $|I_2 \setminus \{\ell_3\}| \geq 1$, a contradiction. If $j = 3$, due to $\mathcal{X}(|A_1| - 1, |A_2|) = (1, 1)$, the fact that ℓ_1 does not receive the service and therefore $\mathcal{X}(|A_1| - 2, |A_2|) = (-, 2)$, it holds that $|I_2 \setminus \{\ell_3\}| \geq 1$, a contraction.
- $(\frac{2}{3}, 1) \to (1, 1) \to (-, 2)$:
  By the same arguments as for the previous transition, $j \notin \{1, 2\}$.

Running Time

For the running time, consider Algorithm 5.1. Given $\mathcal{X}$, every operation outside the loop takes time at most $O(n)$. The loop is executed at most 3 times and each operation in the loop takes at most time $O(n)$. An overall running time of $O(n)$ results.

□

5.7 Conclusion and Open Problems

We regard as the main asset of our work presented in this chapter that it is a systematic first step for finding GSP mechanisms that perform better than Moulin mechanisms. While symmetric costs are arguably of limited practical interest, we yet transferred our techniques to the minimum makespan scheduling problem as an application and also to a setting with non-symmetric costs. Clearly, there are many open issues:

- For symmetric and/or subadditive costs, we still need an exact characterization with respect to the *best* possible approximate budget-balance that GSP mechanisms can achieve.
- For non-symmetric costs, we would like to answer if there are promising generalizations of our techniques.
- Finally, a central question is if we can adjust our cost-sharing forms in some way such that the better budget-balance (compared to Moulin mechanisms) does not come at the price of increased social cost.

6

Egalitarian Mechanisms

6.1 Contribution

We introduce the new behavioral assumption that coalitions do *not* form if some member would lose service. Yet, coalitions *do already* form if at least one player wins the service. Being reminded of collectors, we call resistance against collective collusion in the new sense *group-strategyproof against collectors* (CGSP). We further introduce *weak* CGSP (WCGSP).

- We show that CGSP is strictly stronger than WGSP but incomparable to GSP. Moreover, we prove that – contrary to WGSP – any WCGSP mechanism induces unique cost-shares. This strictly improves on the result by Moulin [97] establishing unique cost-shares for any GSP mechanism. Additionally, Moulin's result is based on strict CS while we solely require CS.
- We give an algorithm for computing CGSP mechanisms that we call 'egalitarian' due to being inspired by Dutta and Ray's [34] 'egalitarian solutions'.
- We show that our egalitarian mechanisms are CGSP by identifying them as a subclass of acyclic mechanisms and by showing that all acyclic mechanisms are CGSP (and thus remarkably stronger than WGSP). In addition, our mechanisms are 1-BB for arbitrary costs and additionally $2H_n$-EFF for the very natural class of subadditive costs. As main tools, we use *most cost-efficient set selection* and *price functions*.
- We give a (rather unrealistic) cost function for which approximation of social cost efficiency by egalitarian mechanisms is infeasible. We nevertheless identify a property of price functions that helps to establish efficiency guarantees for more reasonable cost functions.
- We present a framework for coping with the computational complexity of egalitarian mechanisms, especially for problems in which players are endowed with some size (e.g., processing times). Besides the use of approximation algorithms, the key idea here are 'monotonic' cost functions that must not increase when replacing a player by another one with a smaller size.
- We give applications for scheduling and bin-packing that underline the power of our new approach. For $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems and their subproblems, our results are given in Table 1.1. Notably, our framework also allows for first results on cost-sharing problems for more realistic scheduling models.

All result presented in this chapter are published in [11].

6.2 Organization

In Section 6.3, we give the formal definitions of CGSP and WCGSP as well as their relation to other notions of incentive-compatibility, and show that WCGSP mechanisms induce unique cost-shares. We introduce egalitarian mechanisms together with set selection and price functions in Section 6.4. Section 6.5 proves CGSP of egalitarian mechanisms via acyclic mechanisms, and Section 6.6 gives our results on efficiency approximation. We present our computational framework in Section 6.7, and its applications in Section 6.8.

6.3 Collectors' Behavior

6.3.1 New Behavioral Assumptions

In the demand for GSP lies an implicit modeling assumption that is common to most recent works on cost-sharing mechanisms: First, a player is only willing to be untruthful and join a coalition of false-bidders if this does *not* involve sacrificing his own utility. Second, a coalition always requires an initiating player whose utility *strictly* increases.

Clearly, there are other reasonable behavioral assumptions on coalition formation. We introduce and study the following: First, besides not giving up utility, a player would not sacrifice service, either. (Although his utility is zero both when being served for his valuation and when not being served.) Second, it is sufficient for coalition formation if the initiating player gains *either* utility *or* service. While we consider this behavior very human, it especially reminds us of collectors. We hence denote a mechanism's resistance against coalitions in this new sense as *group-strategyproof against collectors.*

Definition 6.1. *A mechanism $M = (Q, x)$ is* group-strategyproof against collectors *(CGSP) if for every coalition $K \subseteq [n]$ and every true valuation vector $\boldsymbol{v}_K = (v_i)_{i\in K} \in \mathbb{R}^{|K|}$ there is no bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that*

- $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ *and* $i \notin Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K}) \setminus Q(\boldsymbol{b})$ *for all* $i \in K$ *and*
- $u_i(\boldsymbol{b}, v_i) > u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ *or* $i \in Q(\boldsymbol{b}) \setminus Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$ *for at least one* $i \in K$.

We remark that CGSP in a model with quasi-linear utilities is equivalent to GSP in a changed model where a preference of being served for the price of valuation over not being served is internalized in the utilities. To illustrate the interrelation between CGSP and GSP, we introduce a property which is a relaxation of both, called *weak* CGSP:

Definition 6.2. *A mechanism $M = (Q, x)$ is* weakly *CGSP (WCGSP) if for every coalition $K \subseteq [n]$ and every true valuation vector $\boldsymbol{v}_K = (v_i)_{i\in K} \in \mathbb{R}^{|K|}$ there is no bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that*

- $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ *and* $i \notin Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K}) \setminus Q(\boldsymbol{b})$ *for all* $i \in K$ *and*
- $u_i(\boldsymbol{b}, v_i) > u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ *for at least one* $i \in K$.

Lemma 6.3. *The following implications hold. They do generally not hold in the opposite directions.*

$$
\begin{tikzcd}[column sep=small]
\mathit{GSP} \arrow[dr, Rightarrow] & & & \\
& \mathit{WCGSP} \arrow[r, Rightarrow] & \mathit{WGSP} \arrow[r, Rightarrow] & \mathit{SP} \\
\mathit{CGSP} \arrow[ur, Rightarrow] & & &
\end{tikzcd}
$$

Proof. The implications GSP $\Rightarrow$ WCGSP $\Rightarrow$ WGSP $\Rightarrow$ SP and CGSP $\Rightarrow$ WCGSP hold by definition. In the following, we show that the converse directions do not hold in general.

We have GSP $\not\Leftarrow$ WCGSP because on the one hand, we show in Theorem 6.14 that acyclic mechanisms are CGSP (and thus WCGSP), and on the other hand, acyclic mechanisms are not GSP in general [94]. To show CGSP $\not\Leftarrow$ WCGSP, we give a mechanism that is GSP (and thus WCGSP) but not necessarily CGSP. Consider a modified version of Moulin_ξ for a cross-monotonic cost-sharing method ξ (confer Algorithm 4.1). After the execution of Moulin_ξ, delete players who receive the service for a cost share equal to their valuation, if this does not change the cost shares of the remaining service-receiving players. This mechanism is still GSP. However, a player deleted in the additional step may submit a bid strictly larger than his valuation in order to receive the service. His utility in both outcomes is zero. This contradicts CGSP.

We prove that WCGSP $\not\Leftarrow$ WGSP by considering the trivial mechanisms Triv_C proposed by Immorlica et al. [73] for any cost function C, and adapted to negative bids in Algorithm 6.1. We show that Triv_C is WGSP but not necessarily WCGSP.

Algorithm 6.1 (computing $\mathit{Triv}_C(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

$\boldsymbol{x} := \boldsymbol{0}$; $Q := \{i \in [n] \mid b_i \geq 0\}$;
while $Q \neq \emptyset$ and $b_{\min Q} < C(Q)$ **do**
    $Q := Q \setminus \{\min Q\}$;
if $Q \neq \emptyset$ **then**
    $x_{\min Q} := C(Q)$
return $(Q, \boldsymbol{x})$

To see that Triv_C is WGSP, assume that it is not. Then, there exists a coalition $K \subseteq [n]$ with true valuations $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$ and a bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$ for all $i \in K$. Observe that for all $i \in K$, it has to hold that $i \in Q(\boldsymbol{b})$ and $v_i > 0$. Specifically, $b_i \geq 0$. Furthermore, there has to be a $j \in K$ with $j \notin Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$, as otherwise the outcomes for $\boldsymbol{b}$ and $(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$ are the same. Consider the smallest such $j \in K$ and assume that j is deleted from set Q for input $(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$, i.e., $v_j < C(Q)$. When running Triv_C on input $\boldsymbol{b}$, the computation is exactly the same until j is considered (to pay for the same set Q). As $j \in Q(\boldsymbol{b})$, it is $b_j \geq C(Q) = x_j(\boldsymbol{b})$ and $Q = Q(\boldsymbol{b})$, thus $u_j(\boldsymbol{b}, v_j) = v_j - C(Q(\boldsymbol{b})) < 0 = u_j((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_j)$, contradicting $j \in K$.

To see that Triv_C is not WCGSP, consider an example with two players and costs $C(\{1,2\}) = 2$ and $C(1) = C(2) = 1$. Assume that the true valuations are $\boldsymbol{v} := (2 - \varepsilon, \varepsilon)$, resulting in $Q(\boldsymbol{v}) = \emptyset$. However, for vector $\boldsymbol{b} := (2 - \varepsilon, -1)$, the outcome is $Q(\boldsymbol{b}) = \{1\}$ and $x(\boldsymbol{b}) = (1, 0)$.

Finally, WGSP $\not\Leftarrow$ SP can be observed from Example B.9 on page 117 that shows that the SP marginal cost mechanisms are generally not WGSP. □

We remark that the acyclicity of Moulin_ξ according to [94] and Theorem 6.14 will imply that Moulin_ξ is both GSP and CGSP. Furthermore, already the sequential stand alone mechanisms introduced by Moulin [97] achieve CGSP and 1-BB (Lemma B.4, p. 113). Yet, they are only $\Omega(n)$-EFF in general (Example B.2, p. 112).

6.3.2 Sufficient Conditions for Unique Cost Shares

Interestingly, already WCGSP is sufficient for a mechanism to induce unique cost shares. The proof of Theorem 6.4 uses ideas from an analogous result in [97] (confer Theorem C.1). However, Theorem 6.4 is stronger since GSP and *strict* CS are relaxed to WCGSP and CS. Conversely, WGSP mechanisms do not always induce unique cost shares, even if we demand 1-BB, as stated by Lemma 6.5.

Theorem 6.4. *Let $M = (Q, x)$ be a WCGSP mechanism. Then, for any two $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$, it holds that $x(\boldsymbol{b}) = x(\boldsymbol{b}')$. This result holds even if we restrict our model to non-negative bids and only require CS.*

Proof. We first restrict attention to the more intricate case in which $\boldsymbol{b} \in \mathbb{R}^n_{\geq 0}$ for all bid vectors $\boldsymbol{b}$ and strict CS is not required to hold.

For $\boldsymbol{b} \in \mathbb{R}^n_{\geq 0}$ and $S \subseteq [n]$, we define $\boldsymbol{y}^{\boldsymbol{b},S} \in \mathbb{R}^n_{\geq 0}$ by setting

$$y_i^{\boldsymbol{b},S} := \begin{cases} b_i^+ & \text{for all } i \in S \cap Q(\boldsymbol{b}) \\ 0 & \text{for all } i \in S \setminus Q(\boldsymbol{b}) \\ b_i & \text{for all } i \notin S \end{cases}$$

The main part of the proof consists of showing (6.1) and (6.2) for all $\boldsymbol{b} \in \mathbb{R}^n_{\geq 0}$, all $S \subseteq [n]$, all $i \in [n]$, and $T := S \setminus \{j \notin Q(\boldsymbol{b}) \mid b_j = 0\}$:

$$i \in T \Longrightarrow i \in Q(\boldsymbol{b}) \Leftrightarrow i \in Q(\boldsymbol{y}^{\boldsymbol{b},S}) \text{ and } x_i(\boldsymbol{b}) = x_i(\boldsymbol{y}^{\boldsymbol{b},S}) \tag{6.1}$$

$$i \notin T \Longrightarrow u_i(\boldsymbol{b}, b_i) = u_i(\boldsymbol{y}^{\boldsymbol{b},S}, b_i). \tag{6.2}$$

Assume for now that (6.1) and (6.2) hold and consider two bid vectors $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n_{\geq 0}$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$. We have to show that $x(\boldsymbol{b}) = x(\boldsymbol{b}')$. Observe that $x(\boldsymbol{b}) = x(\boldsymbol{y}^{\boldsymbol{b},[n]})$ since for the players $j \in [n]$ with $j \notin Q(\boldsymbol{b})$ and $b_j = 0$ it holds that $x_j(\boldsymbol{b}) = 0$ and $0 = u_j(\boldsymbol{b}, b_j) = u_j(\boldsymbol{y}^{\boldsymbol{b},[n]}, b_j) = 0 - x_j(\boldsymbol{y}^{\boldsymbol{b},[n]})$. Analogously, $x(\boldsymbol{b}') = x(\boldsymbol{y}^{\boldsymbol{b}',[n]})$. Finally, we utilize $\boldsymbol{y}^{\boldsymbol{b},[n]} = \boldsymbol{y}^{\boldsymbol{b}',[n]}$ to conclude that $x(\boldsymbol{b}) = x(\boldsymbol{y}^{\boldsymbol{b},[n]}) = x(\boldsymbol{y}^{\boldsymbol{b}',[n]}) = x(\boldsymbol{b}')$.

To complete this proof, fix $\boldsymbol{b} \in \mathbb{R}^n_{\geq 0}$. We omit the superscript $\boldsymbol{b}$ for $\boldsymbol{y}^{\boldsymbol{b},S}$ and simply write $\boldsymbol{y}^S$. Note that by CS and the definition of $\boldsymbol{y}^S$, $i \in S \cap Q(\boldsymbol{b}) \Rightarrow i \in Q(\boldsymbol{y}^S)$ for all $i \in [n]$. We prove (6.1) and (6.2) by induction on the cardinality of S.

- Let $S := \{k\}$.
  Then $\boldsymbol{y}^S = (y_k^S, \boldsymbol{b}_{-k})$. If $k \notin Q(\boldsymbol{b})$ and $b_k = 0$, (6.2) holds for k with $\boldsymbol{b} = \boldsymbol{y}^S$. Otherwise, we have to verify (6.1). If $k \in Q(\boldsymbol{b})$, it follows from $k \in S$ that $k \in Q(\boldsymbol{y}^S)$. Let $k \in Q(\boldsymbol{y}^S)$ and assume that $k \notin Q(\boldsymbol{b})$. From $y_k^S = 0$, it follows that $x_k(\boldsymbol{y}^S) = 0$. However, $b_k > 0$ already contradicts SP, because for $\boldsymbol{b}$ and true valuation b_k, player k may bid y_k^S in order to receive the service for 0. Formally,

  $$u_k((y_k^S, \boldsymbol{b}_{-k}), b_k) = b_k - 0 > 0 = u_k(\boldsymbol{b}, b_k) \ .$$

  If $x_k(\boldsymbol{b}) > x_k(\boldsymbol{y}^S)$, then $k \in Q(\boldsymbol{b})$ and thus $k \in Q(\boldsymbol{y}^S)$. For $\boldsymbol{b}$ and true valuation b_k, k may bid y_k^S in order to pay a strictly lower cost share, already contradicting SP. Formally,

  $$u_k((y_k^S, \boldsymbol{b}_{-k}), b_k) = b_k - x_k(\boldsymbol{y}^S) > b_k - x_k(\boldsymbol{b}) = u_k((b_k, \boldsymbol{b}_{-k}), b_k) \ .$$

  Analogously, if $x_k(\boldsymbol{b}) < x_k(\boldsymbol{y}^{\boldsymbol{b}})$, then $k \in Q(\boldsymbol{y}^S) \cap Q(\boldsymbol{b})$ and for $\boldsymbol{y}^S$ and true valuation y_k^S, k may bid b_k in order to pay a strictly lower cost share, which again contradicts SP.

We continue to verify (6.2) for all players $j \notin S$. If $u_j(\boldsymbol{b}, b_j) > u_j(\boldsymbol{y}^S, b_j)$ for a $j \neq k$, we get a contradiction to WCGSP with coalition $K = \{j, k\}$ by considering true valuations y_j^S $(= b_j)$ and y_k^S. Formally,

$$
\begin{aligned}
&u_k(\boldsymbol{b}, y_k^S) = u_k(\boldsymbol{y}^S, y_k^S) = u_k((\boldsymbol{y}^S_{\{j,k\}}, \boldsymbol{b}_{-\{j,k\}}), y_k^S)\\
\text{and } &u_j(\boldsymbol{b}, y_j^S) = u_j(\boldsymbol{b}, b_j) > u_j(\boldsymbol{y}^S, b_j) = u_j(\boldsymbol{y}^S, y_j^S) = u_j((\boldsymbol{y}^S_{\{j,k\}}, \boldsymbol{b}_{-\{j,k\}}), y_j^S)\\
\text{and } &k, j \notin Q(\boldsymbol{y}^S_{\{j,k\}}, \boldsymbol{b}_{-\{j,k\}}) \setminus Q(\boldsymbol{b}) \ .
\end{aligned}
$$

The last observation holds for k due to (6.1). For j it is due to $j \in Q(\boldsymbol{b})$ as by assumption $u_j(\boldsymbol{b}, b_j) > 0$. Analogously, if $u_j(\boldsymbol{b}, b_j) < u_j(\boldsymbol{y}^S, b_j)$ for a $j \neq k$, we get a contradiction to WCGSP by considering true valuations y_j^S $(= b_j)$ and b_k.

- Induction hypothesis: (6.1) and (6.2) hold for all $S \subsetneq [n]$ with $|S| < m$.
- Let $S \subseteq [n]$ with $|S| = m$.
  Then, $\boldsymbol{y}^S = (\boldsymbol{y}^S_S, \boldsymbol{b}_{-S})$. We first show (6.1). Let $T := S \setminus \{j \notin Q(\boldsymbol{b}) \mid b_j = 0\}$. For all $i \in T$ with $i \in Q(\boldsymbol{b})$ it is $i \in Q(\boldsymbol{y}^S)$ because of $i \in S$. Consider $i \in T$ with $i \in Q(\boldsymbol{y}^S)$. Assume $i \notin Q(\boldsymbol{b})$, thus $b_i > 0$ and $x_i(\boldsymbol{b}) = 0$. Due to $y_i^S = 0$, it is $x_i(\boldsymbol{y}^S) = 0$. Define $\boldsymbol{z} := (b_i, \boldsymbol{y}^S_{-i})$. Observe that $\boldsymbol{y}^S = (y_i^S, \boldsymbol{z}_{-i})$. By induction hypothesis, $u_i(\boldsymbol{z}, b_i) = u_i(\boldsymbol{b}, b_i) = 0$. This already contradicts SP, since for $\boldsymbol{z}$ and true valuation z_i $(= b_i)$, i may bid y_i^S to receive the service for 0. Formally,

$$u_i((y_i^S, \boldsymbol{z}_{-i}), z_i) = u_i(\boldsymbol{y}^S, z_i) = z_i - 0 > 0 = u_i(\boldsymbol{z}, z_i) \ .$$

  – Assume $x_i(\boldsymbol{b}) > x_i(\boldsymbol{y}^S)$ for an $i \in T$:
    In this case, $x_i(\boldsymbol{b}) > 0$, thus $i \in Q(\boldsymbol{b}) \cap Q(\boldsymbol{y}^S)$. Again, let $\boldsymbol{z} := (b_i, \boldsymbol{y}^S_{-i})$. By induction hypothesis, $u_i(\boldsymbol{b}, b_i) = u_i(\boldsymbol{z}, b_i)$. If $i \in Q(\boldsymbol{z})$, we get $x_i(\boldsymbol{z}) = x_i(\boldsymbol{b})$, already contradicting SP, since for bid vector $\boldsymbol{z}$ and true valuation z_i $(= b_i)$, player i may bid y_i^S in order to pay a strictly smaller cost share. Formally,

$$u_i((y_i^S, \boldsymbol{z}_{-i}), z_i) = u_i(\boldsymbol{y}^S, z_i) = z_i - x_i(\boldsymbol{y}^S) > z_i - x_i(\boldsymbol{z}) = u_i(\boldsymbol{z}, z_i) \ .$$

    If $i \notin Q(\boldsymbol{z})$, it follows by $u_i(\boldsymbol{b}, b_i) = u_i(\boldsymbol{z}, b_i) = 0$ that $b_i = x_i(\boldsymbol{b})$, which again contradicts SP, since for $\boldsymbol{z}$ and true valuation z_i $(= b_i)$, player i may bid y_i^S to receive the service for $x_i(\boldsymbol{y}^S) < b_i$. Formally,

$$u_i((y_i^S, \boldsymbol{z}_{-i}), z_i) = u_i(\boldsymbol{y}^S, z_i) = z_i - x_i(\boldsymbol{y}^S) > 0 = u_i(\boldsymbol{z}, z_i) \ .$$

  – Assume $x_i(\boldsymbol{b}) < x_i(\boldsymbol{y}^S)$ for an $i \in T$:
    For all $j \in T$, it is $x_j(\boldsymbol{b}) \leq x_j(\boldsymbol{y}^S)$ and $j \in Q(\boldsymbol{b}) \Leftrightarrow j \in Q(\boldsymbol{y}^S)$. Now for bid vector $\boldsymbol{y}^S$ and true valuations $\{y_j^S\}_{j \in T}$, T can form coalition $K = T$ to strictly improve the utility of at least one of its members, since $(\boldsymbol{b}_T, \boldsymbol{y}^S_{-T}) = \boldsymbol{b}$. This is a contradiction to WCGSP. Formally,

$$
\begin{aligned}
\forall\, j \in T : u_j((\boldsymbol{b}_T, \boldsymbol{y}^S_{-T}), y_j) &= u_j(\boldsymbol{b}, y_j)\\
&= y_j \cdot \mathsf{in}(Q(\boldsymbol{b})) - x_j(\boldsymbol{b})\\
&= y_j \cdot \mathsf{in}(Q(\boldsymbol{y}^S)) - x_j(\boldsymbol{b})\\
&\geq y_j \cdot \mathsf{in}(Q(\boldsymbol{y}^S)) - x_j(\boldsymbol{y}^S)\\
&= u_j(\boldsymbol{y}^S, y_j)\\
\text{and } \exists\, i \in T :\ u_i((\boldsymbol{b}_T, \boldsymbol{y}^S_{-T}), y_i) &> u_i(\boldsymbol{y}^S, y_i)\\
\text{and } \forall\, j \in T :\ j \notin Q(\boldsymbol{y}^S) &\setminus Q(\boldsymbol{b}_T, \boldsymbol{y}^S_{-T}) \ .
\end{aligned}
$$

Therefore, $x_i(\boldsymbol{b}) = x_i(\boldsymbol{y}^S)$ for all $i \in T$.

- Additionally, $u_i(\boldsymbol{b}, b_i) = u_i(\boldsymbol{y}^S, b_i)$ for all $i \notin T$:
If $u_i(\boldsymbol{b}, b_i) > u_i(\boldsymbol{y}^S, b_i)$ for an $i \notin T$, consider vector $\boldsymbol{y}^S$, coalition $K = T \cup \{i\}$ and true valuations $\boldsymbol{y}^S_{T\cup\{i\}}$. Note that $y_i^S = b_i$ and $i \in Q(\boldsymbol{b})$. As $(\boldsymbol{b}_{\{T\cup i\}}, \boldsymbol{y}^S_{-\{T\cup i\}}) = \boldsymbol{b}$,

$$\begin{aligned} u_i((\boldsymbol{b}_{\{T\cup i\}}, \boldsymbol{y}^S_{-\{T\cup i\}}), y_i^S) &> u_i(\boldsymbol{y}^S, y_i^S) \\ \text{and } \forall\, j \in T:\ u_j((\boldsymbol{b}_{\{T\cup i\}}, \boldsymbol{y}^S_{-\{T\cup i\}}), y_j^S) &= y_j^S \cdot Q(\boldsymbol{b}) - x_j(\boldsymbol{b}) \\ &= y_j^S \cdot Q(\boldsymbol{y}^S) - x_j(\boldsymbol{y}^S) \\ &> u_j(\boldsymbol{y}^S, y_j^S) \\ \text{and } \forall\, j \in T \cup \{i\}:\ j \notin Q(\boldsymbol{y}^S) \setminus Q(\boldsymbol{b}_{\{T\cup i\}}, \boldsymbol{y}^S_{-\{T\cup i\}}) &, \end{aligned}$$

contradicting WCGSP. If $u_i(\boldsymbol{b}, b_i) < u_i(\boldsymbol{y}^S, b_i)$ for an $i \notin T$, we analogously get a contradiction to WCGSP by considering vector $\boldsymbol{b}$ and coalition $K = T \cup \{i\}$ with true valuations $\boldsymbol{b}_{T\cup\{i\}}$ bidding $\boldsymbol{y}^S_{T\cup\{i\}}$.

Thus, (6.1) and (6.2) hold.

In case that bids are allowed to be negative and thus strict CS comes 'for free', we adjust the proof by letting $y_i^{\boldsymbol{b},S} := -1$ for all $i \in S \setminus Q(\boldsymbol{b})$. It now directly holds for all $i \in S$ that $i \in Q(\boldsymbol{b}) \Leftrightarrow i \in Q(\boldsymbol{y}^{\boldsymbol{b},S})$ and (6.1) and (6.2) hold even for replacing T with S, as can be verified along the lines of the above proof. This approach was also used by Moulin [97] who assumed non-negative bids and strict CS, where $y_i^{\boldsymbol{b},S}$ was set such that $i \in S \setminus Q(\boldsymbol{b})$ does not receive the service when bidding $y_i^{\boldsymbol{b},S}$.

However, $i \in Q(\boldsymbol{b}) \Leftrightarrow i \in Q(\boldsymbol{y}^{\boldsymbol{b},S})$ cannot be guaranteed for all $i \in S$ when setting $y_i^{\boldsymbol{b},S} := 0$ for all $i \in S \setminus Q(\boldsymbol{b})$. The main idea of our proof for non-negative bids is to show that (6.1) holds for players in the restricted set $T := S \setminus \{j \notin Q(\boldsymbol{b}) \mid b_j = 0\}$ which still allows to derive contradictions to WCGSP.

□

Lemma 6.5. *For any cost function $C : 2^{[3]} \to \mathbb{R}_{\geq 0}$, there is a WGSP and 1-BB mechanism $M_C = (Q, x)$ such that there are bids $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n_{>0}$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$, but $x(\boldsymbol{b}) \neq x(\boldsymbol{b}')$.*

Proof. Consider mechanism $M_{C,a}$ with cost function C and threshold $a \in \mathbb{R}_{>0}$ from Algorithm 6.2:

Algorithm 6.2 (computing $M_{C,a}(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

```
x := 0, Q := {i ∈ [3] | b_i > 0}
if Q ≠ ∅ then i := min Q
while Q ≠ ∅ do
    if b_i ≥ C(Q) then x_i := C(Q); return
    else Q := Q \ {i}
    if i = 1 then if b_1 ≥ a then i := 2 else i := 3
    else if Q ≠ ∅ then i := min Q
    return (Q, x)
```

Obviously, 1-BB is met. Player 1 may only strictly improve his utility, if 2 or 3 bids in order to not receive the service for $\boldsymbol{b}$. However, such a coalition would not form. Therefore, 1 is not part of any coalition. Furthermore, 2 can only strictly improve his utility, if 3 bids in order to not receive the service for $\boldsymbol{b}$. Vice versa, the same is true for player 3. Therefore, $M_{C,a}$ is WGSP. However, payments are not uniquely determined by the served set. For $a < C([3])$,

$$\boldsymbol{b} := (a, C(\{2,3\}), \varepsilon) \Longrightarrow Q(\boldsymbol{b}) = \{2,3\} \text{ and } x(\boldsymbol{b}) = (0, C\{2,3\}, 0)$$
$$\boldsymbol{b}' := (a-\varepsilon, \varepsilon, C(\{2,3\})) \Longrightarrow Q(\boldsymbol{b}') = \{2,3\} \text{ and } x(\boldsymbol{b}') = (0, 0, C(\{2,3\})) \ .$$

□

6.4 Egalitarian Mechanisms

6.4.1 Set Selection and Price Functions

Egalitarian mechanisms borrow an algorithmic idea proposed by Dutta and Ray [34] for computing the 'egalitarian solution' for a cost-allocation problem. Given a set of players $Q \subseteq [n]$ to be served, cost shares are computed iteratively: Find the *most cost-efficient* subset S of the players that have not been assigned a cost share yet. That is, the quotient of the marginal cost for including S divided by $|S|$ is minimal. Then, assign each player in S this quotient as his cost share. If players remain who have not been assigned a cost share yet, start a new iteration.

We introduce *most cost-efficient set selection functions* σ_C and corresponding *price functions* ρ_C that are used within this iterative process. Specifically, let $Q \subseteq [n]$ be the set of players to be served. For some fixed iteration, let $N \subsetneq Q$ be the subset of players already assigned a cost-share. Then, $\sigma_C(Q,N)$ selects a non-empty set of players from $Q \setminus N$ who are assigned the cost share $\rho_C(Q,N)$ according to Definition 6.6:

Definition 6.6. *Let C be a cost function. The* most cost-efficient *set selection function $\sigma_C : 2^{[n]} \times 2^{[n]} \to 2^{[n]}$ and its corresponding price function $\rho_C : 2^{[n]} \times 2^{[n]} \to \mathbb{R}_{\geq 0}$ are defined as*

$$\sigma_C(Q,N) := \textit{ lexicographic max in } \arg \min_{\emptyset \neq T \subseteq Q \setminus N} \left\{ \frac{C(N \cup T) - C(N)}{|T|} \right\} ,$$

$$\rho_C(Q,N) := \min_{\emptyset \neq T \subseteq Q \setminus N} \left\{ \frac{C(N \cup T) - C(N)}{|T|} \right\} .$$

In general, we allow for other set selection and price functions that fulfill a certain validity requirement introduced in Section 6.4.3 which is in particular met by σ_C and ρ_C. Clearly, evaluating σ_C can take exponentially many steps (in n). Furthermore, evaluating C may be computationally hard. In Section 6.7 we thus study how to pick 'suitable' cost-efficient subsets in polynomial time.

6.4.2 Computing Egalitarian Mechanisms

Based on set selection function σ and price function ρ, we define mechanism $Egal_{\sigma,\rho}$ as computed by Algorithm 6.3. We require that for all $N \subsetneq Q \subseteq [n]$ it holds that $\emptyset \neq \sigma(Q,N) \subseteq Q \setminus N$.

Obviously, $Egal_{\sigma,\rho}$ meets NPT, VP, and strict CS. If the most cost-efficient set selection and price functions σ_C and ρ_C for an arbitrary cost function C are applied, $Egal_{\sigma_C,\rho_C}$ is even 1-BB.

Algorithm 6.3 (computing $Egal_{\sigma,\rho}(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

$Q := [n]$; $N := \emptyset$; $\boldsymbol{x} := 0$
while $N \neq Q$ **do**
  $S := \sigma(Q, N)$, $a := \rho(Q, N)$
  $Q := Q \setminus \{i \in S \mid b_i < a\}$
  if $S \subseteq Q$ **then** $x_i := a$ for all $i \in S$; $N := N \cup S$
return $(Q, \boldsymbol{x})$

Example 6.7. Consider $C : 2^{[4]} \to \mathbb{R}_{\geq 0}$ with $C(\{1,2\}) := 1$, $C(\{1,2,3\}) := 1$ and $C(S) = |S|$ otherwise. Let $\boldsymbol{v} := (\frac{1}{3}, \frac{1}{2}, \frac{1}{4}, 4)$. We compute $Egal_{\sigma_C,\rho_C}(\boldsymbol{v}) = (Q(\boldsymbol{v}), x(\boldsymbol{v}))$ for the most cost-efficient set selection an price function σ_C and ρ_C from Definition 6.6. Denote all variables in iteration k of Algorithm 6.3 with subscript k.

Then, $S_1 = \{1,2,3\}$ and $a_1 = \frac{1}{3}$. As $v_3 < \frac{1}{3}$, $Q_1 = \{1,2,4\}$ and $N_1 = \emptyset$. In the second iteration, $S_2 = \{1,2\}$, $a_2 = \frac{1}{2}$, and $Q_2 = \{1,2,4\}$. Thus, $N_2 = \{1,2\}$ and $\boldsymbol{x}_2 = (\frac{1}{2}, \frac{1}{2}, 0, 0)$. In the third iteration, $S_3 = \{4\}$, $a_3 = 2$, and $Q_3 = \{1,2,4\}$. Thus, $N_3 = \{1,2,4\} = Q(\boldsymbol{v})$ and $\boldsymbol{x}_3 = (\frac{1}{2}, \frac{1}{2}, 0, 2) = x(\boldsymbol{v})$.

We remark that the mechanisms given by Devanur et. al. [31] are not just acyclic (see [94]) but also egalitarian. Using the terminology as in [31], they could be computed by Algorithm 6.3 by letting $\sigma(Q, N)$ be the next set that 'goes tight' after all players in N have been 'frozen' and all in $[n] \setminus Q$ have been dropped.

6.4.3 Validity of Set Selection and Price Functions

For a set selection function σ and a price function ρ we demand the validity requirement from Definition 6.8 which is motivated with respect to Algorithm 6.3.

Definition 6.8. *A set selection function $\sigma : 2^{[n]} \times 2^{[n]} \to 2^{[n]}$ and a price function $\rho : 2^{[n]} \times 2^{[n]} \to \mathbb{R}_{\geq 0}$ are* valid *if, for all $N \subsetneq Q, Q' \subseteq [n]$:*

- *Any player is assigned a cost-share only once and the algorithm terminates:*
$$\emptyset \neq \sigma(Q, N) \subseteq Q \setminus N \tag{6.3}$$
- *Players in $Q \setminus N$ not in the selected set have no influence on it's selection:*
$$\sigma(Q, N) \subseteq Q' \subseteq Q \Longrightarrow \sigma(Q, N) = \sigma(Q', N) \text{ and } \rho(Q, N) = \rho(Q', N) \tag{6.4}$$
- *The assigned prices are non-decreasing throughout the iterations of the algorithm:*
$$Q' \subseteq Q \Longrightarrow \rho(Q, N) \leq \rho(Q', N) \text{ and} \tag{6.5}$$
$$0 \leq \rho(Q, N) \leq \rho(Q, N \cup \sigma(Q, N)) \tag{6.6}$$

Lemma 6.9. *For any cost function C, the corresponding most cost-efficient set selection and price functions σ_C and ρ_C (from Definition 6.6) are valid.*

Proof. It is a straightforward observation that σ_C and ρ_C fulfill properties (6.3)–(6.5) of Definition 6.8. To see property (6.6), let $N \subsetneq Q \subseteq [n]$. Define $S := \sigma(Q, N)$, $a := \rho(Q, N)$ and $S' := \sigma(Q, N \cup S), a' := \rho(Q, N \cup S)$. Then,
$$a \leq \frac{C(N \cup S \cup S') - C(N)}{|S| + |S'|} = \frac{C(N \cup S \cup S') - C(N \cup S) + |S| \cdot a}{|S| + |S'|},$$
thereby implying that
$$a \leq \frac{C(N \cup S \cup S') - C(N \cup S)}{|S'|} = a'.$$
□

6.5 Egalitarian Mechanisms are CGSP

Theorem 6.10 follows as a direct corollary of Theorem 6.14 in Section 6.5.2 showing that acyclic mechanisms are CGSP and of Theorem 6.17 in Section 6.5.3 showing that egalitarian mechanisms (for valid set selection and price functions) are acyclic.

Theorem 6.10. *For any valid set selection function σ and price function ρ, $Egal_{\sigma,\rho}$ is CGSP.*

6.5.1 Acyclic Mechanisms

By introducing acyclic mechanisms, Mehta et al. [94] gave a framework for constructing WGSP mechanisms. An acyclic mechanism $Acyc_{\xi,\tau}$ makes use of a cost-sharing method ξ and an *offer function* $\tau : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ which specifies a non-negative offer time $\tau_i(Q)$ for every subset $Q \subseteq [n]$ and every player $i \in Q$. The acyclic mechanism $Acyc_{\xi,\tau}$ is computed by Algorithm 6.4.

Algorithm 6.4 (computing $Acyc_{\xi,\tau}(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

$Q := [n]$
while $\exists i \in Q$ with $b_i < \xi_i(Q)$ **do**
  Let $j \in \arg\min_{i \in Q}\{\tau_i(Q) \mid b_i < \xi_i(Q)\}$ (use arbitrary tie breaking rule)
  $Q := Q \setminus \{j\}$
return $(Q, \xi(Q))$

Mehta et al. [94] showed that if τ satisfies a certain validity requirement with respect to ξ, $Acyc_{\xi,\tau}$ is WGSP. We specify this requirement in Definition 6.11. For every $Q \subseteq [n]$ and every $i \in Q$, let

$$E_i(Q) := \{j \in Q \mid \tau_j(Q) = \tau_i(Q)\}$$
$$L_i(Q) := \{j \in Q \mid \tau_j(Q) < \tau_i(Q)\}$$
$$G_i(Q) := \{j \in Q \mid \tau_j(Q) > \tau_i(Q)\}$$

be the sets of players with equal, lesser, and greater offer time compared to i.

Definition 6.11. *An offer function τ is* valid *for cost-sharing method ξ if for all $i \in [n]$ it holds that*

- *$\xi_i(Q \setminus T) = \xi_i(Q)$ for every subset $T \subseteq G_i(Q)$ and*
- *$\xi_i(Q \setminus T) \geq \xi_i(Q)$ for every subset $T \subseteq G_i(Q) \cup (E_i(Q) \setminus \{i\})$.*

Consider the set Q at the beginning of iteration k and a player $i \in Q$. Let $p := \xi_i(Q)$. Like in [94], we say that $Acyc_{\xi,\tau}$ *offers the price p to player i in iteration k* if either no player is deleted in iteration k (i.e., $Acyc_{\xi,\tau}$ terminates) or it holds for i and the deleted player j that $\tau_i(Q) \leq \tau_j(Q)$ (it is possible that $j = i$). We use this terminology in Lemma 6.12 and Lemma 6.13 needed for the proof that acyclic mechanisms are CGSP.

Lemma 6.12 ([94]). *Let $Acyc_{\xi,\tau} = (Q, x)$ be an acyclic mechanism and Q be the set at the beginning of iteration k. If player $i \in Q$ is offered price p_1 in iteration k and price p_2 in some subsequent iteration, then $p_1 \leq p_2$. In particular, $\xi_j(Q(\boldsymbol{b})) = \xi_j(Q)$ for all $j \in L_i(Q)$ and therefore $L_i(Q) \subseteq Q(\boldsymbol{b})$.*

Lemma 6.13. *Let $Acyc_{\xi,\tau} = (Q, x)$ be an acyclic mechanism. For any bid vector $\boldsymbol{b}$ and any set $T \subseteq [n] \setminus Q(\boldsymbol{b})$, there is an $i \in T$ with $b_i < \xi_i(Q(\boldsymbol{b}) \cup T)$.*

Proof. Consider the first iteration k in which an $i \in T$ is deleted for the first time from set Q, i.e., $T \subseteq Q$. For all $j \in Q \setminus Q(\boldsymbol{b})$, it holds that $\tau_j(Q) \geq \tau_i(Q)$. Otherwise, $j \in Q(\boldsymbol{b})$ by Lemma 6.12. In particular, $Q \setminus \{Q(\boldsymbol{b}) \cup T\} \subseteq G_i(Q) \cup (E_i(Q) \setminus \{i\})$. Therefore, by validity of τ, $b_i < \xi_i(Q) \leq \xi_i(Q \setminus \{Q \setminus \{Q(\boldsymbol{b}) \cup T\}\}) = \xi_i(Q(\boldsymbol{b}) \cup T)$. □

6.5.2 Acyclic Mechanisms are CGSP

Theorem 6.14. *For any cost-sharing method ξ and any offer function τ that is valid for ξ, $Acyc_{\xi,\tau}$ is CGSP.*

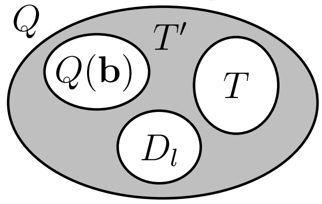
Proof. Assume that $Acyc_{\xi,\tau}$ is not CGSP. Then, there exists a coalition $K \subseteq [n]$ with true valuations $\boldsymbol{v}_K = (v_i)_{i \in K} \in \mathbb{R}^{|K|}$ and a bid vector $\boldsymbol{b} \in \mathbb{R}^n$ such that for all $i \in K$, $u_i(\boldsymbol{b}, v_i) \geq u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ and $i \notin Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K}) \setminus Q(\boldsymbol{b})$ and for at least one $i \in K$, $u_i(\boldsymbol{b}, v_i) > u_i((\boldsymbol{v}_K, \boldsymbol{b}_{-K}), v_i)$ or $i \in Q(\boldsymbol{b}) \setminus Q(\boldsymbol{v}_K, \boldsymbol{b}_{-K})$. Without loss of generality, let $\boldsymbol{v} := (\boldsymbol{v}_K, \boldsymbol{b}_{-K})$.

First, we show that $Q(\boldsymbol{b}) \subseteq Q(\boldsymbol{v})$. Let $m := n - |Q(\boldsymbol{v})|$. Consider the first non-identical iteration $k \in [m]$ for $\boldsymbol{v}$ and $\boldsymbol{b}$. Denote Q to be the set of players at the beginning of k. If $Q = Q(\boldsymbol{v})$, there is nothing to show. Denote the player deleted for $\boldsymbol{v}$ in iteration $l \in [m]$ by d_l. Let $D_l := \{d_k, \ldots, d_l\}$. Note that $Q(\boldsymbol{v}) = Q \setminus D_m$. We prove by induction that $d_l \notin Q(\boldsymbol{b})$ for all $l \in \{k, \ldots, m\}$.

- $l = k$:
  If $Q = Q(\boldsymbol{b})$, $b_{d_k} \geq \xi_{d_k}(Q) > v_{d_k}$. Then, $d_k \in K$ and $u_{d_k}(\boldsymbol{b}, v_{d_k}) < 0 = u_{d_k}(\boldsymbol{v}, v_{d_k})$, a contradiction. Thus, let d'_k be the player deleted for $\boldsymbol{b}$ in iteration k. If $\tau_{d'_k}(Q) < \tau_{d_k}(Q)$, then by Lemma 6.12, $d'_k \in Q(\boldsymbol{v})$ and $v_{d'_k} \geq \xi_{d'_k}(Q(\boldsymbol{v})) = \xi_{d'_k}(Q) > b_{d'_k}$. But $d'_k \in K$ contradicts $d'_k \notin Q(\boldsymbol{b})$. Thus, $\tau_{d'_k}(Q) \geq \tau_{d_k}(Q)$, implying that for $\boldsymbol{b}$, d_k is offered price $\xi_{d_k}(Q)$ in iteration k. If $d_k \in Q(\boldsymbol{b})$, then by Lemma 6.12, $b_{d_k} \geq \xi_{d_k}(Q(\boldsymbol{b})) \geq \xi_{d_k}(Q) > v_{d_k}$. Then, $d_k \in K$ and $u_{d_k}(\boldsymbol{b}, v_{d_k}) < 0 = u_{d_k}(\boldsymbol{v}, v_{d_k})$, a contradiction.
- $l \to l + 1$:
  Assume $d_{l+1} \in Q(\boldsymbol{b})$. By induction assumption, $Q(\boldsymbol{b}) \subseteq Q \setminus D_l$. By $\xi_{d_{l+1}}(Q \setminus D_l) > v_{d_{l+1}}$, we get $\xi_{d_{l+1}}(Q(\boldsymbol{b})) < \xi_{d_{l+1}}(Q \setminus D_l)$. Otherwise, it holds that $d_{l+1} \in K$ and $u_{d_{l+1}}(\boldsymbol{b}, v_{d_{l+1}}) < 0 = u_{d_{l+1}}(\boldsymbol{v}, v_{d_{l+1}})$, a contradiction. Thus, by validity of τ, there exists $i \in Q \setminus D_l$ with $i \in L_{d_{l+1}}(Q \setminus D_l)$ and $i \notin Q(\boldsymbol{b})$. Let
  $$T := \{i \in Q \setminus D_l \mid i \in L_{d_{l+1}}(Q \setminus D_l) \text{ and } i \notin Q(\boldsymbol{b})\} \ .$$
  It is $T \cap K = \emptyset$, since $T \subseteq Q(\boldsymbol{v})$ (confer Lemma 6.12). Note that D_l, $Q(\boldsymbol{b})$, and T are disjoint. Let $T' := Q \setminus (D_l \cup Q(\boldsymbol{b}) \cup T)$. Figure 6.1 shows the relations between sets $Q, D_l, Q(\boldsymbol{b}), T$, and T'. By definition of T, it holds that for all $j \in T'$ that $\tau_j(Q \setminus D_l) \geq \tau_{d_{l+1}}(Q \setminus D_l)$, as $j \notin T$. Thus, for all $j \in T'$ and all $i \in T$, we have that $\tau_j(Q \setminus D_l) > \tau_i(Q \setminus D_l)$. Therefore,
  $$\text{for all } i \in T: \ b_i = v_i \geq \xi_i(Q \setminus D_l) = \xi_i(Q \setminus (D_l \cup T')) = \xi_i(Q(\boldsymbol{b}) \cup T) \ .$$
  However, this contradicts Lemma 6.13. Therefore, $d_{l+1} \notin Q(\boldsymbol{b})$.

From $Q(\boldsymbol{b}) \subseteq Q(\boldsymbol{v})$, it now follows that $K \cap Q(\boldsymbol{v}) = K \cap Q(\boldsymbol{b})$. Hence, by assumption, there exists $j \in K$ with $j \in Q(\boldsymbol{b}) \cap Q(\boldsymbol{v})$ such that $u_j(\boldsymbol{b}, v_j) > u_j(\boldsymbol{v}, v_j)$, and specifically $\xi_j(Q(\boldsymbol{b})) < \xi_j(Q(\boldsymbol{v}))$. Then, there has to be at least one player $i \in Q(\boldsymbol{v})$ with $i \in L_j(Q(\boldsymbol{v}))$ and $i \notin Q(\boldsymbol{b})$. Defining $T := \{i \in Q(\boldsymbol{v}) \mid i \in L_j(Q(\boldsymbol{v})) \text{ and } i \notin Q(\boldsymbol{b})\}$ and $T' := Q(\boldsymbol{v}) \setminus (Q(\boldsymbol{b}) \cup T)$ leads again to a contradiction to Lemma 6.13. □

Fig. 6.1. Set relations for the proof of Theorem 6.14



6.5.3 Egalitarian Mechanisms are Acyclic

In order to show that egalitarian mechanisms are acyclic, we make the observation that instead of using the tie-breaking rule in line 3 of Algorithm 6.4, we may very well simultaneously delete all players approved for deletion at the same time:

Lemma 6.15. *Lines 3 and 4 of Algorithm 6.4 can be replaced by*

$$Q := Q \setminus \arg\min_{i \in Q}\{\tau_i(Q) \mid b_i < \xi_i(Q)\}$$

without changing the computed mechanism $Acyc_{\xi,\tau}$.

Proof. The proof uses the same technique as the proof of Theorem 6.14. Define mechanism (Q', x') by replacing lines 3 and 4 of Algorithm 6.4 computing $Acyc_{\xi,\tau} = (Q, x)$ by '$Q := Q \setminus \arg\min_{i \in Q}\{\tau_i(Q) \mid b_i < \xi_i(Q)\}$'. We show that $(Q', x') = (Q, x)$.

It is an easy observation that Lemma 6.12 and Lemma 6.13 hold as well for (Q', x'). Fix an arbitrary bid vector $\boldsymbol{b} \in \mathbb{R}^n$. First, we show that $Q(\boldsymbol{b}) \subseteq Q'(\boldsymbol{b})$. Consider the first non-identical iteration k for (Q, x) and (Q', x'). Let Q be the set of players at the beginning of k. If $Q = Q'(\boldsymbol{b})$, there is nothing to show. Assume computing $(Q'(\boldsymbol{b}), x'(\boldsymbol{b}))$ needs m iterations and denote the players deleted by (Q', x') in iteration $l \in [m]$ by $\mathcal{D}_l$. Let $D_l := \mathcal{D}_k \cup \ldots \cup \mathcal{D}_l$. Note that $Q'(\boldsymbol{b}) = Q \setminus D_m$. We prove by induction that $\mathcal{D}_l \cap Q(\boldsymbol{b}) = \emptyset$ for all $l \in \{k, \ldots, m\}$.

- $l = k$:
  Assume there is a $d \in \mathcal{D}_k$ with $d \in Q(\boldsymbol{b})$. Since $b_d(Q) < \xi_d(Q)$, there has to be an $i \in L_d(Q)$ with $i \notin Q(\boldsymbol{b})$, a contradiction to Lemma 6.12.
- $l \to l + 1$:
  Assume that there is a $d \in \mathcal{D}_{l+1}$ with $d \in Q(\boldsymbol{b})$. By induction hypothesis, $Q(\boldsymbol{b}) \subseteq Q \setminus D_l$. Since $b_d < \xi_d(Q \setminus D_l)$, there has to be an $i \in L_d(Q \setminus D_l)$ with $i \notin Q(\boldsymbol{b})$. Define

  $$T := \{i \in Q \setminus D_l \mid i \in L_d(Q \setminus D_l) \text{ and } i \notin Q(\boldsymbol{b})\} \text{ and } T' := Q \setminus (D_l \cup Q(\boldsymbol{b}) \cup T)\ .$$

  By definition of T, it holds that for all $j \in T'$, we have that $\tau_j(Q \setminus D_l) > \tau_i(Q \setminus D_l)$ for all $i \in T$. Hence, $b_i \geq \xi_i(Q \setminus D_l) = \xi_i(Q \setminus (D_l \cup T')) = \xi_i(Q(\boldsymbol{b}) \cup T)$ for all $i \in T$. However, this contradicts Lemma 6.13. Therefore, $\mathcal{D}_{l+1} \cap Q(\boldsymbol{b}) = \emptyset$.

Furthermore, $Q'(\boldsymbol{b}) \subseteq Q(\boldsymbol{b})$ can be proven analogously. Then, $x(\boldsymbol{b}) = x'(\boldsymbol{b})$ directly follows from $Q(\boldsymbol{b}) = Q'(\boldsymbol{b})$ and Theorems 6.14 and 6.4.

□

Finally, Theorem 6.17 showing that egalitarian mechanisms are acyclic, is based on the fact that Algorithm 6.4 computes $Egal_{\sigma,\rho}$ for the cost-sharing method $\xi^{\sigma,\rho}$ and offer function $\tau^{\sigma,\rho}$ as defined by Algorithm 6.5.

Algorithm 6.5 (computing $\xi^{\sigma,\rho}(Q)$ and $\tau^{\sigma,\rho}(Q)$).
Input: set selection and price functions σ, ρ; set of players $Q \subseteq [n]$
Output: cost-sharing vector $\xi^{\sigma,\rho}(Q) \in \mathbb{R}^n_{\geq 0}$; offer-time vector $\tau^{\sigma,\rho}(Q) \in \mathbb{R}^n_{\geq 0}$

1: $N := \emptyset$; $\boldsymbol{\xi} := 0$; $\boldsymbol{\tau} := 0$
2: **while** $N \neq Q$ **do**
3: $\quad S := \sigma(Q, N)$, $a := \rho(Q, N)$
4: $\quad \xi_i := a$ and $\tau_i := 1 + \max_{j \in Q}\{\tau_j\}$ for all $i \in S$; $N := N \cup S$
5: **return** $(\boldsymbol{\xi}, \boldsymbol{\tau})$

Example 6.16. Continuing Example 6.7, is is $\xi^{\sigma_C,\rho_C}(\{1,2,3,4\}) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 3)$ and $\tau^{\sigma_C,\rho_C}(\{1,2,3,4\}) = (1,1,1,2)$, furthermore $\xi^{\sigma_C,\rho_C}(\{2,3,4\}) = (0,1,1,1)$ and $\tau^{\sigma_C,\rho_C}(\{2,3,4\}) = (0,1,1,1)$, and for another example, $\xi^{\sigma_C,\rho_C}(\{1,2,4\}) = (\frac{1}{2}, \frac{1}{2}, 0, 2)$ and $\tau^{\sigma_C,\rho_C}(\{2,3,4\}) = (1,1,0,2)$.

Theorem 6.17. *For any cost-sharing method ξ and any offer function τ that is valid for ξ, $Egal_{\xi,\tau}$ is acyclic.*

Proof. We show that for any valid σ and ρ, $Egal_{\sigma,\rho}$ can also be computed by the acyclic mechanism from Algorithm 6.4 with cost-sharing method $\xi^{\sigma,\rho}$ and offer function $\tau^{\sigma,\rho}$ as computed by Algorithm 6.5, and hence is acyclic.

Fix valid σ and ρ. Furthermore, set $\xi := \xi^{\sigma,\rho}$ and $\tau := \xi^{\sigma,\rho}$.

- τ is valid for ξ:
  Denote all variables in iteration k of Algorithm 6.5 just before line 3 with a subscript k and with the input player set in parentheses. Let $Q \subseteq [n]$ and $i \in Q$ be arbitrary. Let k be the iteration such that $i \in S_k(Q)$.
  - For all $T \subseteq G_i(Q)$ it follows by induction that $N_m(Q) = N_m(Q \setminus T)$ and $S_m(Q) = S_m(Q \setminus T)$ and $a_m(Q) = a_m(Q \setminus T)$ for all $m \in [k]$. This follows by by Property (6.4) of Definition 6.8 because $S_m(Q) \subseteq Q \setminus T$. Hence, $\xi_i(Q) = a_k(Q) = a_k(Q \setminus T) = \xi_i(Q \setminus T)$.
  - For all $T \subseteq G_i(Q) \cup (E_i(Q) \setminus \{i\})$, it is $N_m(Q) = N_m(Q \setminus T)$ and $S_m(Q) = S_m(Q \setminus T)$ and $a_m(Q) = a_m(Q \setminus T)$ for all $m \in [k-1]$. Since still $N_k(Q) = N_k(Q \setminus T)$ and $Q \setminus T \subseteq Q$, it is $a_k(Q) \leq a_k(Q \setminus T)$ because of Property (6.5). Furthermore, $a_k(Q \setminus T) \leq \xi_i(Q \setminus T)$ since a is non-decreasing in Algorithm 6.5 by Property (6.6). Thus, $\xi_i(Q) = a_k(Q) \leq a_k(Q \setminus T) \leq \xi_i(Q \setminus T)$.
- $Egal_{\sigma,\rho} = Acyc_{\xi,\tau}$:
  Whenever Algorithm 6.3 accepts a set $S := \sigma(Q, N)$ this means that players in S have the minimum offering time of those in $Q \setminus N$ and $b_i \geq a := \rho(Q, N)$ for all $i \in S$. Consequently, also the acyclic mechanism serves these players for the same price. On the other hand, when Algorithm 6.3 rejects players from S, the same players are also rejected by the acyclic mechanism (see Lemma 6.15).

□

6.6 Efficiency of Egalitarian Mechanisms

Theorem 6.18 is the main theorem of this section and directly follows from Theorem 6.21 and Lemma 6.22. Specifically, Theorem 6.21 provides a way to determine the efficiency approximation not only for the most cost-efficient, but for any valid set of set-selection and price functions, via the *β-average* property of price functions introduced in Definition 6.19.

Theorem 6.18. *For any subadditive cost function C and most cost-efficient set selection and price functions σ_C, ρ_C, $Egal_{\sigma_C,\rho_C}$ is $2H_n$-EFF for C.*

Definition 6.19. *Let C be a cost function, ρ be a price function, and $\beta > 0$. Then, ρ is called* β-average for C *if for all $N \subsetneq Q \subseteq [n]$ and all $\emptyset \neq S \subseteq Q \setminus N$, it holds that $\rho(Q, N) \leq \beta \cdot \frac{C(S)}{|S|}$.*

Lemma 6.20. *Let C be a cost function and σ and ρ be valid set selection and price functions such that ρ is β-average for C. Moreover, let $S \subseteq [n]$ and $\boldsymbol{b} \in \mathbb{R}^n$ be a bid vector with $b_i \geq \beta \cdot \frac{C(S)}{|S|}$ for all $i \in S$. Then, $Egal_{\sigma,\rho} = (Q, x)$ serves at least one player $i \in S$, i.e., $S \cap Q(\boldsymbol{b}) \neq \emptyset$.*

Proof. Assume that $S \cap Q(\boldsymbol{b}) = \emptyset$. Consider the first iteration k in which Algorithm 6.3 rejects a player $i \in S$: This happens in line 4. We indicate all variables in that iteration immediately before that line by subscript k. Since player i is dropped,

$$b_i < a_k = \rho(Q_k, N_k) \leq \beta \cdot \frac{C(S)}{|S|} \,,$$

where the last inequality holds because of $S \subseteq Q_k \setminus N_k$. A contradiction. □

Theorem 6.21. *Let C and C' be cost functions and σ and ρ be valid set selection and price functions such that ρ is β-average for C. Let $Egal_{\sigma,\rho} = (Q, x)$. Then, if for all $\boldsymbol{b} \in \mathbb{R}^n$, $\sum_{i=1}^n x_i(\boldsymbol{b}) \geq C'(Q(\boldsymbol{b}))$, $Egal_{\sigma,\rho}$ is $(2\beta \cdot H_n)$-EFF for C and C'.*

Proof. Let $Egal_{\sigma,\rho} = (Q, x)$ and fix a true valuation vector $\boldsymbol{v} \in \mathbb{R}^n$. Denote $Q := Q(\boldsymbol{v})$, $\boldsymbol{x} := x(\boldsymbol{v})$. Moreover, let $P \subseteq [n]$ be a set that minimizes the optimal social cost, i.e., $P \in \arg\min_{T \subseteq [n]}\{SC_C(T, \boldsymbol{v})\}$. Without loss of generality, we may assume that $v_i \geq 0$ for all $i \in [n]$ since negative bids have no impact on the social cost. We have

$$\begin{aligned}
SC_{C'}(Q, \boldsymbol{v}) &= C'(Q) + \sum_{i \in [n] \setminus Q} v_i \\
&\leq \sum_{i \in Q \cap P} x_i + \sum_{i \in Q \setminus P} \underbrace{x_i}_{\leq v_i} + \sum_{i \in [n] \setminus Q} v_i \\
&\leq \sum_{i \in Q \cap P} x_i + \sum_{i \in P \setminus Q} v_i + \sum_{i \in [n] \setminus P} v_i \ , \text{ and thus}
\end{aligned}$$

$$\begin{aligned}
\frac{SC_{C'}(Q, \boldsymbol{v})}{SC_C(P, \boldsymbol{v})} &\leq \frac{\sum_{i \in Q \cap P} x_i + \sum_{i \in P \setminus Q} v_i + \sum_{i \in [n] \setminus P} v_i}{C(P) + \sum_{i \in [n] \setminus P} v_i} \\
&\leq \frac{\sum_{i \in Q \cap P} x_i + \sum_{i \in P \setminus Q} v_i}{C(P)} \ .
\end{aligned}$$

The last inequality holds since the left fraction is at least 1. Now, consider the iteration k when for the first time Algorithm 6.3 decides to accept a player $i \in Q \cap P$ (line 5). Fix all variables just after line 3 in that iteration k and indicate them with a subscript k. We have $x_i = a_k = \rho(Q_k, N_k) \leq \beta \cdot \frac{C(Q \cap P)}{|Q \cap P|}$, because $Q \cap P \subseteq Q_k \setminus N_k$. With the same arguments, for the second player $i \in Q \cap P$, we can bound his cost-share $x_i \leq \beta \cdot \frac{C(Q \cap P)}{|Q \cap P| - 1}$, and so forth. Finally, $\sum_{i \in Q \cap P} x_i \leq \beta \cdot H_{|Q \cap P|} \cdot C(Q \cap P)$.

On the other hand, in $P \setminus Q$, there is at least one player i with $v_i < \beta \cdot \frac{C(P\setminus Q)}{|P\setminus Q|}$. Otherwise, due to Lemma 6.20, we would have $(P \setminus Q) \cap Q \neq \emptyset$, a contradiction. Inductively and by the same lemma, for every $j = 1, \ldots, |P \setminus Q| - 1$, there has to be a player $i \in P \setminus Q$ with $v_i < \beta \cdot \frac{C(P\setminus Q)}{|P\setminus Q|-j}$. Finally, $\sum_{i \in P\setminus Q} v_i \leq \beta \cdot H_{|P\setminus Q|} \cdot C(P \setminus Q)$. Since C is non-decreasing, we get

$$\frac{SC_{C'}(Q, \boldsymbol{v})}{SC_C(P, \boldsymbol{v})} \leq \frac{\beta \cdot H_{\max\{|Q\cap P|, |P\setminus Q|\}} \cdot (C(Q \cap P) + C(P \setminus Q))}{C(P)} \leq 2\beta \cdot H_n\,.$$

Lemma 6.22. *For any subadditive cost function C, the most cost-efficient price function ρ_C is 1-average for C.*

Proof. Let $N \subsetneq Q \subseteq [n]$ and $\emptyset \neq S \subseteq Q \setminus N$. Then,

$$\rho_C(Q, N) = \min_{\emptyset \neq T \subseteq Q\setminus N} \left\{ \frac{C(Q \cup T) - C(Q)}{|T|} \right\} \leq \frac{C(Q \cup S) - C(Q)}{|S|} \leq \frac{C(S)}{|S|}\,.$$

□

We conclude by showing that our efficiency bound is tight up to a factor of 2 and that the approximate efficiency of egalitarian mechanisms is unbounded for arbitrary cost functions.

Lemma 6.23. *For the cost function C with $C(T) = 1$ for all $\emptyset \neq T \subseteq [n]$, $Egal_{\sigma_C, \rho_C}$ for most cost-efficient set selection and price functions σ_C and ρ_C is no better than H_n-EFF for C.*

Proof. Let $\boldsymbol{v} := (\frac{1}{i} - \epsilon)_{i=1}^n \in \mathbb{R}^n_{>0}$ be the true valuation vector, $\epsilon \in (0, \frac{1}{n})$. Then, $Q(\boldsymbol{v}) = \emptyset$ because in Algorithm 6.3, line 4, one player after the other would be dropped. However, $SC_C([n], \boldsymbol{v}) = 1$ while $SC_C(\emptyset, \boldsymbol{v}) = H_n - n \cdot \epsilon$.

□

Lemma 6.24. *For any $\gamma > 1$, there is a cost function C for which $Egal_{\sigma_C, \rho_C}$ for most cost-efficient set selection and price functions σ_C and ρ_C is no better than γ-EFF for C.*

Proof. Define $C : 2^{[4]} \to \mathbb{R}_{\geq 0}$ as follows: Let $C(\{i\}) = 1$ for all $i \in [4]$. Let $C(\{1, 2\}) := 2$ and $C(T) := 3$ for any other $T \subsetneq [4]$ with $|T| = 2$. Let $C(\{1, 2, 3\}) := 4$ and $C(T) := 5$ for any other $T \subsetneq [4]$ with $|T| = 3$. Furthermore, $C([4]) := M \in \mathbb{R}_{>0}$, where M is sufficiently large.

Let $Egal_{\sigma_C, \rho_C} = (Q, x)$ and let the true valuation vector be $\boldsymbol{v} = (1, 1, 2, \frac{M}{2})$. The algorithm first accepts $\{1, 2\}$, each for a price of 1. Subsequently, it gives the service to 3 for a price of 2 and in the next iteration, player 4 is rejected. Therefore, $Q(\boldsymbol{v}) = \{1, 2, 3\}$. We have that $SC_C(\{1, 2, 3\}, \boldsymbol{v}) = 4 + \frac{M}{2}$ and $SC_C(\{2, 3, 4\}, \boldsymbol{v}) = 6$.

□

6.7 Computational Framework

For a given optimization problem Π, the computational complexity of the corresponding cost-sharing problems can be partially eased by resorting to approximate solutions. Certainly, this does not yet remedy the need to iterate through all available subsets in order to pick the most cost-efficient one. The basic idea therefore consists of using an approximation algorithm ALG that induces *monotonic* costs C_{ALG} (see, e.g., [100]): Seemingly favorable changes to the input must not worsen the algorithm's performance. In the problems considered here, every player is endowed with a size (e.g., processing time in the case of scheduling) and replacing a player with a player with smaller size must not increase the cost of the algorithm's solution. We can then simply number the players in the order of their size such that $C_{\mathsf{ALG}}(\min_{|U|} T) \leq C_{\mathsf{ALG}}(U)$ for all $U \subseteq T \subseteq [n]$. Finding the most cost-efficient set then only requires iterating through all possible cardinalities.

We generalize this basic idea such that only a (polynomial-time computable) monotonic bound C_{mono} on C_{ALG} is needed whereas the costs C_{ALG} induced by ALG itself do not need to be monotonic any more.

Definition 6.25. *Let Π be an optimization problem and $(I_S)_{S\subseteq[n]}$ specify a Π-*cost-sharing problem. *A tuple $R := (\mathsf{ALG}, C_{\text{mono}})$ is a β-*relaxation for $(I_S)_{S\subseteq[n]}$ *if ALG is an approximation algorithm for Π inducing cost C_{ALG} and C_{mono} is a cost function such that the following holds:*

- *For all $T \subseteq [n]$: $C_{\mathsf{ALG}}(T) \leq C_{\text{mono}}(T) \leq \beta \cdot C(T)$.*
- *For all $U \subseteq T \subseteq [n]$: $C_{\text{mono}}(\min_{|U|} T) \leq C_{\text{mono}}(U)$.*

Note that C_{mono} does not necessarily have to be subadditive (as required for $2H_n$-EFF in Section 6.6), even if C is. Thus, some additional care is needed. Given a β-relaxation, we can adapt the most cost-efficient set selection and price functions from Definition 6.6 to obtain valid and β-average set selection and price functions:

Definition 6.26. *Given a β-relaxation $R := (\mathsf{ALG}, C_{\text{mono}})$, we define set selection and price functions σ_R and ρ_R recursively as follows: For $N \subsetneq Q \subseteq [n]$, let $\xi^{\sigma_R,\rho_R}(N)$ as computed by Algorithm 6.5. Furthermore, let*

$$k := \max\left\{\arg\min_{i\in[|Q\setminus N|]}\left\{\frac{C_{\text{mono}}(N \cup MIN_i(Q \setminus N)) - \sum_{i\in N}\xi_i^{\sigma_R,\rho_R}(N)}{i}, \frac{C_{\text{mono}}(MIN_i(Q\setminus N))}{i}\right\}\right\}$$

and $S := MIN_k(Q \setminus N)$. Then, $\sigma_R(Q,N) := S$ and

$$\rho_R(Q,N) := \min\left\{\frac{C_{\text{mono}}(N\cup S) - \sum_{i\in N}\xi_i^{\sigma_R,\rho_R}(N)}{k}, \frac{C_{\text{mono}}(S)}{k}\right\}.$$

Note that this recursion is well-defined. Computing $\sigma_R(Q,N)$ and $\rho_R(Q,N)$ requires $\xi^{\sigma_R,\rho_R}(N)$ for which only $\sigma_R(N,\cdot)$ and $\rho_R(N,\cdot)$ is needed (unless $N = \emptyset$). Yet, $N \subsetneq Q$ by assumption.

Lemma 6.27. *Let $R = (\mathsf{ALG}, C_{\text{mono}})$ be a β-relaxation for some Π-cost-sharing problem $(I_S)_{S\subseteq[n]}$. Then σ_R and ρ_R are valid, and ρ_R is β-average for C.*

Proof. Let $\sigma := \sigma_R$ and $\rho := \rho_R$.

- σ and ρ are valid (confer Definition 6.8):
  Clearly, properties (6.3) and (6.4) are fulfilled. To see (6.5), let $N \subsetneq Q' \subseteq Q \subseteq [n]$. Define $\Sigma(N) := \sum_{i\in N}\xi_i^{\sigma,\rho}(N)$ and $S := \sigma(Q,N), k := |S|$ and $S' := \sigma(Q',N), k' := |S'|$. Since $1 \leq k' \leq |Q' \setminus N| \leq |Q \setminus N|$,

$$\begin{aligned}\rho(Q,N) &\le \frac{C_{\text{mono}}(MIN_{k'}(Q\setminus N))}{k'}\\ &\le \frac{C_{\text{mono}}(MIN_{k'}(Q'\setminus N))}{k'}\\ &= \frac{C_{\text{mono}}(S')}{k'}\ .\end{aligned}$$

Furthermore,

$$\begin{aligned}\rho(Q,N) &\le \frac{C_{\text{mono}}(N\cup MIN_{k'}(Q\setminus N))-\Sigma(N)}{k'}\\ &\le \frac{C_{\text{mono}}(N\cup MIN_{k'}(Q'\setminus N))-\Sigma(N)}{k'}\\ &= \frac{C_{\text{mono}}(N\cup S')-\Sigma(N)}{k'}\ .\end{aligned}$$

Since $\rho(Q',N)$ is equal to one of these upper bounds, we have $\rho(Q,N)\le\rho(Q',N)$. Finally, to see property (6.6), let $N\subsetneq Q\subseteq[n]$ and define $S:=\sigma(Q,N), k:=|S|$ and $N':=N\cup S, S':=\sigma(Q,N'), k':=|S'|$. Then,

$$\rho(Q,N)\le\frac{C_{\text{mono}}(MIN_{k'}(Q\setminus N))}{k'}\le\frac{C_{\text{mono}}(MIN_{k'}(Q\setminus N'))}{k'}=\frac{C_{\text{mono}}(S')}{k'}\ .$$

Moreover, it is $MIN_{k+k'}(Q\setminus N)=S\cup S'$. Also, it is easy to see that $\Sigma(N')=\Sigma(N)+k\cdot\rho(Q,N)$ by making use of property (6.4) (similarly as in first part of the proof of Theorem 6.17). Consequently,

$$\begin{aligned}\rho(Q,N) &\le \frac{C_{\text{mono}}(N\cup S\cup S')-\Sigma(N)}{k+k'}\\ &= \frac{C_{\text{mono}}(N'\cup S')-\Sigma(N)}{k+k'}\\ &= \frac{C_{\text{mono}}(N'\cup S')-\Sigma(N')+k\cdot\rho(Q,N)}{k+k'}\ ,\end{aligned}$$

implying that

$$\rho(Q,N)\le\frac{C_{\text{mono}}(N'\cup S')-\Sigma(N')}{k'}\ .$$

Again, $\rho(Q,N')$ coincides with one of the upper bounds, thus it follows that $\rho(Q,N)\le\rho(Q,N\cup\sigma(Q,N))$.

- ρ is β-average for C:
  Let $N\subsetneq Q\subseteq[n]$ and $S\subseteq Q\setminus N$. Then,

$$\rho(Q,N)\le\frac{C_{\text{mono}}(MIN_{|S|}(Q\setminus N))}{|S|}\le\frac{C_{\text{mono}}(S)}{|S|}\le\frac{\beta\cdot C(S)}{|S|}\ .$$

□

With Theorem 6.10, we know that $Egal_{\sigma_R,\rho_R}$ is CGSP. To also be able to apply Theorem 6.21, we have to check that the sum of the cost-shares computed by $Egal_{\sigma_R,\rho_R}$ always covers the cost of some solution for the instance induced by the selected players. This can generally not be ensured for the solution computed by ALG. In order to compute a proper solution, we need Definition 6.28 and some further notation: For a solution X for instance I_S of a cost-sharing problem $(I_S)_{S\subseteq[n]}$, we denote by $C_X(S)$ the value of solution X (i.e., $C_{\mathsf{ALG}}(S)=C_{\mathsf{ALG}(I_S)}(S)$).

Definition 6.28. *For an optimization problem Π, consider a Π-cost-sharing problem $(I_S)_{S\subseteq[n]}$. Then, $(I_S)_{S\subseteq[n]}$ is called* mergable *if for all disjoint $T, U \subseteq [n]$, $T \cap U = \emptyset$, and for all feasible solutions X for I_T and Y for I_U with costs $C_X(T)$ and $C_Y(U)$, respectively, there is a feasible solution Z for $I_{T\cup U}$ with cost $C_Z(T \cup U)$ such that $C_Z(T \cup U) \leq C_X(T) + C_Y(T)$. We denote this operation by $Z = X \oplus Y$.*

Based on σ_R and ρ_R, Algorithm 6.6 solves all of the service provider's tasks, including computing a feasible solution of the underlying optimization problem. Remember that for a problem instance I, we let $\mathsf{ALG}(I)$ denote the solution computed by ALG. We address the running time afterwards.

Algorithm 6.6 (computing $Egal_{\sigma_R,\rho_R}$ via β-Relaxations).
Input: β-relaxation $R = (\mathsf{ALG}, C_{\text{mono}})$; bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output: player set $Q(\boldsymbol{b}) \in 2^{[n]}$, cost-share vector $\boldsymbol{x}(\boldsymbol{b}) \in \mathbb{R}^n_{\geq 0}$, solution $Z(\boldsymbol{b})$ for $I_{Q(\boldsymbol{b})}$

1: $\boldsymbol{x} := 0$, $Q := [n]$, $N := \emptyset$, $Z :=$ 'empty solution'
2: **while** $N \neq Q$ **do**
3: $\quad S := \sigma_R(Q, N)$; $a := \rho_R(Q, N)$
4: $\quad Q := Q \setminus \{i \in S \mid b_i < a\}$
5: $\quad$ **if** $S \subseteq Q$ **then**
6: $\quad\quad Z := \begin{cases} \mathsf{ALG}(I_{N\cup S}) & \text{if } C_{\text{mono}}(N \cup S) - \sum_{i\in N} x_i \leq C_{\text{mono}}(S) \\ Z \oplus \mathsf{ALG}(I_S) & \text{otherwise} \end{cases}$
7: $\quad\quad N := N \cup S$; $x_i := a$ for all $i \in S$
8: **return** $(Q, \boldsymbol{x}, Z)$

Lemma 6.29. *Let $R = (\mathsf{ALG},\ C_{\text{mono}})$ be a β-relaxation for a mergable Π-cost-sharing problem $(I_S)_{S\subseteq[n]}$.*

1. *At the end of each iteration of Algorithm 6.6, it holds that $\boldsymbol{x} = \xi^{\sigma_R,\rho_R}(N)$.*
2. *Line 3 of Algorithm 6.6 needs at most $2n$ evaluations of C_{mono}.*
3. *The mechanism computed by Algorithm 6.6 is β-BB (for costs induced by the returned solution Z).*

Proof. Let $\boldsymbol{b} \in \mathbb{R}^n$ be the input bid vector, and $Q := Q(\boldsymbol{b})$, $\boldsymbol{x} := x(\boldsymbol{b})$, and $Z := Z(\boldsymbol{b})$ be the output set of served players, cost-share vector, and solution. Assume there are m iterations. For any $k \in [m]$, indicate the value of all variables at the end of the k-th iteration (just after line 7) with a subscript k.

1. Let $\sigma := \sigma_R$ and $\rho := \rho_R$. We show by induction over $k \in [m]$ that $\xi^{\sigma,\rho}(N_i) = \boldsymbol{x}_i$, by making frequent use of validity property (6.4).
   The base case $\boldsymbol{x}_1 = \xi^{\sigma,\rho}(N_1)$ holds trivially if line 5 evaluated to false in iteration 1. Otherwise, it holds since $N_1 = \sigma(Q, \emptyset) = \sigma(N_1, \emptyset)$ and $(\boldsymbol{x}_1)_i = \rho(Q, \emptyset) = \rho(N_1, \emptyset)$ for all $i \in N_1$.
   Now consider the induction step $k - 1 \to k$. Without loss of generality, assume that line 5 evaluated to true in iteration k as otherwise $N_k = N_{k-1}$ and $\boldsymbol{x}_k = \boldsymbol{x}_{k-1}$. Observe that $N_k = N_{k-1} \cup \sigma(Q, N_{k-1})$. As above, $S := \sigma(Q, N_{k-1}) = \sigma(N_k, N_{k-1})$ and $a := \rho(Q, N_{k-1}) = \rho(N_k, N_{k-1})$. Define $\boldsymbol{y}$ with $y_i := a$ for all $i \in S$ and 0 otherwise. Since $\xi^{\sigma,\rho}(N_k) = \xi^{\sigma,\rho}(N_{k-1}) + \boldsymbol{y}$ and $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \boldsymbol{y}$, $\boldsymbol{x}_k = \xi^{\sigma,\rho}(N_k)$ by induction hypothesis.
2. The claim directly follows from (1).
3. Define $\Sigma(N_k) := \sum_{i\in N_k} x_i$. We show that $C_{Z_k}(N_k) \leq \Sigma(N_k) \leq C_{\text{mono}}(N_k)$ by induction over $k \in [m]$.

The base case $k = 1$ holds because $C_{Z_1}(N_1) = C_{\mathsf{ALG}}(N_1) \leq C_{\mathrm{mono}}(N_1) = \Sigma(N_1)$. Now consider the induction step $k-1 \to k$. Without loss of generality, assume that line 5 evaluated to true in iteration k as otherwise $Z_k = Z_{k-1}$ and $N_k = N_{k-1}$. If $C_{\mathrm{mono}}(N_{k-1} \cup S_k) - \Sigma(N_{k-1}) \leq C_{\mathrm{mono}}(S_k)$ then again

$$C_{Z_k}(N_k) = C_{\mathsf{ALG}}(N_k) \leq C_{\mathrm{mono}}(N_k) = \Sigma(N_k) \ .$$

Otherwise,

$$\begin{aligned} C_{Z_k}(N_k) &= C_{Z_{k-1} \oplus \mathsf{ALG}(I_{S_k})}(N_{k-1} \cup S_k) \\ &\leq C_{Z_{k-1}}(N_{k-1}) + C_{\mathsf{ALG}}(S_k) \\ &\leq \Sigma(N_{k-1}) + C_{\mathrm{mono}}(S_k) \\ &< C_{\mathrm{mono}}(N_{k-1} \cup S_k) \\ &= C_{\mathrm{mono}}(N_k) \ . \end{aligned}$$

Now by $\Sigma(N_{k-1}) + C_{\mathrm{mono}}(S_k) = \Sigma(N_k)$, the claim follows. Clearly, $Z = Z_m$, $Q = N_m$, and thus $C_Z(Q) \leq \Sigma(Q) \leq C_{\mathrm{mono}}(Q) \leq \beta \cdot C(Q)$. □

Theorem 6.30 (Corollary of Lemmata 6.27, 6.29, Theorems 6.10 and 6.21**).** *For an optimization problem Π, let $(I_S)_{S \subseteq [n]}$ be a mergable Π-cost-sharing problem having a β-relaxation $(\mathsf{ALG}, C_{\mathrm{mono}})$. Then the mechanism computed by Algorithm 6.6 is CGSP, β-BB, and $(2\beta \cdot H_n)$-EFF for C' induced by the solution computed by Algorithm 6.6. Moreover, Algorithm 6.6 evaluates C_{mono} for no more than $2n^2$ subsets of $[n]$, makes no more than n (direct) calls to ALG, and the number of merge operations is no more than n. (Note: C' not necessarily coincides with C_{ALG}.)*

6.8 Applications to Scheduling and Bin Packing

We use three approaches for obtaining β-relaxations that are polynomial-time computable: Monotonic approximation algorithms (Theorem 6.32 and Lemma 6.33), non-monotonic approximation algorithms with monotonic bounds C_{mono} (Theorems 6.34 and 6.35), and optimal costs that are monotonic and polynomial-time computable (discussed at the end of this section).

Note here that we assume that each player is given a unique number in $[n]$ in advance (outside the scope of Algorithm 6.6) and that players are sorted according to the respective monotonicity criterion.

Lemma 6.31. *Each* BinPacking *or* $(\mathrm{Q}||\mathrm{C}_{\max})$ *cost-sharing problem is mergable in time $O(n)$.*

Proof. For BinPacking with disjoint object/player sets T and U, we obtain a bin packing for $T \cup U$ by taking both the bins with objects from T and the bins with objects from U. The costs (number of bins) simply add up. For $(\mathrm{Q}||\mathrm{C}_{\max})$ and disjoint job/player sets T and U, we obtain a schedule for $T \cup U$ by assigning each job to the machine assigned before. The resulting makespan doesn't exceed the sum of the two makespans.

First, we consider $(\mathrm{P}||\mathrm{C}_{\max})$ cost-sharing problems:

Theorem 6.32. *For each* $(\mathrm{P}||\mathrm{C}_{\max})$ *cost-sharing problem it holds that $(\mathsf{LPT}, C_{\mathsf{LPT}})$ is a $\frac{4m-1}{3m}$-relaxation and Algorithm 6.6 runs in time $O(n^3 \cdot (\log n + \log m))$.*

Proof. We show that C_{LPT} is monotonic itself, i.e., for all $U \subseteq T \subseteq [n]$: $C_{\mathsf{LPT}}(\min_{|U|} T) \leq C_{\mathsf{LPT}}(U)$. Fix $\boldsymbol{p} \in \mathbb{N}^n$ and consider the cost-sharing problem $(\boldsymbol{p}, \mathbf{1})$. Consider $U, U' \subseteq [n]$ with $|U| = |U'|$ such that $\boldsymbol{p}_{U'}$ is obtained by reducing exactly one entry of $\boldsymbol{p}_U$ (and re-sorting). We show that $C_{\mathsf{LPT}}(U') \leq C_{\mathsf{LPT}}(U)$.

Let $U := \{u_1, \dots, u_{|U|}\}$ and $U' := \{u'_1, \dots, u'_{|U|}\}$ such that $u_1 < \dots < u_{|U|}$ and thus by Assumption 3.1, $p_{u_1} \geq \dots \geq p_{u_{|U|}}$; analogously for U'. For each $k \in [|U|]$, denote U_k and U'_k to be the first k elements in U and U', respectively. In addition, let ϕ and ϕ' the LPT assignments for U and U'. We will show by induction that

$$p(\{i \in U'_k \mid \phi'(i) = \phi'(u'_k)\}) \leq p(\{i \in U_k \mid \phi(i) = \phi(u_k)\}) \ ,$$

which immediately implies $C_{\mathsf{LPT}}(U') \leq C_{\mathsf{LPT}}(U)$. The case $k = 1$ is trivially satisfied by $p_{u'_1} \leq p_{u_1}$. For $k \to k+1$, by induction hypothesis, it holds that

$$\begin{aligned} p(\{i \in U'_k \mid \phi'(i) = \phi'(u'_{k+1})\}) &= \min_{h \in [m]} p(\{i \in U'_k \mid \phi'(i) = h\}) \\ &\leq \min_{h \in [m]} p(\{i \in U_k \mid \phi(i) = h\}) \\ &\leq p(\{i \in U_k \mid \phi(i) = \phi(u_{k+1})\}) \ . \end{aligned}$$

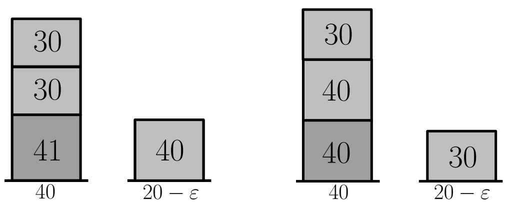
Thus,

$$\begin{aligned} p(\{i \in U'_{k+1} \mid \phi'(i) = \phi'(u'_{k+1})\}) &= p(\{i \in U'_k \mid \phi'(i) = \phi'(u'_{k+1})\}) + p_{u'_{k+1}} \\ &\leq p(\{i \in U_k \mid \phi(i) = \phi(u_{k+1})\}) + p_{u_{k+1}} \\ &= p(\{i \in U_{k+1} \mid \phi(i) = \phi(u_{k+1})\}) \ . \end{aligned}$$

□

Yet, LPT is not monotonic for general $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems. Consider $(\boldsymbol{p}, \boldsymbol{s})$ with $n = 5$, $m = 2$, $\boldsymbol{p} := (41, 40, 40, 30, 30)$, and speeds $\boldsymbol{s} := (40, 20 - \varepsilon)$ (scale $\boldsymbol{p}$ and $\boldsymbol{s}$ to guarantee $\boldsymbol{s} \in \mathbb{N}^m$). Let $U := \{1, 2, 4, 5\}$ and $U' := \{2, 3, 4, 5\}$. The corresponding costs are $C_{\mathsf{LPT}}(U) = 2.525$ and $C_{\mathsf{LPT}}(U') = 2.75$ (confer Figure 6.2).

Fig. 6.2. LPT is not monotonic in general



$\boldsymbol{p}_U = (41, 40, 30, 30)$ $\boldsymbol{p}_{U'} = (40, 40, 30, 30)$

The following 2-relaxations for BinPacking cost-sharing problems will be the key for obtaining 2-relaxations for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems:

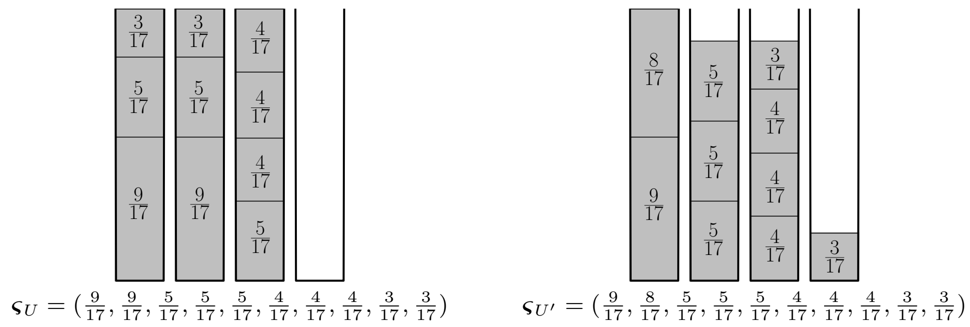
Lemma 6.33. *For each* BinPacking *cost-sharing problem, there is a 2-relaxation for which Algorithm 6.6 runs in time* $O(n^3 \cdot \log n)$.

Proof. Let RFFD denote the following algorithm: Given $\boldsymbol{\varsigma} \in (0,1]$, round each weight up to the next power of 2, i.e., let object i's rounded weight be $\varsigma'_i := 2^{\lceil \log_2 \varsigma_i \rceil}$ for all $i \in [n]$. Then, run the FFD which is known to produce an optimal packing for this modified instance $\boldsymbol{s}'$ ([27]). Clearly, RFFD is a 2-approximation algorithm running in time $O(n \cdot \log n)$. Since RFFD is optimal for the rounded sizes it is monotonic. Thus, $(\mathsf{RFFD}, C_{\mathsf{RFFD}})$ is a 2-relaxation that can be computed in time $O(n \cdot \log n)$.

We also remark that it is known that NFD is monotonic [100] and a 2-approximation algorithm for BinPacking. Hence, also $(\mathsf{NFD}, C_{\mathsf{NFD}})$ is a 2-relaxation. □

We note that FFD is not monotonic. For $n = 11$, define the cost-sharing problem $\boldsymbol{\varsigma} := (\frac{9}{17}, \frac{9}{17}, \frac{8}{17}, \frac{5}{17}, \frac{5}{17}, \frac{5}{17}, \frac{4}{17}, \frac{4}{17}, \frac{4}{17}, \frac{3}{17}, \frac{3}{17})$. Let $U := [11] \setminus 3$ and $U' := [11] \setminus \{1\}$. The respective costs are $C_{\mathsf{LPT}}(U) = 3$ and $C_{\mathsf{LPT}}(U') = 4$ (confer Figure 6.3).

Fig. 6.3. FFD is not monotonic in general



We obtain two different relaxations for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems, both based on applying binary search to the associated BinPacking decision problems. The first result from Theorem 6.34 is a more practicable one with respect to computability, obtained essentially by turning the RFFD algorithm from Lemma 6.33 into a decision procedure; the second result from Theorem 6.35 is obtained by adapting the PTAS for $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ optimization problems by Hochbaum and Shmoys [68], tweaking a bound computed inside the algorithm such that it becomes monotonic. For details, we refer to [11].

Theorem 6.34 ([11]). *For each* $(\mathrm{Q}||\mathrm{C}_{\max})$ *cost-sharing problem there is a 2-relaxation for which Algorithm 6.6 runs in time* $O(n^3 \cdot \log m \cdot \log \sum_{i \in [n]} p_i)$.

Theorem 6.35 ([11]). *For each* $(\mathrm{P}||\mathrm{C}_{\max})$ *cost-sharing problem, there is a* $(1+\varepsilon)$-*relaxation for which Algorithm 6.6 runs in time* $O(n^{2+\frac{1}{\varepsilon^2}} \cdot \log \sum_{i \in [n]} p_i)$.

There are several mergable scheduling problems for which optimal costs are monotonic and computable in polynomial time. For instance,

Lemma 6.36. *For each* $(\mathrm{Q}|p_i{=}1|\mathrm{C}_{\max})$ *cost-sharing problem,* $(\mathit{LPT}, C_{\mathit{LPT}})$ *is a 1-relaxation and Algorithm 6.6 runs in time* $O(n^3 \cdot \log m)$.

In the following, we give a selection of further scheduling problems, taken from Brucker [17].

We restrict our attention to the cases in which only one of the properties p_i, r_i and w_i is variable and let the others be fixed with $p_i := 1$, $w_i := 1$, and $r_i := 0$. We get that 1-relaxations exist for:

– Variable processing times: $(\mathrm{Q}|\mathrm{pmtn}|\mathrm{C}_{\max})$, $(\mathrm{Q}||\sum_i \mathrm{C}_i)$, and $(\mathrm{Q}|\mathrm{pmtn}|\sum_i \mathrm{C}_i)$
– Variable weights: $(\mathrm{P}||\sum_i w_i\mathrm{C}_i)$ and $(\mathrm{P}|\mathrm{pmtn}|\sum_i w_i\mathrm{C}_i)$
– Variable release dates: $(\mathrm{Q}|\mathrm{pmtn}, r_i|\mathrm{C}_{\max})$

The result for $(\mathrm{Q}||\sum_i \mathrm{C}_i)$ especially implies 1-BB for $(1||\sum_i \mathrm{C}_i)$. This is a drastic improvement over Moulin mechanisms, since by Brenner and Schäfer [16], no cross-monotonic cost-sharing method for $(1||\sum_i \mathrm{C}_i)$ cost-sharing problems can generally be better than $\frac{n+1}{2}$-BB.

6.9 Conclusion and Open Problems

The pivotal point of this chapter is our new modeling assumption on coalition formation. We believe that CGSP is a viable replacement for the often too limiting GSP requirement. Besides this novel structural property, we consider the main asset of our work to be threefold:

- Egalitarian mechanisms; showing existence of CGSP, 1-BB, and $2H_n$-EFF mechanisms for any non-decreasing subadditive cost function.
- Our framework for polynomial-time computation that reduces constructing CGSP, $O(1)$-BB, and $O(\log n)$-EFF mechanisms to finding monotonic approximation algorithms.
- Showing that acyclic mechanisms are CGSP and thus remarkably stronger than was known before.

An immediate issue left often by our work is, of course, to find more applications of our polynomial-time framework. For instance, it is easy to see that SteinerTree problems are mergable and their costs non-decreasing and subadditive; but do they allow for a β-relaxation?

7

Group-Strategyproof Mechanisms for General Demand

7.1 Contribution

The point of departure for this chapter is a rather obvious idea for generalizing Moulin mechanisms: Start with the maximum allocation and iteratively reduce service levels until every player can afford his remaining levels. The cost shares are extracted from marginal cost-sharing methods χ (confer Definition 2.28). We term these mechanisms $\mathit{MoulinGD}_{\chi}$.

- Whereas this idea of constructing a general demand cost-sharing mechanism has already been used by Mehta et al. [94], it was not known how GSP could be achieved. We identify three properties of marginal cost-sharing methods χ that are sufficient for $\mathit{MoulinGD}_{\chi}$ to be GSP. It comes as no surprise that a generalization of binary-demand cross-monotonicity is among them.
- We introduce natural mechanisms Level_{χ} that for each marginal cost-sharing method χ compute the same output as $\mathit{MoulinGD}_{\chi}$. We prefer to work with mechanisms Level_{χ} since they naturally give away service levels incrementally. In our opinion, this is much more intuitive and easier to handle.
- We give marginal cost-sharing methods χ^{FL} for every FaultTolerantFL cost-sharing problem and show that $\mathit{Level}_{\chi^{\text{FL}}}$ is GSP, $(3L)$-BB, $(3L \cdot (1 + H_n))$-EFF, and computable in polynomial time. This improves on the results of Mehta et al. [94], since a stronger notion of incentive-compatibility is guaranteed (GSP instead of WGSP) while reducing the budget-balance and efficiency approximations by a factor of L. The general idea is to tweak the binary demand cost-sharing mechanism for FacilityLocation by Pál and Tardos [106] in a clever way and then show that iteratively invoking the tweaked version works.
- For each GenSteinerForest cost-sharing problem, we give marginal cost-sharing methods χ^{GS} such that $\mathit{Level}_{\chi^{\text{GS}}}$ is GSP, $(2 \cdot H_L)$-BB, $O(\log^2 n \cdot \log L)$-EFF and computable in polynomial time. Here, the idea is to iteratively apply the binary demand cost-sharing mechanisms for SteinerForest cost-sharing problems proposed by Könemann et al [82]. Prior to this result, no GSP cost-sharing mechanisms for GenSteinerForest cost-sharing problems had been known.

The results presented in this chapter are published in [13].

7.2 Organization

We present mechanism $\mathit{MoulinGD}_\chi$ in Section 7.3. Specifically, the validity properties of marginal cost-sharing methods that guarantee that $\mathit{MoulinGD}_\chi$ is GSP are stated in Section 7.3.1. We introduce mechanism Level_χ in Section 7.3.2. The applications to FaultTolerantFL and GenSteinerForest cost-sharing problems are presented in Section 7.4 and Section 7.5.

7.3 Generalized Moulin Mechanisms

Given a marginal cost-sharing method χ, we propose to generalize Moulin mechanisms (confer Algorithm 4.1) as in Algorithm 7.1:

Algorithm 7.1 (computing $\mathit{MoulinGD}_\chi(\boldsymbol{B}) = (q(\boldsymbol{B}), x(\boldsymbol{B}))$).

1: $\boldsymbol{q} := (L_1, \ldots, L_n)$
2: **while** there exists i with $q_i > 0$ and $b_{i,q_i} < \chi_{i,q_i}(\boldsymbol{q})$ **do**
3: $\quad q_j := q_j - 1$ for an arbitrary j with $q_j > 0$ and $b_{j,q_j} < \chi_{j,q_j}(\boldsymbol{q})$
4: **return** $(\boldsymbol{q}, \boldsymbol{x})$ with $x_i := \sum_{\ell=1}^{q_i} \chi_{i,\ell}(\boldsymbol{q})$ for all $i \in [n]$

Clearly, if χ is β-BB then so is $\mathit{MoulinGD}_\chi$.

Note that in line 2, we only check if player i can pay for the current largest level q_i. However, the marginal cost-sharing methods χ that guarantee GSP of $\mathit{MoulinGD}_\chi$ imply that $\chi_{i,1}(\boldsymbol{q}) \leq \ldots \leq \chi_{i,q_i}(\boldsymbol{q})$ (see Lemma 7.5). As additionally $b_{i,1} \geq \ldots \geq b_{i,q_i}$ by Assumption 2.2, this check guarantees that all remaining levels can be paid for.

7.3.1 Validity of Marginal Cost-Sharing Methods

We now state the three properties of χ that are sufficient for $\mathit{MoulinGD}_\chi$ to be GSP. The first is a generalization of binary demand cross-monotonicity and states that the marginal cost share of a player for a certain service level can only decrease if the service levels of other players are increased:

Definition 7.1. *A marginal cost-sharing method χ is* cross-monotonic *if for all allocations $\boldsymbol{a} \in \mathcal{A}$, all players $i \in [n]$ and $j \in [n] \setminus \{i\}$ with $a_j < L_j$, and all service levels $\ell \in [L_i]$, it holds that $\chi_{i,\ell}(\boldsymbol{a}) \geq \chi_{i,\ell}(\boldsymbol{a} + \boldsymbol{e}_j)$.*

The second property ensures that the marginal cost-share $\chi_{i,\ell}(\boldsymbol{a})$ of player i with $a_i \geq \ell$ is exactly the marginal cost-share $\chi_{i,\ell}(\boldsymbol{a}^{\leq \ell})$ (confer Section 2.2), i.e., this marginal cost share for ℓ is independent of service levels larger than ℓ:

Definition 7.2. *A marginal cost-sharing method χ is* level-restricted *if for all allocations $\boldsymbol{a} \in \mathcal{A}$, for all players $i \in [n]$, and for all service levels $\ell \in [L_i]$, it holds that $\chi_{i,\ell}(\boldsymbol{a}) = \chi_{i,\ell}(\boldsymbol{a}^{\leq \ell})$.*

The third property together with cross-monotonicity and level-restriction implies that the marginal cost-share of a player is non-decreasing in the number of levels, as we show in Lemma 7.5.

Definition 7.3. *A marginal cost-sharing method χ is* non-decreasing *if for all allocations $\boldsymbol{a} \in \mathcal{A}$, for service level $\ell := \max_{i \in [n]}\{a_i\}$, and for all players $i \in [n]$ with $a_i = \ell < L_i$, it holds that $\chi_{i,\ell}(\boldsymbol{a}) \leq \chi_{i,\ell+1}(\boldsymbol{a} + \sum_{j \in [n]: a_j = \ell < L_j} \boldsymbol{e}_j)$.*

We merge the three properties into the central term *validity*:

Definition 7.4. *A marginal cost-sharing method χ is* valid *if it is level-restricted, cross-monotonic, and non-decreasing.*

Lemma 7.5. *If χ is valid, it holds for all allocations $\boldsymbol{a} \in \mathcal{A}$, for all service levels $\ell \in [L]$, and for all players $i \in [n]$ with $a_i > \ell$ that $\chi_{i,\ell}(\boldsymbol{a}) \leq \chi_{i,\ell+1}(\boldsymbol{a})$.*

Proof. Fix allocation $\boldsymbol{a}$, service level ℓ and player i with $a_i > \ell$. It is

$$\begin{aligned}
\chi_{i,\ell}(\boldsymbol{a}) &= \chi_{i,\ell}(\boldsymbol{a}^{\leq \ell}) && (\chi \text{ level restricted}) \\
&\leq \chi_{i,\ell+1}(\boldsymbol{a}^{\leq \ell} + \sum_{j \in [n]: \boldsymbol{a}_j^{\leq \ell} = \ell < L_j} \boldsymbol{e}_j) && (\chi \text{ non-decreasing}) \\
&\leq \chi_{i,\ell+1}(\boldsymbol{a}^{\leq \ell+1}) && (\chi \text{ cross-monotonic}) \\
&= \chi_{i,\ell+1}(\boldsymbol{a}) \ . && (\chi \text{ level-restricted})
\end{aligned}$$

□

Together with Assumption 2.2, Lemma 7.5 states that

$$b_{i,\ell+1} - \chi_{i,\ell+1}(\boldsymbol{a}) \leq b_{i,\ell} - \chi(i,\ell)(\boldsymbol{a}) \tag{7.1}$$

for all $\boldsymbol{B} \in \mathcal{R}$, all, $\boldsymbol{a} \in \mathcal{A}$, all $\ell \in [L]$, and all $i \in [n]$ with $a_i > \ell$. The results in this section heavily rely on this observation.

Lemma 7.6. *MoulinGD$_\chi$ meets NPT, VP and strict CS, given any valid marginal cost-sharing method χ.*

Proof. Fix $\boldsymbol{B} \in \mathcal{R}$. NPT holds by definition of χ. VP follows from the fact that $b_{i,q_i(\boldsymbol{B})} \geq \chi_{i,q_i(\boldsymbol{B})}(q(\boldsymbol{B}))$ and thus $\sum_{\ell=1}^{q_i(\boldsymbol{B})} b_{i,\ell} \geq \sum_{\ell=1}^{q_i(\boldsymbol{B})} \chi_{i,\ell}(q(\boldsymbol{B})) = x_i(\boldsymbol{B})$ by (7.1). To receive service level $\ell \in [L_i]_0$, player i may submit $\boldsymbol{b}_i$ with $b_{i,m} := M$ for $m \leq \ell$ and a sufficient large M and $b_{i,m} := -1$ otherwise. Thus, strict CS holds as well.

□

To gain insight on the importance of each of the three validity requirements, we first show that if exactly one of the validity properties required for χ does not hold, *MoulinGD$_\chi$* is *not* GSP. For all examples, let $n = 2$, $L_1 = L_2 = 2$, and $\chi_{2,\ell}(\boldsymbol{a}) := 2$ for all $\ell \in [2]$ and all $\boldsymbol{a} \in \mathcal{A}$ with $a_2 \geq \ell$. We always assume that $\boldsymbol{v}_2 = (2, 2)$.

Example 7.7. Consider χ with $\chi_{1,\ell}(\boldsymbol{a}) := 1$ for all $\ell \in [2]$ and all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 \geq \ell$, with the only exception that $\chi_{1,2}(2,2) := 2$. Obviously, χ is level-restricted and non-decreasing, but not cross-monotonic since $\chi_{1,2}(2,1) < \chi_{1,2}(2,2)$. For the case that $\boldsymbol{v}_1 = (2,2)$, both players get service level 2, where $u_1((\boldsymbol{v}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = 1$ and $u_2((\boldsymbol{v}_1, \boldsymbol{v}_2), \boldsymbol{v}_2) = 0$. Player 2 may then bid $\boldsymbol{b}_2 = (-1,-1)$ in order to not receive the service with the result that player 1 receives level 2 with utility $u_1((\boldsymbol{v}_1, \boldsymbol{b}_2), \boldsymbol{v}_1) = 2$.

Example 7.8. Consider method χ with $\chi_{1,1}(\boldsymbol{a}) := 1$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 = 1$, and $\chi_{1,1}(\boldsymbol{a}) := 2$ and $\chi_{1,2}(\boldsymbol{a}) := 3$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 = 2$. Here, χ is cross-monotonic and non-decreasing but fails to be level-restricted due to $\chi_{1,1}(2,2) > \chi_{1,1}(1,1)$. If now $\boldsymbol{v}_1 = (3,3)$, then player 1 receives level 2 and $u_1((\boldsymbol{v}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = 1$. However, for $\boldsymbol{b}_1 = (3,-1)$, player 1 receives only level 1, and $u_1((\boldsymbol{b}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = 2$.

On the other hand, we get $\chi_{1,1}(2,2) < \chi_{1,1}(1,1)$ when we change χ such that $\chi_{1,1}(\boldsymbol{a}) = 2$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 = 1$ and $\chi_{1,1}(\boldsymbol{a}) = 1$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 = 2$. If $\boldsymbol{v}_1 = (3, 3-\varepsilon)$, then player 1 receives only one level and $u_1((\boldsymbol{v}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = 1$. However, he may bid $\boldsymbol{b}_1 = (3,3)$ to receive both levels such that $u_1((\boldsymbol{b}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = 2 - \varepsilon$.

Example 7.9. Consider χ with $\chi_{1,1}(\boldsymbol{a}) := 2$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 \geq 1$ and $\chi_{1,2}(\boldsymbol{a}) := 1$ for all $\boldsymbol{a} \in \mathcal{A}$ with $a_1 \geq 2$. Now we have the case that χ is cross-monotonic and level-restricted, but not non-decreasing. For $\boldsymbol{v}_1 = (1,1)$, player 1 receives level 2 and has a utility of $u_1((\boldsymbol{v}_1, \boldsymbol{v}_2), \boldsymbol{v}_1) = -1$. However, bidding $\boldsymbol{b}_1 = (-1,-1)$ ensures a utility of zero.

7.3.2 Level Mechanisms

Given any valid marginal cost-sharing method χ, we define a general demand mechanism $Level_\chi$ in Algorithm 7.3 and show in Theorem 7.10 that it computes the same output as $MoulinGD_\chi$. Subsequently, Theorem 7.11 states that validity of χ implies GSP of $Level_\chi$. We consider the runtime of $Level_\chi$ in Lemma 7.12.

Although it is possible to prove GSP of $MoulinGD_\chi$ directly, we prefer this workaround for two reasons: whereas $MoulinGD_\chi$ iteratively decrements arbitrary service levels that cannot be paid for, $Level_\chi$ iteratively increases only service levels of those players that have been granted the previous level as well. We think that this is a much more intuitive way to determine an allocation. Furthermore, we may use results for binary demand Moulin mechanisms, as these are employed within $Level_\chi$.

For a consistent notation, we adjust cost-sharing methods ξ and $Moulin_\xi$ to work with vectors $\boldsymbol{p} \in \{0,1\}^n$ instead of sets. We give the adjusted mechanism $Moulin_\xi$ below:

Algorithm 7.2 (computing $Moulin_\xi(\boldsymbol{b}) = (q(\boldsymbol{b}), x(\boldsymbol{b}))$).

$\boldsymbol{p} := \boldsymbol{1}$;
while there exists i with $p_i = 1$ and $b_{i,1} < \xi_i(\boldsymbol{p})$ **do**
  $p_j := 0$ for an arbitrary j with $p_j = 1$ and $b_{j,1} < \xi_j(\boldsymbol{p})$
return $(\boldsymbol{p}, \xi(\boldsymbol{p}))$

Specifically, given a marginal cost-sharing method χ, $Level_\chi$ gives away service levels incrementally. In iteration ℓ, only players that were given service level $\ell - 1$ are considered as potential receivers of service level ℓ. In order to determine the actual receivers of level ℓ, when previously having computed allocation $\boldsymbol{q}$ with $q_i \leq \ell - 1$ for all i in iterations 1 to $\ell - 1$, we apply the binary demand Moulin mechanism $Moulin_\xi$ with a binary demand cost-sharing method $\xi := \chi^{\boldsymbol{q}} : \{0,1\}^n \to \mathbb{R}^n_{\geq 0}$ defined by

$$\chi_i^{\boldsymbol{q}}(\boldsymbol{p}) := \begin{cases} \chi_{i,q_i+1}(\boldsymbol{q} + \boldsymbol{p}) & \text{if } q_i < L_i \\ 0 & \text{otherwise} . \end{cases}$$

$Moulin_{\chi^{\boldsymbol{q}}}$ gets as an argument the bids $b_{i,\ell}$ from players i with $\ell - 1 = q_i < L_i$ for service level ℓ. For the other players, negative bids are simulated to ensure that they do not receive service level ℓ. Our new mechanism $Level_\chi$ is computed by Algorithm 7.3:

Algorithm 7.3 (computing $Level_\chi(\boldsymbol{B}) := (q(\boldsymbol{B}), x(\boldsymbol{B}))$).

$(\boldsymbol{q}, \boldsymbol{x}) := (\boldsymbol{0}, \boldsymbol{0})$; $\ell := 1$
$\boldsymbol{b}' := (b_{1,1}, \ldots, b_{n,1})$ ▷ bids for the first service level
repeat
  $(\boldsymbol{q}, \boldsymbol{x}) := (\boldsymbol{q}, \boldsymbol{x}) + Moulin_{\chi^{\boldsymbol{q}}}(\boldsymbol{b}')$
  $\ell := \ell + 1$; Define $\boldsymbol{b}'$ by $b'_i := \begin{cases} b_{i,\ell} & \text{if } \ell - 1 = q_i < L_i \\ -1 & \text{otherwise} \end{cases}$
until $b'_i = -1$ for all $i \in [n]$
return $(\boldsymbol{q}, \boldsymbol{x})$

The validity of χ has a very descriptive interpretation for $Level_\chi$. Cross-monotonicity implies that in each iteration, it can only be beneficial for all players if more players receive the current level. Level-restriction states that the marginal cost-share of a level due to which a player was selected in line 4 to receive this level remains fixed during further computation. Finally, if χ is non-decreasing, the marginal cost-shares for higher levels can only increase.

Theorem 7.10. *For each bid matrix $\boldsymbol{B} \in \mathcal{R}$ and each valid marginal cost-sharing method χ, $Level_\chi$ and $MoulinGD_\chi$ compute the same output.*

Proof. Fix a valid marginal cost-sharing method χ and a bid matrix $\boldsymbol{B} \in \mathcal{R}$. Let $\boldsymbol{q}^L$ and $\boldsymbol{x}^L$ be the allocation and the vector of cost-shares computed by $Level_\chi$, and $\boldsymbol{q}^M$ and $\boldsymbol{x}^M$ be the corresponding values computed by $MoulinGD_\chi$. We first show by induction that $(q^{\boldsymbol{L}})^{\leq \ell} = (q^{\boldsymbol{M}})^{\leq \ell}$ for all $\ell \in [L]$.

- $\ell = 1$
  Assume that $T := \{i \in [n] \mid (q_i^L)^{\leq 1} = 0 \text{ and } (q_i^M)^{\leq 1} = 1\} \neq \emptyset$. Consider the first iteration of $Level_\chi$. Within $Moulin_{\chi^{\boldsymbol{0}}}$, called by $Level_\chi$ in line 4, consider the first time that $Moulin_{\chi^{\boldsymbol{0}}}$ rejects a player $j \in T$, i.e., sets p_j to 0 in line 3. Let $\boldsymbol{p}$ denote the corresponding vector right before this event. The way that j and $\boldsymbol{p}$ were chosen, it holds that $p_j = 1$ and $(\boldsymbol{q}^M)^{\leq 1} \leq \boldsymbol{p}$. Hence, by validity of χ,

  $$b_{j,1} < \chi_j^{\boldsymbol{0}}(\boldsymbol{p}) = \chi_{j,1}(\boldsymbol{p}) \leq \chi_{j,1}((\boldsymbol{q}^M)^{\leq 1}) = \chi_{j,1}(q^M) \ .$$

  Using (7.1) we get that

  $$x_j^M = \sum_{k=1}^{q_j^M} \chi_{j,k}(q^M) > \sum_{k=1}^{q_j^M} b_{j,k} \ ,$$

  contradicting VP of $MoulinGD_\chi$.
  Similarly, it is $T' := \{i \in [n] \mid (q_i^L)^{\leq 1} = 1 \text{ and } (q_i^M)^{\leq 1} = 0\} = \emptyset$, by considering the first time that $MoulinGD_\chi$ sets q_j to 0 for a $j \in T'$ in line 3. Let $\boldsymbol{q}$ be the corresponding allocation vector right before this event. Here, it is $q_j = 1$ and $(\boldsymbol{q}^L)^{\leq 1} \leq \boldsymbol{q}$ and hence by validity of χ,

  $$b_{j,1} < \chi_{j,1}(\boldsymbol{q}) \leq \chi_{j,1}((\boldsymbol{q}^L)^{\leq 1}) = \chi_j^{\boldsymbol{0}}((\boldsymbol{q}^L)^{\leq 1}) \ .$$

  Then, however, in the first iteration of $Level_\chi$, $Moulin_{\chi^{\boldsymbol{0}}}$ would have rejected player j which contradicts $q_j^L \geq 1$.

- $\ell - 1 \to \ell$
  Assume that $T := \{i \in [n] \mid (q_i^L)^{\leq \ell} = \ell - 1 \text{ and } (q_i^M)^{\leq \ell} = \ell\} \neq \emptyset$. For the ℓ-th iteration of $Level_\chi$, consider the first time that $Moulin_\xi$ with $\xi = \chi^{(q^L)^{\leq \ell-1}}$ sets p_j to 0 for a $j \in T$. Let $\boldsymbol{p}$ be the corresponding vector right before this event. By the choice of $\boldsymbol{p}$, we have that $p_j = 1$ and $\boldsymbol{p} \geq (\boldsymbol{q}^M)^\ell$. With $(\boldsymbol{q}^L)^{\leq \ell-1} = (\boldsymbol{q}^M)^{\leq \ell-1}$ by induction assumption, it holds that $(\boldsymbol{q}^L)^{\leq \ell-1} + \boldsymbol{p} \geq (\boldsymbol{q}^M)^{\leq \ell-1} + (\boldsymbol{q}^M)^\ell = (\boldsymbol{q}^M)^{\leq \ell}$. Now, by (7.1) and validity of χ,

  $$\begin{aligned} b_{j,q_j^M} \leq b_{j,\ell} &< \chi_j^{(\boldsymbol{q}^L)^{\leq \ell-1}}(\boldsymbol{p}) = \chi_{j,\ell}((\boldsymbol{q}^L)^{\leq \ell-1} + \boldsymbol{p}) \\ &\leq \chi_{j,\ell}((\boldsymbol{q}^M)^{\leq \ell}) = \chi_{j,\ell}(\boldsymbol{q}^M) \\ &\leq \chi_{j,q_j^M}(\boldsymbol{q}^M) \ . \end{aligned}$$

However, this contradicts the termination condition of $MoulinGD_\chi$.

Now assume $T' := \{i \in [n] \mid (q_i^L)^{\leq \ell} = \ell \text{ and } (q_i^M)^{\leq \ell} = \ell - 1\} \neq \emptyset$. For $MoulinGD_\chi$, consider the first iteration that sets q_j to $\ell - 1$ for a $j \in T$. Let $\boldsymbol{q}$ be the corresponding allocation vector right before this event. Specifically, together with $(\boldsymbol{q}^L)^{\leq \ell - 1} = (\boldsymbol{q}^M)^{\leq \ell - 1}$ by induction assumption, it is $\boldsymbol{q} \geq (\boldsymbol{q}^L)^{\leq \ell}$. By (7.1) and validity of χ,

$$b_{j,q_j^L} \leq b_{j,\ell} < \chi_{j,\ell}(\boldsymbol{q}) \leq \chi_{j,\ell}((\boldsymbol{q}^L)^{\leq \ell}) = \chi_{j,\ell}(\boldsymbol{q}^L) \leq \chi_{j,q_j^L}(\boldsymbol{q}^L) ,$$

a contradiction to the fact that $Level_\chi$ has given service level q_j^L to j.

For the computed cost-shares, we have with $\boldsymbol{q} := \boldsymbol{q}^L = \boldsymbol{q}^M$ that for all $i \in [n]$

$$x_i^L = \sum_{\ell=1}^{q_i} \chi_{i,\ell}(\boldsymbol{q}^{\leq \ell}) = \sum_{\ell=1}^{q_i} \chi_{i,\ell}(\boldsymbol{q}) = x_i^M .$$

□

The proof of Theorem 7.11 uses the main idea from [99], which shows that $Moulin_\xi$ is GSP if ξ is cross-monotonic (confer the proof of Theorem 4.1).

Theorem 7.11. *For any valid marginal cost-sharing method χ, $Level_\chi$ is GSP.*

Proof. Assume that $Level_\chi$ is not GSP. Then, there exists $K \subseteq [n]$ with true valuations $\boldsymbol{V}_K = (\boldsymbol{v}_i)_{i \in K}$, and a bid matrix $\boldsymbol{B} \in \mathcal{R}$ such that $u_i(\boldsymbol{B}) \geq u_i(\boldsymbol{V}_K, \boldsymbol{B}_{-K})$ for all $i \in K$, with at least one strict inequality. Without loss of generality, let $\boldsymbol{V} := (\boldsymbol{V}_K, \boldsymbol{B}_{-K})$.

We first prove by induction that $q(\boldsymbol{B})^{\leq \ell} \leq q(\boldsymbol{V})^{\leq \ell}$ for all $\ell \in [L]$.

- $\ell = 1$
  Assume that $T := \{i \in [n] \mid q_i(\boldsymbol{B})^{\leq 1} = 1 \text{ and } q_i(\boldsymbol{V})^{\leq 1} = 0\} \neq \emptyset$. Consider the first iteration of $Level_\chi$ with input $\boldsymbol{V}$. Within $Moulin_{\chi^{\mathbf{0}}}$, called by $Level_\chi$ in line 4, consider the first time that $Moulin_{\chi^{\mathbf{0}}}$ rejects a player $j \in T$, i.e., sets p_j to 0 in line 3. Let $\boldsymbol{p}$ denote the corresponding vector right before this event. It is $p_j = 1$ and $\boldsymbol{p} \geq q(\boldsymbol{B})^{\leq 1}$. By cross-monotonicity,

  $$v_{j,1} < \chi_j^{\mathbf{0}}(\boldsymbol{p}) = \chi_{j,1}(\boldsymbol{p}) \leq \chi_{j,1}(q(\boldsymbol{B})^{\leq 1}) = \chi_{j,1}(q(\boldsymbol{B})) .$$

  Thus, $j \in K$. Furthermore, by (7.1) we get

  $$x_j(\boldsymbol{B}) = \sum_{k=1}^{q_i(\boldsymbol{B})} \chi_{j,k}(q(\boldsymbol{B})) > \sum_{k=1}^{q_i(\boldsymbol{B})} b_{j,k} .$$

  Hence, $u_j(\boldsymbol{V}, \boldsymbol{v}_j) = 0 > u_j(\boldsymbol{B}, \boldsymbol{v}_j)$ which contradicts $j \in K$.

- $\ell - 1 \to \ell$
  By induction hypothesis, for all players i with $q_i(\boldsymbol{B})^{\leq \ell} > q_i(\boldsymbol{V})^{\leq \ell}$ we have that $q_i(\boldsymbol{B})^{\leq \ell - 1} = q_i(\boldsymbol{V})^{\leq \ell - 1} = q_i(\boldsymbol{V})^{\leq \ell} = \ell - 1$ and $q_i(\boldsymbol{B})^{\leq \ell} = \ell$. Assume that $T := \{i \in [n] \mid q_i(\boldsymbol{B})^{\leq \ell} = \ell \text{ and } q_i(\boldsymbol{V})^{\leq \ell} = \ell - 1\} \neq \emptyset$ and obtain $j \in T$ and $\boldsymbol{p}$ as in case $\ell = 1$, for $Moulin_\xi$ with $\xi = \chi^{q(\boldsymbol{V})^{\leq \ell - 1}}$. It holds that $\boldsymbol{p} \geq (q_i(\boldsymbol{B}))^\ell$. By induction hypothesis and validity of χ,

$$\begin{aligned} v_{j,\ell} &< \chi_j^{q(\boldsymbol{V})^{\leq \ell-1}}(\boldsymbol{p}) = \chi_{j,\ell}(q(\boldsymbol{V})^{\leq \ell-1} + \boldsymbol{p}) \\ &\leq \chi_{j,\ell}(q(\boldsymbol{B})^{\leq \ell-1} + (q_i(\boldsymbol{B}))^{\ell}) = \chi_{j,\ell}(q(\boldsymbol{B})^{\leq \ell}) \ . \end{aligned}$$

Thus, $j \in K$. However, by (7.1), induction hypothesis, and $q_j(\boldsymbol{V}) = \ell - 1$, we get a contraction to $j \in K$, namely,

$$\begin{aligned} u_j(\boldsymbol{B}, \boldsymbol{v}_j) &= \sum_{k=1}^{q_j(\boldsymbol{B})} \Big(v_{j,k} - \chi_{j,k}(q(\boldsymbol{B})^{\leq k}) \Big) \\ &< \sum_{k=1}^{\ell-1} \Big(v_{j,k} - \chi_{j,k}(q(\boldsymbol{B})^{\leq k}) \Big) \\ &\leq \sum_{k=1}^{q_j(\boldsymbol{V})} \Big(v_{j,k} - \chi_{j,k}(q(\boldsymbol{V})^{\leq k}) \Big) = u_j(\boldsymbol{V}, \boldsymbol{v}_j) \ . \end{aligned}$$

As an immediate consequence of $q(\boldsymbol{V}) \geq q(\boldsymbol{B})$, we get a contraction to the assumption that $Level_\chi$ is not GSP, since no player strictly improves his utility for $\boldsymbol{B}$: For all $i \in [n]$, by cross-monotonicity,

$$\begin{aligned} u_i(\boldsymbol{B}, \boldsymbol{v}_i) &\leq \sum_{\ell=1}^{q_i(\boldsymbol{B})} \Big(v_{i,\ell} - \chi_{i,\ell}(q(\boldsymbol{V})^{\leq \ell}) \Big) \\ &\leq \sum_{\ell=1}^{q_i(\boldsymbol{V})} \Big(v_{i,\ell} - \chi_{i,\ell}(q(\boldsymbol{V})^{\leq \ell})) \Big) \\ &= u_i(\boldsymbol{V}, \boldsymbol{v}_i) \ . \end{aligned}$$

□

In order to determine the runtime of $Level_\chi$ for a specific χ, we introduce Lemma 7.12:

Lemma 7.12. *For each marginal cost-sharing method χ such that for every $\boldsymbol{a} \in \mathcal{A}$ and each $\ell \in [L]$, $\chi_{*,\ell}(\boldsymbol{a})$ is computable in time $O(t)$, $Level_\chi$ is computable in time $O(L \cdot n \cdot t)$.*

Proof. A call of $Moulin_\chi^{\boldsymbol{q}}$ in iteration $\ell \in [L]$ of $Level_\chi$ with the current allocation $\boldsymbol{q} \in [\ell-1]_0^n$ takes at most time $O(n \cdot t)$, as it has to eventually compute $\chi_{*,\ell}(\boldsymbol{q}+\boldsymbol{p})$ for n different values for $\boldsymbol{p} \in \{0,1\}$ ($\boldsymbol{p}$ simulates the set of served agents for this level). □

7.4 Applications to Fault Tolerant Facility Location

In the examples within this section, we define instances of FAULTTOLERANTFL via networks. Facilities are illustrated as houses, where the roofs are labeled with the opening cost. Players are circles labeled with the player's identity. Edges (i, j) are labeled with $d(i, j)$. If i and j are not directly linked, $d(i, j)$ is defined as the cost of a shortest path between i and j in the network.

Theorem 7.13 is the main theorem of this section:

Theorem 7.13. *For each* FAULTTOLERANTFL *cost-sharing problem there is a marginal cost-sharing method* χ^{FL} *and an approximation algorithm* ALG *such that* $Level_{\chi^{\text{FL}}}$ *is GSP,* $3L$*-BB, and* $3L\cdot(1+H_n)$*-EFF for* C_{ALG}*. Furthermore,* $Level_{\chi^{\text{FL}}}$ *is computable in time* $O(n^2 \cdot L \cdot F \cdot (n + \log |F|))$ *and* ALG *in time* $O(n \cdot |F| \cdot (n \cdot L + \log |F|)$*.*

Guideline of Proof: Fix a FAULTTOLERANTFL cost-sharing problem $(F, \boldsymbol{o}, d)$ (for the remaining Section 7.4). In order to proof Theorem 7.13, we define a valid marginal cost-sharing method χ^{FL} (together with an auxiliary method χ) in Section 7.4.1 and show that for every $\boldsymbol{a} \in \mathcal{A}$ and each $\ell \in [L]$, $\chi^{\text{FL}}_{*,\ell}(\boldsymbol{a})$ is computable in time $O(n \cdot |F| \cdot (n + \log |F|))$. Section 7.4.2 gives the approximation algorithm ALG and also discusses its polynomial-time computability. Section 7.4.3 then shows $3L$-BB and $3L \cdot (1 + H_n)$-EFF for C_{ALG}. Theorem 7.13 then follows directly from Theorem 7.11 and Lemma 7.12.

7.4.1 The Marginal Cost-Sharing Method

In this section, we explain how to define χ^{FL}. To this end, we define a marginal cost-sharing method χ and obtain χ^{FL} by multiplying χ by 3. We make this detour as χ is consistent with the methods by Pál and Tardos [106] and Mehta et al. [94] and allows for simpler comparisons. For $\boldsymbol{a} \in \{0, 1\}^n$, the computation of χ reduces to the method by Pál and Tardos [106] for binary demand facility location.

Fix $\boldsymbol{a} \in \mathcal{A}$ and $\ell \in [L]$. We only need to determine $\chi_{i,\ell}(\boldsymbol{a})$ for all players i with $i \in A_\ell := \{j \in [n] \mid a_j \geq \ell\}$. Simultaneously, every player i in A_ℓ uniformly grows a ball with i at its center to infinity. This ball, the *ghost* of i, has radius t at time t. We say that the ghost of i *touches* facility f at time t if $d(i, f) \leq t$. If the ghost of i touches f it starts *filling* f, contributing $t - d(i, f)$ at time $t \geq d(i, f)$. Facility f is said to be *full*, if all such contributions sum up to its opening cost o_f. Let $t(f)$ denote the time when f becomes full, and $S_f := \{i \in [n] \mid d(i, f) < t(f)\}$ the set of players that contributed to filling f. It holds that

$$\sum_{i \in S_f} (t(f) - d(i, f)) = o_f \; . \tag{7.2}$$

Definition 7.14. *For all* $\boldsymbol{a} \in \mathcal{A}$*, all* $\ell \in [L]$ *and all* $i \in A_\ell := \{i \in [n] \mid a_i \geq \ell\}$*, we define* $\chi_{i,\ell}(\boldsymbol{a})$ *to be the time that it takes for the ghost of* i *to touch* ℓ *full facilities when all players in the set* A_ℓ *grow their ghosts. For all other cases, we let* $\chi_{i,\ell}(\boldsymbol{a}) := 0$*. Furthermore, we let* $\chi^{\text{FL}}_{i,\ell}(\boldsymbol{a}) := 3 \cdot \chi_{i,\ell}(\boldsymbol{a})$ *for all* $\boldsymbol{a} \in \mathcal{A}$*, all* $i \in [n]$*, and all* $\ell \in [L_i]$*.*

Example 7.15. Given the instance in Figure 7.1, we look at allocation $\boldsymbol{a} = (2, 2, 2, 1)$ and explain how to compute $\chi(\boldsymbol{a})$. To denote certain events that occur during the ghost-growing process, we introduce some notation. For the event that i touches f, but f is not full yet, we write $i \circ f$. The event that f becomes full is denoted by $\mathbf{f}$, and the event that i touches a full facility f is written as $i \bullet \mathbf{f}$.

The marginal cost shares for level 1 are determined by growing the ghosts of player set $A_1 = \{1, 2, 3, 4\}$, where $\boldsymbol{a}^1 = (1, 1, 1, 1)$. For level 2, we grow the ghosts of player set $A_2 = \{1, 2, 3\}$, where $\boldsymbol{a}^2 = (1, 1, 1, 0)$. The events occuring at certain time steps t are illustrated in Figure 7.2. Note that $i \in S_f$ iff event $i \circ f$ occurs at a strictly smaller time step than event $i \bullet \mathbf{f}$. Thus, for level 1, $S_1 = \{1\}, S_2 = \{2\}$ and $S_3 = \{3, 4\}$. For level 2, $S_1 = \{1\}, S_2 = \{2\}, S_3 = \{3\}$ and $S_4 = \{1, 2\}$. The cost shares are $\chi_{*,1}(\boldsymbol{a}) = \chi_{*,1}(1, 1, 1, 1) = (2, 2, \frac{3}{2}, \frac{3}{2})$ and $\chi_{*,2}(\boldsymbol{a}) = \chi_{*,2}(2, 2, 2, 0) = (3, 3, 4, 0)$. The final cost shares are thus $(5, 5, \frac{11}{2}, \frac{3}{2})$.

Fig. 7.1. Example instance for FaultTolerantFL

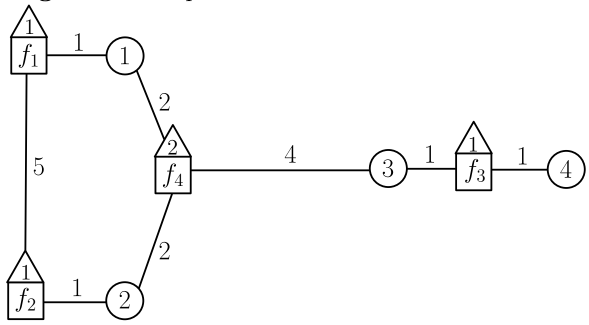


Fig. 7.2. Events during the computation of $\chi(2,2,2,1)$ for the instance in Figure 7.1

Level 1:

t	Event
1	$1 \circ f_1, 2 \circ f_2, 3 \circ f_3, 4 \circ f_3$
$\frac{3}{2}$	$\mathbf{f_3}, 3 \bullet \mathbf{f_3}, 4 \bullet \mathbf{f_3}$
2	$\mathbf{f_1}, \mathbf{f_2}$ $1 \bullet \mathbf{f_1}, 2 \bullet \mathbf{f_2}, 1 \circ f_4, 2 \circ f_4$

Level 2:

t	Event
1	$1 \circ f_1, 2 \circ f_2, 3 \circ f_3$
2	$\mathbf{f_1}, \mathbf{f_2}, \mathbf{f_3}$ $1 \bullet \mathbf{f_1}, 2 \bullet \mathbf{f_2}, 3 \bullet \mathbf{f_3}, 1 \circ f_4, 2 \circ f_4$
3	$\mathbf{f_4}, 1 \bullet \mathbf{f_4}, 2 \bullet \mathbf{f_4}$
4	$3 \bullet \mathbf{f_4}$

Note that $\chi_{*,\ell}(\boldsymbol{a})$ only depends on $A_\ell = \{j \in [n] \mid a_j \geq \ell\}$; it therefore holds that $\chi_{*,\ell}(\boldsymbol{a}) = \chi_{*,\ell}(\ell \cdot \boldsymbol{a}^\ell)$. This is even stronger than level-restriction.

Lemma 7.16. *χ and χ^{FL} are valid.*

Proof. We only show validity of χ. Validity of χ^{FL} obviously follows. By construction, χ is level-restricted. Cross-monotonicity is met, since facilities are filled faster for a larger set of players: For all allocations $\boldsymbol{a} \in \mathcal{A}$, all players $i \in [n]$ and $j \in [n] \setminus \{i\}$ with $a_j < L_j$, and all levels $\ell \in [L_i]$, it is

$$\chi_{i,\ell}(\boldsymbol{a}) = \chi_{i,\ell}(\ell \cdot \boldsymbol{a}^\ell) \geq \chi_{i,\ell}(\ell \cdot (\boldsymbol{a} + \boldsymbol{e}_j)^\ell) = \chi_{i,\ell}(\boldsymbol{a} + e_j) \ .$$

To show that χ is non-decreasing, observe that if the cost shares for level ℓ and level $\ell + 1$ are determined for the same set of players, then the cost shares for level $\ell + 1$ cannot be strictly smaller. Fix $\boldsymbol{a}$ and let $\ell := \max_i\{a_i\}$. For all $i \in [n]$ with $a_i = \ell < L_i$, it is

$$\begin{aligned}\chi_{i,\ell}(\boldsymbol{a}) = \chi_{i,\ell}(\ell \cdot \boldsymbol{a}^\ell) &\leq \chi_{i,\ell}(\sum_{\substack{j\in[n]:\\ a_j=\ell<L_j}} \ell \cdot \boldsymbol{e}_j) \leq \chi_{i,\ell+1}(\sum_{\substack{j\in[n]:\\ a_j=\ell<L_j}} (\ell+1) \cdot \boldsymbol{e}_j) \\ &= \chi_{i,\ell+1}((\ell+1) \cdot (\boldsymbol{a} + \sum_{\substack{j\in[n]:\\ a_j=\ell<L_j}} \boldsymbol{e}_j)^{(\ell+1)}) = \chi_{i,\ell+1}(\boldsymbol{a} + \sum_{\substack{j\in[n]:\\ a_j=\ell<L_j}} \boldsymbol{e}_j) \ .\end{aligned}$$

□

We now discuss computability. Fix $\boldsymbol{a} \in \mathcal{A}$ and $\ell \in [L]$. For each facility $f \in F$, we can determine $t(f)$ by ordering the players in $A_\ell = \{i \in [n] \mid a_i \geq \ell\}$ by non-decreasing distance to f. Assume without loss of generality that $A_\ell = \{1, \ldots, |A_\ell|\}$

and that $d(1,f) \leq \cdots \leq d(|A_\ell|,f)$. For $j \in [|A_\ell|]$, let $t_j(f)$ be the time that facility f gets full if only the ghosts in the set $\{1,\ldots,j\}$ grow their ghosts. It holds that $t_j(f) = d(j,f) + \frac{1}{j}\left(o_f - \sum_{k=1}^{j-1} k \cdot (d(k+1,f) - d(k,f))\right)$. Compute $t_j(f)$ iteratively until $j = |A_\ell|$ or $t_j(f) \leq d(j+1,f)$. Then $t(f) = t_j(f)$. Thus, the set $\{t(f)\}_{f\in F}$ can be computed in time $O(n^2 \cdot |F|)$.

Now, for a player $i \in A_\ell$, order the set $\{d(i,f), t(f)\}_{f\in F}$ by non-decreasing values. Let the corresponding ordered values be $s_1,\ldots,s_{2|F|}$. Find the smallest $j \in [2|F|]$ such that $|\{f \in F \mid d(i,f) \leq s_j \text{ and } t(f) \leq s_j\}| = \ell$. Then $\chi_{i,\ell}(\boldsymbol{a}) = s_j$. The vector $\chi_{*,\ell}(\boldsymbol{a})$ can thus be computed in time $O(n \cdot |F| \cdot (n + \log|F|))$.

7.4.2 The Approximate Solution

Given an allocation $\boldsymbol{a} \in \mathcal{A}$, we give an approximation algorithm ALG that constructs a solution with cost $C_{\mathsf{ALG}}(\boldsymbol{a})$. We decide which facilities to open during the iterative computation of $\chi_{i,\ell}(\boldsymbol{a})$ for all $i \in A_\ell = \{j \in [n] \mid a_j \geq \ell\}$ for $\ell = 1,\ldots,\max_i\{a_i\}$. Fix an iteration ℓ. Let $t(f)$ and S_f be the values obtained for all $f \in F$ by growing the ghosts of A_ℓ.

Facilities are opened in iteration ℓ according to the following rule: Let $F_{\ell-1}$ be the set of the already opened facilities in iterations $1,\ldots,\ell-1$. If in iteration ℓ, a facility $f \notin F_{\ell-1}$ becomes full, we open f if and only if conditions O_1 and O_2 hold:

O_1) There is no facility g that was already opened in iteration ℓ and for which $d(g,f) \leq 2 \cdot t(f)$.

O_2) There are no ℓ distinct facilities $g_1,\ldots,g_\ell \in F_{\ell-1}$ for which $d(g_k,f) \leq 2 \cdot t(f)$ for all $k \in [\ell]$.

If facilities become full at the same time, we break ties arbitrarily. Finally, we connect each player $i \in [n]$ to a_i distinct closest open facitities. It is straightforward to see that this solution is computable in time $O(n \cdot |F| \cdot (n \cdot L + \log|F|))$.

For the sake of simplifying the analysis, we specify other connection rules that may only increase the final cost: In each iteration ℓ, we connect every player $i \in A_\ell$ to one (more) open facility according to the following rules:

C_1) If $i \in S_f$ for an f opened in iteration ℓ, we connect i to f.

C_2) Otherwise, if at time $\chi_{i,\ell}(\boldsymbol{a})$ the ghost of i touches an open facility f to which i is not connected yet, we connect i to f.

C_3) Otherwise, let f be a full but closed facility that the ghost of i touches at time $\chi_{i,\ell}(\boldsymbol{a})$; f was not opened in iteration ℓ, because O_1 or O_2 do not hold:
  a) If O_1 does not hold because of facility g, connect i to g.
  b) If O_2 does not hold because of facilities $g_1,\ldots,g_\ell$, connect i to a facility $g \in \{g_1,\ldots,g_\ell\}$ to which i is not connected yet.

If there are ties in C_3a and C_3b, break them arbitrarily.

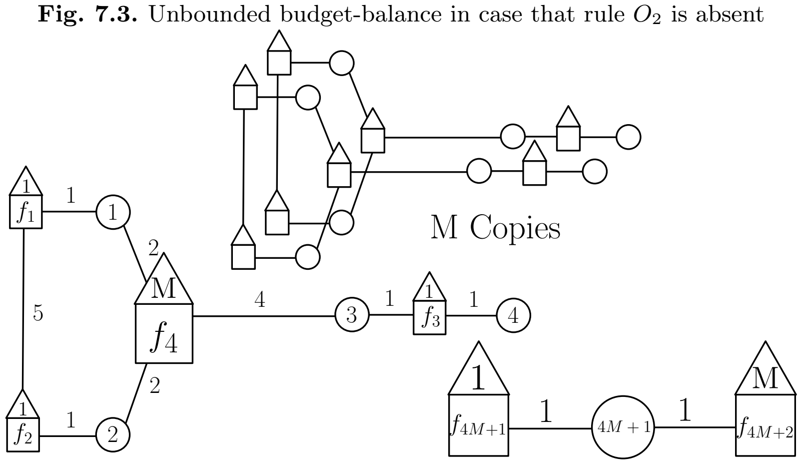
Example 7.17. We continue Example 7.15. Let us now determine which facilities to open and how to connect players to open facilities (for analysis). For Level 1, we open f_3, since there is no other already open facility. Then f_1 and f_2 become full. We open f_1, since $8 = d(f_1,f_3) > 2 \cdot t(f_1) = 4$. We also open f_2, since $d(f_2,f_3) = d(f_1,f_3) > 2 \cdot t(f_1) = 2 \cdot t(f_2)$ and $5 = d(f_2,f_1) > 2 \cdot t(f_2) = 4$. Due to C_1, we connect players 3 and 4 to f_3, player 1 to f_1, and player 2 to f_2. For Level 2, the first (and only) facility that becomes full and is not opened yet is facility f_4 with $t(f_4) = 3$.

However, $3 = d(f_4, f_i) \leq 2 \cdot t(f_4) = 6$ for $i \in \{1, 2\}$. Thus, f_4 stays closed due to O_2. All players in $\{1, 2, 3\}$ are connected due to C_3b, i.e., 1 is connected to f_2, 2 is connected to f_1, and 3 is connected to f_1 or f_2.

Deleting O_2 and C_3b, we get the opening and connection rules of [106]. However, Example 7.18 shows that rule O_2 is crucial for a reasonable BB approximation:

Example 7.18. We continue Example 7.15 and Example 7.17. Our constructed solution has cost 24. The optimum solution has cost 17, with all facilities being opened. So what is the reason that opening rule O_2 forbids to open f_4? Consider the instance in Figure 7.18 which consist of M copies of the above instance, where o_{f_4} is replaced by M. We assume that these copies are sufficiently far away from each other. To ensure that f_4 (and the corresponding facilities in the copies) is filled before all cost shares are determined, we introduce an additional construct (also sufficiently far away from the others), where player $4M + 1$ has to fill facilities f_{4M+1} with $o_{f_{4M+1}} = 1$ and f_{4M+2} with $o_{f_{4M+2}} = M$, when his cost-share for level 2 is computed. Consider $\boldsymbol{a}$ with $a_i = 1$ if $i \in \{4, 8, \ldots, 4M\}$ and $a_i = 2$ otherwise. If we would open f_4 and the corresponding facilities in the copies in the second iteration, the cost of the constructed solution is larger than $M \cdot (M + 1)$. However, the optimum cost for this instance is $24 \cdot M + M + 3$. Thus, without rule O_2, the BB factor would be unbounded.

Fig. 7.3. Unbounded budget-balance in case that rule O_2 is absent



7.4.3 Budget-Balance and Efficiency

Theorem 7.19. *χ^{FL} is $(3L)$-BB for C_{ALG}.*

Proof. We show for χ, that for any $\boldsymbol{a} \in \mathcal{A}$ and $X(\boldsymbol{a}) := \sum_{\ell=1}^{L} \sum_{i \in A_\ell} \chi_{i,\ell}(\boldsymbol{a})$, it holds that $\frac{1}{3} \cdot C_{\mathsf{ALG}}(\boldsymbol{a}) \leq X(\boldsymbol{a}) \leq L \cdot \mathsf{FTFL}(\boldsymbol{a})$.

Fix $\boldsymbol{a} \in \mathcal{A}$. We first show the upper bound. Consider an arbitrary facility set $F' \subseteq F$ with $|F'| \geq \max_i\{a_i\}$. Fix $\ell \in [\max_i\{a_i\}]$ and $i \in A_\ell := \{i \in [n] \mid a_i \geq l\}$. Let $t(f)$ and S_f be the values obtained for all $f \in F$ by growing the ghosts of A_ℓ. Let $F'_i \subseteq F'$ be an arbitrary set of a_i distinct closest facilities in F' to i. We show:

$$\exists f \in F'_i : \chi_{i,\ell}(\boldsymbol{a}) \leq \begin{cases} t(f) & \text{if } i \in S_f \\ d(i, f) & \text{otherwise .} \end{cases} \tag{7.3}$$

Assume that (7.3) does not hold. Then for all $f \in F'_i$, $\chi_{i,\ell}(\boldsymbol{a}) > t(f) > d(i,f)$ if $i \in S_f$ and $\chi_{i,\ell}(\boldsymbol{a}) > d(i,f) \geq t(f)$ otherwise. Thus, at time $t := \max_{f \in F'_i}\{t(f), d(i,f)\}$, the ghost of i touches at least $a_i \geq \ell$ full facilities, a contradiction to $t < \chi_{i,\ell}(\boldsymbol{a})$. Note that (7.3) especially holds for $F' = F^*$, when F^* is an optimal facility set for $\boldsymbol{a}$. Then,

$$\begin{aligned}
\sum_{i \in A_\ell} \chi_{i,\ell}(\boldsymbol{a}) &\leq \sum_{i \in [n]} \left(\sum_{f \in F^*_i : i \in S_f} t(f) + \sum_{f \in F^*_i : i \notin S_f} d(i,f) \right) \\
&= \sum_{f \in F^*} \sum_{i \in S_f : f \in F^*_i} (t(f) - d(i,f)) + \sum_{i \in [n]} \sum_{f \in F^*_i} d(i,f) \\
&\leq \sum_{f \in F^*} o_f + \sum_{i \in [n]} \sum_{f \in F^*_i} d(i,f) = \mathsf{FTFL}(\boldsymbol{a}) \ .
\end{aligned}$$

For the last inequality, confer (7.2). Thus, $X(\boldsymbol{a}) = \sum_{\ell=1}^{L} \sum_{i \in A_\ell} \chi_{i,\ell}(\boldsymbol{a}) \leq L \cdot \mathsf{FTFL}(\boldsymbol{a})$.

Now we show the lower bound, i.e., that $\frac{1}{3} \cdot C_{\mathsf{ALG}}(\boldsymbol{a}) \leq X(\boldsymbol{a})$. Fix an iteration ℓ. Let F_ℓ be the newly opened facilities in ℓ, and $F_{\ell-1}$ be the facilities opened in iterations $1 \ldots, \ell-1$. We show that the marginal cost shares of A_ℓ for level ℓ recover at least $\frac{1}{3}$ of the new opening and connection costs. Due to rule O_1, for $\{g, f\} \subseteq F_\ell$, it holds that $S_g \cap S_f = \emptyset$, as shown in [106].

- Each player set S_f for an $f \in F_\ell$ pays at least $\frac{1}{3}$ of $o_f + \sum_{i \in S_f} d(i,f)$:
  Due to C_1, all $i \in S_f$ for an $f \in F_\ell$ are connected to f. By (7.2), it holds that $o_f = \sum_{i \in S_f}(t(f) - d(i,f))$ and thus $o_f + \sum_{i \in S_f} d(i,f) = |S_f| \cdot t(f)$. We show that for all $i \in S_f$ it holds that $\chi_{i,\ell}(\boldsymbol{a}) \geq \frac{1}{3} \cdot t(f)$.
  Assume otherwise and consider $i \in S_f$ with $\chi_{i,\ell}(\boldsymbol{a}) < \frac{1}{3} \cdot t(f)$. Let $\{f_1, \ldots, f_\ell\}$ be the full facilities that i touches at time $\chi_{i,\ell}(\boldsymbol{a})$. For all $m \in [\ell]$, $d(f_m, f) \leq d(f_m, i) + d(i,f) \leq 2 \cdot \chi_{i,\ell}(\boldsymbol{a}) < \frac{2}{3} \cdot t(f) < 2 \cdot t(f)$. If all facilities in $\{f_1, \ldots, f_\ell\}$ are in $F_{\ell-1}$, we get a contradiction to $f \in F_\ell$ by O_2. Thus, there exists $f' \in \{f_1, \ldots, f_\ell\}$ with $f' \notin F_{\ell-1}$. By $d(f', f) < 2 \cdot t(f)$ and O_1, f' is not open at time $t(f)$ (but full, since $t(f') \leq \chi_{i,\ell}(\boldsymbol{a}) < t(f)$). Facility f' was not opened at $t(f')$, since one of the following holds:
  - There is a facility g that was already opened in ℓ and $d(g, f') \leq 2 \cdot t(f')$. This contradicts $f \in F_\ell$, since $d(f,g) \leq d(f,i) + d(i,f') + d(f',g) < t(f) + \chi_{i,\ell}(\boldsymbol{a}) + 2 \cdot t(f') \leq t(f) + 3 \cdot \chi_{i,\ell}(\boldsymbol{a}) < 2 \cdot t(f)$.
  - There are facilities $\{g_1, \ldots, g_\ell\} \subseteq F_{\ell-1}$ with $d(g_m, f') \leq 2 \cdot t(f')$ for all $m \in [\ell]$. Again, for all $m \in [\ell]$, $d(f, g_m) < 2 \cdot t(f)$, contradicting $f \in F_\ell$.
- Each player not in S_f for $f \in F_\ell$ pays at least $\frac{1}{3}$ of his connection cost:
  If i is connected to $g \in F_{\ell-1} \cup F_\ell$ due to C_2, $\chi_{i,\ell}(\boldsymbol{a}) \geq d(i,g)$. If it is connected due to C_3a or C_3b, it is $d(i,g) \leq d(i,f) + d(f,g) \leq \chi_{i,\ell}(\boldsymbol{a}) + 2 \cdot t(f) \leq 3 \cdot \chi_{i,\ell}(\boldsymbol{a})$ in both cases.

□

Theorem 7.20. *Level*$_{\chi^{FL}}$ *is* $(3L \cdot (1 + H_n))$*-EFF for* C_{ALG}.

We first show a property of χ similar to (binary demand) summability in Lemma 7.21, which constitutes the main part of the proof.

Lemma 7.21. *For any $\boldsymbol{a} \in \mathcal{A}$ and any ordering $s_1, \ldots, s_{|S|}$ of $S := \{i \in [n] \mid a_i > 0\}$ with $\boldsymbol{s}^j := \mathit{in}(\{s_1, \ldots, s_j\})$ for all $j \in [|S|]$, it is*

$$\sum_{j=1}^{|S|} \chi_{s_j, a_{s_j}}(a_{s_j} \cdot \boldsymbol{s}^j) \le H_n \cdot \mathit{FTFL}(\boldsymbol{a}) \ .$$

Proof. Roughly speaking, given $\boldsymbol{a} \in \mathcal{A}$, the main idea of the proof is a 'reduction' to the summability of a (binary demand) cost-sharing method ξ that we define according to Pál and Tardos [106] for a (binary demand) FacilityLocation cost-sharing problem $(G, \boldsymbol{o}, e)$ derived from $(F, \boldsymbol{o}, d)$ and an optimal solution for $\boldsymbol{a}$.

Fix $\boldsymbol{a} \in \mathcal{A}$ and an ordering $s_1, \ldots, s_{|S|}$ of $S := \{i \in [n] \mid a_i > 0\}$. Fix $j \in [|S|]$ and look at $\chi_{s_j, a_{s_j}}(a_{s_j} \cdot \boldsymbol{s}^j)$, computed for the original instance. For all $f \in F$, let $t(f)$ and S_f be the corresponding values for growing the ghosts of set $\{s_1, \ldots, s_j\}$. In the original problem, let F^* be an optimal facility set for $\boldsymbol{a}$, and $F^*_{s_j}$ be the facilities that s_j is connected to in an optimal solution. It is $F^* \subseteq \mathcal{F}_{\max_i\{a_i\}}$. In the proof of Theorem 7.19, we have already shown that there exists $g_j \in F^*_{s_j}$, such that $\chi_{s_j, a_{s_j}}(a_{s_j} \cdot \boldsymbol{s}^j)$ is at most $t(g_j)$ if $i \in S_{g_j}$, or $d(s_j, g_j)$ otherwise.

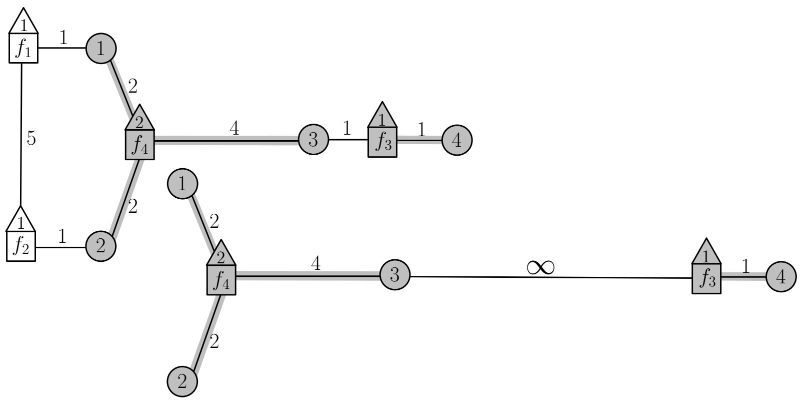
Let the new facility set be $G := \{g_1, \ldots, g_{|S|}\}$. For a each pair in $\{(s_j, g_j)\}_{j \in [|S|]}$, let $e(s_j, g_j) := d(s_j, g_j)$. Furthermore, for all j, j' such that $g_j = g_{j'}$, let $e(s_j, s_{j'}) := d(s_j, s_{j'})$. All other distances are defined to be sufficiently large, while ensuring that e is a metric. For an illustration, see Example 7.22.

By construction of the new problem, it is $\chi_{s_j, a_{s_j}}(a_{s_j} \cdot \boldsymbol{s}^j) \le \xi_{s_j}(\boldsymbol{s}^j)$ for all $j \in [|S|]$, where $\xi_{s_j}(\boldsymbol{s}^j)$ is computed on the new instance. Additionally, $\mathsf{FL}(S) \le \mathsf{FTFL}(\boldsymbol{a})$. We further use the fact that ξ is H_n-SUM [115] in order to obtain

$$\sum_{j=1}^{|S|} \chi_{s_j, a_{s_j}}(a_{s_j} \cdot \boldsymbol{s}^j) \le \sum_{j=1}^{|S|} \xi_{s_j}(\boldsymbol{s}^j) \le H_n \cdot \mathsf{FL}(S) \le H_n \cdot \mathsf{FTFL}(\boldsymbol{a}) \ .$$

□

Example 7.22. Let $\boldsymbol{a} = (2, 2, 2, 1)$ and look at $S = \{1, 2, 3, 4\}$ for the upper network below.



The new problem defined in Lemma 7.21 is illustrated in the lower network, where gray parts correspond to the unchanged distances. Particular values are:
- $\chi_{1,a_1}(a_1 \cdot \boldsymbol{s}^1) = \chi_{1,2}(2 \cdot (1,0,0,0)) = 4$, $\chi_{2,a_2}(a_2 \cdot \boldsymbol{s}^2) = \chi_{2,2}(2 \cdot (1,1,0,0)) = 3$,
- $\chi_{3,a_3}(a_3 \cdot \boldsymbol{s}^3) = \chi_{3,2}(2 \cdot (1,1,1,0)) = 4$, $\chi_{4,a_4}(a_4 \cdot \boldsymbol{s}^4) = \chi_{4,1}(1 \cdot (1,1,1,1)) = 1.5$,
- $F^* = \{f_1, f_2, f_3, f_4\}$, $F^*_1 = \{f_1, f_3\}, F^*_2 = \{f_2, f_3\},$, $F^*_3 = \{f_3, f_4\}, F^*_4 = \{f_4\}$,
- $g_1 = g_2 = g_3 = f_3, g_4 = f_4$.

Proof of Theorem 7.20:

Fix $\boldsymbol{V} \in \mathcal{R}$ and let $\boldsymbol{q} := q(\boldsymbol{V})$. Let $\boldsymbol{a} \in \mathcal{A}$ be a service vector with optimal social cost. We have to upper bound the value:

$$\gamma := \frac{SC_{C_{\mathsf{ALG}}}(\boldsymbol{q}, \boldsymbol{V})}{SC_{\mathsf{FTFL}}(\boldsymbol{a}, \boldsymbol{V})} = \frac{C_{\mathsf{ALG}}(\boldsymbol{q}) + \sum_{i=1}^{n} \sum_{\ell=q_i+1}^{L_i} \max\{0, v_{i,\ell}\}}{\mathsf{FTFL}(\boldsymbol{a}) + \sum_{i=1}^{n} \sum_{\ell=a_i+1}^{L_i} \max\{0, v_{i,\ell}\}} \ .$$

Let $\boldsymbol{r} \in \mathcal{A}$ with $r_i := \min\{q_i, a_i\}$ for all $i \in [n]$. Let $R_\ell := \{i \in [n] \mid r_i \geq \ell\}$ and $Q_\ell := \{i \in [n] \mid q_i \geq \ell\}$ for all $\ell \in [L]$. Let $\boldsymbol{r}^\ell := \mathsf{in}(R_\ell)$ and $\boldsymbol{q}^\ell := \mathsf{in}(Q_\ell)$. Obviously, $R_\ell \subseteq Q_\ell$, i.e., $\boldsymbol{r}^\ell \leq \boldsymbol{q}^\ell$. We have that

$$\begin{aligned}
X^{\text{FL}}(\boldsymbol{q}) := &\sum_{i=1}^{n} \sum_{\ell=1}^{r_i} \chi_{i,\ell}^{\text{FL}}(\ell \cdot \boldsymbol{q}^\ell) + \sum_{i=1}^{n} \sum_{\ell=r_i+1}^{q_i} \chi_{i,\ell}^{\text{FL}}(\ell \cdot \boldsymbol{q}^\ell) \\
\leq &\sum_{i=1}^{n} \sum_{\ell=1}^{r_i} \chi_{i,\ell}^{\text{FL}}(\ell \cdot \boldsymbol{r}^\ell) + \sum_{i=1}^{n} \sum_{\ell=r_i+1}^{q_i} \chi_{i,\ell}^{\text{FL}}(\ell \cdot \boldsymbol{q}^\ell) \\
\leq &\ 3L \cdot \mathsf{FTFL}(\boldsymbol{r}) + \sum_{i=1}^{n} \sum_{\ell=r_i+1}^{q_i} \chi_{i,\ell}^{\text{FL}}(\ell \cdot \boldsymbol{q}^\ell) \\
\leq &\ 3L \cdot \mathsf{FTFL}(\boldsymbol{a}) + \sum_{i=1}^{n} \sum_{\ell=r_i+1}^{q_i} v_{i,\ell} \ .
\end{aligned}$$

With $C_{\mathsf{ALG}}(\boldsymbol{q}) \leq X^{\text{FL}}(\boldsymbol{q})$, we get

$$\begin{aligned}
\gamma &\leq \frac{3 \cdot L \cdot \mathsf{FTFL}(\boldsymbol{a}) + \sum_{i=1}^{n} \sum_{\ell=r_i+1}^{q_i} v_{i,\ell} + \sum_{i=1}^{n} \sum_{\ell=q_i+1}^{L_i} \max\{0, v_{i,\ell}\}}{\mathsf{FTFL}(\boldsymbol{a}) + \sum_{i=1}^{n} \sum_{\ell=a_i+1}^{L_i} \max\{0, v_{i,\ell}\}} \\
&\leq 3 \cdot L + \frac{\sum_{i:a_i>q_i} \sum_{\ell=q_i+1}^{a_i} \max\{0, v_{i,\ell}\}}{\mathsf{FTFL}(\boldsymbol{a})} \ .
\end{aligned}$$

Here, the last inequality follows because the fraction is at least 1, so that we get another upper bound when subtracting $\sum_{i=1}^{n} \sum_{\ell=a_i+1}^{L_i} \max\{0, v_{i,\ell}\}$ in both the numerator and the denominator.

We finally show that $\sum_{i:a_i>q_i} \sum_{\ell=q_i+1}^{a_i} \max\{0, v_{i,\ell}\} \leq 3 \cdot L \cdot H_n \cdot \mathsf{FTFL}(\boldsymbol{a})$. Consider the players in $S := \{i \in [n] \mid a_i > q_i\}$ in the reverse order in which they are deleted by $Level_{\chi^{\text{FL}}}$ within the Moulin mechanism in line 4. Let this order be $s_1, \ldots, s_{|S|}$. For $j \in [|S|]$, let $\boldsymbol{p}^j \in \{0,1\}^n$ be the vector right before $p_{s_j}^j$ is set to 0 by the Moulin mechanism. Let $\boldsymbol{s}^j := \mathsf{in}(\{s_1, \ldots, s_j\})$. By Assumption 2.2 and a pretty rough estimate, we get

$$\sum_{i:a_i>q_i} \sum_{\ell=q_i+1}^{a_i} \max\{0, v_{i,\ell}\} \leq L \cdot \sum_{i:a_i>q_i} \max\{0, v_{i,q_i+1}\} \ .$$

For player $s_j \in S$ at position j, let $\boldsymbol{q}' \in [q_{s_j}]^n$ be the vector of service levels already determined by $Level_{\chi^{\text{FL}}}$ when line 4 is invoked and in this call of $Moulin_\xi$ with $\xi = (\chi^{\text{FL}})^{\boldsymbol{q}'}$, s_j is rejected due to $v_{s_j, q'_{s_j}+1} < (\chi^{\text{FL}})_{s_j}^{\boldsymbol{q}'}(\boldsymbol{p}^j)$. It is $q'_{s_j} = q_{s_j}$. By definition,

$$(\chi^{\text{FL}})_{s_j}^{\boldsymbol{q}'}(\boldsymbol{p}^j) = \chi_{s_j, q_{s_j}+1}^{\text{FL}}(\boldsymbol{q}' + \boldsymbol{p}^j) \ .$$

Furthermore, by construction,

$$\chi^{\text{FL}}_{s_j, q_{s_j}+1}(\boldsymbol{q}' + \boldsymbol{p}^j) = \chi^{\text{FL}}_{s_j, q_{s_j}+1}((q_{s_j} + 1) \cdot \boldsymbol{p}^j) \ .$$

Using cross-monotonicity and the fact that $\boldsymbol{p}^j \geq \boldsymbol{s}^j$ yields

$$\begin{aligned} L \cdot \sum_{i: a_i > q_i} \max\{0, v_{i,q_i+1}\} &< L \cdot \sum_{j=1}^{|S|} \chi^{\text{FL}}_{s_j, q_{s_j}+1}((q_{s_j} + 1) \cdot \boldsymbol{p}^j) \\ &\leq L \cdot \sum_{j=1}^{|S|} \chi^{\text{FL}}_{s_j, q_{s_j}+1}((q_{s_j} + 1) \cdot \boldsymbol{s}^j) \end{aligned}$$

We define $\boldsymbol{t}$ by $t_i := q_i + 1$ for $i \in S$ and 0 otherwise. By Lemma 7.21,

$$\begin{aligned} L \cdot \sum_{j=1}^{|S|} \chi^{\text{FL}}_{s_j, q_{s_j}+1}((q_{s_j} + 1) \cdot \boldsymbol{s}^j) &= 3 \cdot L \cdot \sum_{j=1}^{|S|} \chi_{s_j, q_{s_j}+1}((q_{s_j} + 1) \cdot \boldsymbol{s}^j) \\ &\leq 3 \cdot L \cdot H_n \cdot \mathsf{FTFL}(\boldsymbol{t}) \\ &\leq 3 \cdot L \cdot H_n \cdot \mathsf{FTFL}(\boldsymbol{a}) \ . \end{aligned}$$

□

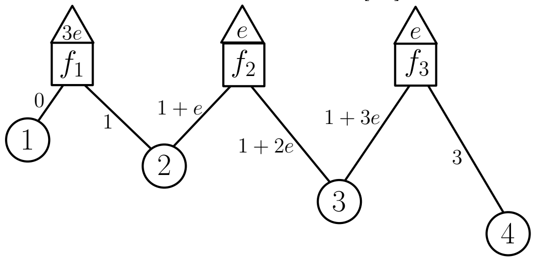
7.4.4 Comparison to the Method of Mehta et al.

We restate an example from Mehta et al. [94], showing that the cost shares of their acyclic mechanisms are in general not cross-monotonic. The main difference to χ (and χ^{FL}, respectively) is that $\chi_{i,\ell}(\boldsymbol{a})$ is independent of connections computed in iterations 1 to $\ell - 1$.

We shortly describe the marginal cost-sharing method χ^{M} used in the acyclic mechanism introduced by Mehta et al. [94]. The mechanism itself is essentially Mechanism *MoulinGD*$_\chi$ from Algorithm 7.1, where line 2 is replaced by 'while there exists i with $b_{i,\ell} < \chi_{i,\ell}(\boldsymbol{q})$ for an $\ell \in [q_i]$'.

The marginal cost shares $\chi^{\text{M}}_{i,\ell}(\boldsymbol{a})$ for all $i \in A_\ell$ are computed iteratively for $\ell = 1, \ldots, L$. In each iteration ℓ, the cross-monotonic cost-sharing method of Pál and Tardos [106] is invoked for all players with $a_i \geq \ell$. The instance is changed in such a way that opening costs are set to 0 for already opened facilities. In order to ensure that each player i is connected to a_i distinct facilities, the distance $d(i, f)$ for already existing connections between i and f is set to infinity. All other distances stay the same. This destroys the metric property of d and, according to Mehta et al. [94], complicates the analysis. We give the example from [94] which shows that this marginal cost-sharing method is not cross-monotonic. In contrast, we compute the cross-monotonic cost shares introduced in this work for this example. The instance is given in Figure 7.4. We first consider the method from Mehta et al. [94]. For $\boldsymbol{a} = (2, 2, 2, 2)$, the cost shares for the first level are $\chi^{\text{M}}_{*,1}((2, 2, 2, 2)) = (3\varepsilon, 1, 1 + 2\varepsilon, 3)$, and for the second level, $\chi^{\text{M}}_{*,2}((2, 2, 2, 2)) = (2 + \varepsilon, 1 + 2\varepsilon, 1 + 2\varepsilon, 5 + 5\varepsilon)$. For $\boldsymbol{a} = (0, 2, 2, 2)$, we get $\chi^{\text{M}}_{*,1}((0, 2, 2, 2)) = (0, 1+2\varepsilon, 1+2\varepsilon, 3)$, and $\chi^{\text{M}}_{*,2}((0, 2, 2, 2)) = (0, 1+3\varepsilon, 1+3\varepsilon, 3)$. Cross-monotonicity is violated by $\chi^{\text{M}}_{4,2}((2, 2, 2, 2)) = 5 + 5\varepsilon > 3 = \chi^{\text{M}}_{4,2}(0, 2, 2, 2)$. Essentially, this happens due to that fact that for $(2, 2, 2, 2)$, facilities 1 and 3 are opened in the first iteration, player 4 is connected to 3, and $c(4, 3)$ is set to infinity. Thus, in the second iteration, his ghost has to grow longer to connect to a full facility, namely, facility 2. However, for $(0, 2, 2, 2)$, only facility 2 is opened in the first iteration due to the opening rule (To ensure that there is no tie between opening f_1

Fig. 7.4. Instance for which the method in [94] is not cross-monotonic



and f_2 in the first iteration, we have slightly changed the original example, letting o_{f_1} be 3ε instead of 2ε). Therefore, in the first and second iteration, $d(4,3)$ is 3.

Now, let us look at our method χ. The cost shares only differ at two entries: it is $\chi_{3,2}(2,2,2,2) = 1+4\varepsilon$ and, interestingly, $\chi_{4,2}(0,2,2,2) = 5+5\varepsilon$. We state the events for both $\boldsymbol{a} = (2,2,2,2)$ and $\boldsymbol{a} = (0,2,2,2)$ in Figure 7.5. Note that for both vectors, is is sufficient to consider growing the ghosts of players in A_2, since $A_1 = A_2$ and the cost shares for level 1 can be deduced from this process as well.

Fig. 7.5. Events for computing $\chi(\boldsymbol{a})$ for $\boldsymbol{a} = (2,2,2,2)$ (left) and for $\boldsymbol{a} = (0,2,2,2)$ (right)

$\boldsymbol{A}_2 = \{1,2,3,4\}$

t	*Event*
0	$1 \circ f_1$
3ε	$\mathbf{f_1}, 1 \bullet \mathbf{f_1}$
1	$2 \bullet \mathbf{f_1}$
$1+\varepsilon$	$2 \circ f_2$
$1+2\varepsilon$	$\mathbf{f_2}, 2 \bullet \mathbf{f_2}, 3 \bullet \mathbf{f_2}$
$1+3\varepsilon$	$3 \circ f_3$
$1+4\varepsilon$	$\mathbf{f_3}, 3 \bullet \mathbf{f_3}$

t	*Event*
$2+\varepsilon$	$1 \bullet \mathbf{f_2}$
3	$4 \bullet \mathbf{f_3}$
$3+3\varepsilon$	$3 \bullet \mathbf{f_1}$
$3+6\varepsilon$	$2 \bullet \mathbf{f_3}$
$4+6\varepsilon$	$1 \bullet \mathbf{f_3}$
$5+5\varepsilon$	$4 \bullet \mathbf{f_2}$

$\boldsymbol{A}_2 = \{2,3,4\}$

t	*Event*
1	$2 \circ f_1$
$1+\varepsilon$	$2 \circ f_2$
$1+2\varepsilon$	$\mathbf{f_2}, 2 \bullet \mathbf{f_2}, 3 \bullet \mathbf{f_2}$
$1+3\varepsilon$	$\mathbf{f_1}, 2 \bullet \mathbf{f_1}, 3 \circ f_3$
$1+4\varepsilon$	$\mathbf{f_3}, 3 \bullet \mathbf{f_3}$
3	$4 \bullet \mathbf{f_3}$
$3+3\varepsilon$	$3 \bullet \mathbf{f_1}$
$3+6\varepsilon$	$2 \bullet \mathbf{f_3}$
$5+5\varepsilon$	$4 \bullet \mathbf{f_2}$

7.5 Applications to Generalized Steiner Forest

Theorem 7.23 is the main theorem of this section:

Theorem 7.23. *For each* GenSteinerForest *cost-sharing problem there is a marginal cost-sharing method χ^{GS} and an approximation algorithm ALG such that Level$_{\chi^{\mathrm{GS}}}$ is GSP, $(2 \cdot H_L)$-BB, and $O(\log^2 n \cdot \log L)$-EFF for C_{ALG}. Furthermore, both Level$_{\chi^{\mathrm{GS}}}$ and ALG are computable in time polynomial in $|L|, |V|, |E|$, and n.*

Guideline of Proof: Fix a GenSteinerForest cost-sharing problem $(G, \boldsymbol{w}, \boldsymbol{s}, \boldsymbol{t})$ (for the remaining Section 7.5). In order to proof Theorem 7.23, we define a valid marginal cost-sharing method χ^{GS} in Section 7.5.1 and show that for every $\boldsymbol{a} \in \mathcal{A}$ and each $\ell \in [L]$, $\chi^{\text{GS}}_{*,\ell}(\boldsymbol{a})$ is computable in time polynomial in $|V|, |E|$, and n. Section 7.5.2 gives a polynomial-time computable approximation algorithm ALG. Section 7.5.3 then shows $(2 \cdot H_L)$-BB and $O(\log^2 n \cdot \log L)$-EFF for C_{ALG}. Finally, Theorem 7.23 follows directly from Theorem 7.11 and Lemma 7.12.

7.5.1 The Marginal Cost-Sharing Method

We utilize the binary demand cost-sharing method ξ^{K} for SteinerForest cost-sharing problems by Könemann et al. [82], which is known to be 2-BB [82] and $O(\log^2 n)$-SUM [19]. We define χ^{GS} simply by

$$\chi^{\text{GS}}_{*,\ell}(\boldsymbol{a}) := \xi^{\text{K}}(\boldsymbol{a}^\ell) \text{ for all allocations } \boldsymbol{a} \in \mathcal{A} \ .$$

Lemma 7.24. χ^{GS} *is valid.*

Proof. Let $\chi := \chi^{\text{GS}}$ and $\xi := \xi^{\text{K}}$.

Level restriction follows from $\chi_{i,\ell}(\boldsymbol{a}) = \chi_{i,\ell}(\ell \cdot \boldsymbol{a}^\ell)$ for all allocations $\boldsymbol{a} \in \mathcal{A}$, all players $i \in [n]$, and all $\ell \in [L_i]$. Cross-monotonicity is met, since for all allocations $\boldsymbol{a} \in \mathcal{A}$, all players $i \in [n]$ and $j \in [n] \setminus \{i\}$ with $a_j < L_j$, and all levels $\ell \in [L_i]$, it follows by the cross-monotonicity of ξ that

$$\chi_{i,\ell}(\boldsymbol{a}) = \xi_i(\boldsymbol{a}^\ell) \geq \xi_i((\boldsymbol{a} + e_j)^\ell) = \chi_{i,\ell}(\boldsymbol{a} + e_j) \ .$$

To show that χ is non-decreasing, fix $\boldsymbol{a} \in \mathcal{A}$ and let $\ell := \max_i\{a_i\}$. For all i with $a_i = \ell < L_i$,

$$\chi_{i,\ell}(\boldsymbol{a}) = \xi_i(\boldsymbol{a}^\ell) \leq \xi_i((\boldsymbol{a} + \sum_{\substack{j \in [n]: \\ a_j = \ell < L_j}} \boldsymbol{e}_j)^{\ell+1}) \leq \chi_{i,\ell+1}(\boldsymbol{a} + \sum_{\substack{j \in [n]: \\ a_j = \ell < L_j}} \boldsymbol{e}_j) \ .$$

□

The binary demand cost-sharing method ξ^{K} can essentially be computed by an approximation algorithm of Agrawal et al. [1] for SteinerForest with only a small modification which is crucial for cross-monotonicity (confer [82]). It goes beyond the scope of this thesis to discuss this computation in detail. We refer to [1] and [82] to verify polynomial time computability of $\xi^{\text{K}}(S)$ for all $S \subseteq [n]$ in time polynomial in $|V|, |E|$ and n.

7.5.2 The Approximate Solution

During the computation of $\xi^{\text{K}}(\boldsymbol{a}^\ell)$ for allocation $\boldsymbol{a} \in \mathcal{A}$ and each $\ell \in [\max_i a_i]$, a Steiner forest for the set induced by a^ℓ is computed in time polynomial in $|V|$, $|E|$, and n (essentially by the AKR algorithm of Agrawal, Klein, and Ravi; confer [1]). We let the cost of the Steiner forest for $\boldsymbol{a}^\ell$ be $C_{\mathsf{AKR}}(\boldsymbol{a}^\ell)$. We construct a solution for $\boldsymbol{a}$ that simply consists of the union of the Steiner forests from each iteration, where multiple edges count as copies. This straightforward solution construction for GenSteinerForest was also suggested by Goemans and Bertsimas [55]. We denote the induced cost by C_{ALG}, where $C_{\mathsf{ALG}}(\boldsymbol{a}) = \sum_{\ell=1}^{\max_i a_i} C_{\mathsf{AKR}}(\boldsymbol{a}^\ell)$ for all allocations $\boldsymbol{a} \in \mathcal{A}$.

7.5.3 Budget-Balance and Efficiency

Theorem 7.25. *χ^{GS} is $(2 \cdot H_L)$-BB for C_{ALG}.*

Proof. Let $\chi := \chi^{\text{GS}}$ and $\xi := \xi^{\text{K}}$. Fix $\boldsymbol{a} \in \mathcal{A}$ and let $X(\boldsymbol{a}) := \sum_{i=1}^{n} \sum_{\ell=1}^{a_i} \chi_{i,\ell}(\boldsymbol{a})$. We let $A_\ell := \{i \in [n] \mid a_i \geq \ell\}$ for all $\ell \in [\max_i a_i]$. It is

$$X(\boldsymbol{a}) = \sum_{\ell=1}^{\max_i a_i} \sum_{i \in A_\ell} \xi_i(\boldsymbol{a}^\ell) \geq \sum_{\ell=1}^{\max_i a_i} C_{\mathsf{AKR}}(\boldsymbol{a}^\ell) = C_{\mathsf{ALG}}(\boldsymbol{a}) \ ,$$

where the inequality holds as ξ^{K} is 2-BB for C_{AKR} [82]. We again utilize this property to show that

$$\begin{aligned} X(\boldsymbol{a}) = \sum_{\ell=1}^{\max_i a_i} \sum_{i \in A_\ell} \xi_i(\boldsymbol{a}^\ell) &\leq \sum_{\ell=1}^{\max_i a_i} 2 \cdot \mathsf{SF}(\boldsymbol{a}^\ell) \\ &= 2 \cdot \sum_{\ell=1}^{\max_i a_i} \frac{1}{\ell} \cdot \mathsf{GSF}(\ell \cdot \boldsymbol{a}^\ell) \leq 2 \cdot H_L \cdot \mathsf{GSF}(\boldsymbol{a}) \ . \end{aligned}$$

Theorem 7.26. *$Level_{\chi^{\text{GS}}}$ is $O(\log^2 n \cdot \log L)$-EFF for C_{ALG}.*

Proof. Let $\chi := \chi^{\text{GS}}$ and $\xi := \xi^{\text{K}}$. Fix $\boldsymbol{V} \in \mathcal{R}$ and let $\boldsymbol{q} := q(\boldsymbol{V})$. Let $\boldsymbol{a} \in \mathcal{A}$ be a service vector with optimal social cost. We show that

$$\sum_{i: a_i > q_i} \sum_{\ell=q_i+1}^{a_i} \max\{0, v_{i,\ell}\} \leq O(\log^2 n + \log L) \cdot \mathsf{GSF}(\boldsymbol{a}) \ .$$

The rest of the proof is along the lines of the proof of Theorem 7.20, and as a result, we get $O(\log^2 n \cdot \log L)$-EFF.

We consider the players in $S := \{i \in [n] \mid a_i > q_i\}$ in the reverse order they are deleted within $Level_\chi$ from vector $\boldsymbol{p}$ in the Moulin mechanism in line 4. Let this order be $s_1, \ldots, s_{|S|}$. For $j \in [|S|]$, let $\boldsymbol{p}^j \in \{0,1\}^n$ be the vector in which the entry $p^j_{s_j}$ is set to 0 by the Moulin mechanism. Furthermore, let $S_\ell := \{i \in S \mid a_i \geq \ell\}$, and $\boldsymbol{s}^\ell := \mathsf{in}(S_\ell)$. Finally, let $\boldsymbol{s}^{\ell,j} \in \{0,1\}^n$ be such that $s^{\ell,j}_i := 1 \Leftrightarrow (s^\ell_i = 1$ and $p^j_i = 1)$. Note that $\boldsymbol{s}^{\ell,j}$ indicates the first j elements of S_ℓ (according to the order $s_1, \ldots, s_{|S|}$).

By Assumption 2.2, Lemma 7.5, and definition of the vectors $\boldsymbol{p}^j$,

$$\begin{aligned} \sum_{i: a_i > q_i} \sum_{\ell=q_i+1}^{a_i} \max\{0, v_{i,\ell}\} &= \sum_{s_j \in S} \sum_{\ell=q_{s_j}+1}^{a_{s_j}} \max\{0, v_{s_j,\ell}\} \\ \leq \sum_{s_j \in S} \sum_{\ell=q_{s_j}+1}^{a_{s_j}} v_{s_j, q_{s_j}+1} &\leq \sum_{\ell=1}^{L} \sum_{s_j \in S_\ell} \xi_{s_j}(\boldsymbol{p}^j) \ . \end{aligned}$$

Furthermore, by cross-monotonicity and $O(\log^2 n)$-SUM of ξ (for function SF),

$$\sum_{\ell=1}^{L} \sum_{s_j \in S_\ell} \xi_{s_j}(\boldsymbol{p}^j) \leq \sum_{\ell=1}^{L} \sum_{s_j \in S_\ell} \xi_{s_j}(\boldsymbol{s}^{\ell,j}) \leq O(\log^2 n) \cdot \sum_{\ell=1}^{L} \mathsf{SF}(S_\ell) \ .$$

Finally,

$$\sum_{\ell=1}^{L} \mathsf{SF}(S_\ell) \leq \sum_{\ell=1}^{L} \frac{1}{\ell} \cdot \mathsf{GSF}(\boldsymbol{a}) \leq H_L \cdot \mathsf{GSF}(\boldsymbol{a}) \ .$$

$\square$

7.6 Conclusion and Open Problems

Among the few works for general demand cost sharing, we regard our work to be a substantial contribution to the development of GSP mechanisms for general demand.

- Central open questions are whether the provided budget-balance and efficiency approximations are tight under the validity requirement or even for the GSP requirement.
- In addition, it remains an open problem whether validity of marginal cost-shares is not only sufficient, but also necessary for GSP mechanisms.
- Observe, that we did not fully exploit the potential of $Level_\chi$. For both applications, the cost shares of level l only depend on the players that receive at least level l, while validity allows to make them dependent on the levels of the other players as well. It is an interesting question if this degree of freedom may lead to better approximations.

A

Budget-Balance and Efficiency Bounds for Moulin Mechanisms and Acyclic Mechanisms

Existing Moulin mechanisms are summarized in Table A.1. Budget-balance and efficiency approximations together in one line imply that there are Moulin mechanisms that fulfill these approximations simultaneously.

Table A.1. Best known budget-balance and social cost efficiency approximations of Moulin mechanisms (n.k. for not known)

Problem	from	BB	EFF	confer
general[1]	[114]	β	$\beta+\alpha$	Thm. 4.4, p. 33
submodular costs[2]	[114]	1	H_n	
SpanningTree	[76, 81]	1	n.k.	
SteinerTree[3]	[76, 81, 114]	2	$O(\log^2 n)$	
SteinerForest	[19, 82]	2	$O(\log^2 n)$	
FacilityLocation	[106, 115]	3	$O(\log n)$	
PriceCollectingSteinerForest	[66]	3	$O(\log^2 n)$	
ConnectedFacilityLocation[4]	[86]	30	n.k.	
VertexCover	[73]	$2\cdot\sqrt{n}$	n.k.	
EdgeCover	[73]	2	n.k.	
SingleSourceRentOrBuy[4]	[106] [67, 86, 115]	15 $4\cdot(1+\varepsilon)$	n.k. $O(\log^2 n)$	
MulticommodityRentOrBuy	[115]	$O(1)$	$\log^2 n$	
TravelingSalesman[3]	[76, 81]	2	n.k.	
$(\mathrm{Q}\|\|\mathrm{C}_{\max})$[5]	[14]	$2\cdot d$	$O(d\cdot\log n)$	Thm. 4.14, p. 38
$(\mathrm{Q}\|p_i{=}1\|\mathrm{C}_{\max})$[6]	[14]	$\frac{2m}{m+1}$	$O(\log n)$	Thm. 4.12, p. 36
$(\mathrm{P}\|\|\mathrm{C}_{\max})$[6]	[10] [16]	$\frac{2m}{m+1}$ $2-\frac{1}{m}$	$O(n)$ $O(\log n)$	

[1] α,β: smallest numbers such that $\xi(\cdot)$ is α-SUM and β-BB
[2] via the Shapley value [116]
[3] 2-BB, as a minimum spanning tree is a 2-approximation
[4] randomized, sharing of *expected* cost
[5] d: number of different processing times
[6] m: number of machines

Table A.2 gives lower bounds on the approximate budget-balance and the approximate efficiency of Moulin mechanisms.

Table A.2. Budget-balance and social cost efficiency lower bounds of Moulin mechanisms

Problem	from	BB	EFF	confer
general[1]	[114]	$\max\{\alpha, \beta\}$	$\max\{\alpha, \beta\}$	Thm. 4.4, p. 33
constant costs	[114]	1	H_n	Lemma 4.5, p. 35
SteinerTree	[76, 83] [115]	2	 $\Omega(\log^2 n)$	
SteinerForest[2]		2	$\Omega(\log^2 n)$	
SingleSourceRentOrBuy[2]		2	$\Omega(\log^2 n)$	
MulticommodityRentOrBuy[2]		2	$\Omega(\log^2 n)$	
FacilityLocation	[73] [115]	3	 $\Omega(\log n)$	
VertexCover[6]	[73]	$\Omega(n^{1/3})$	$\Omega(n^{1/3})$	
SetCover[6]	[73]	$\Omega(n)$	$\Omega(n)$	
EdgeCover[7]	[73]	$2-\varepsilon$	H_n	
$(1\|\|\sum \mathrm{C}_i)$[3,6]	[16]	$\frac{n}{2}$	$\frac{n}{2}$	
$(\mathrm{Q}\|\|\mathrm{C}_{\max})$[4,6,7]	[10]	d	d H_n	Thm. 4.9, p. 36
$(\mathrm{P}\|p_i{=}1\|\mathrm{C}_{\max})$[5,7]	[10]	$\frac{2m}{m+1}$	H_n	Thm. 4.7, p. 35

[1] α, β: smallest numbers such that $\xi(\cdot)$ is α-SUM and β-BB
[2] follows from bounds for Steiner tree
[3] already for a single machine
[4] d: number of different processing times
[5] m: number of machines
[6] efficiency entry due to the first line
[7] efficiency entry due to the second line

Table A.3 summarizes known approximation results for acyclic mechanisms compared to the performance of Moulin mechansims.

Table A.3. Approximation results from [31, 94] for acyclic mechanisms compared to lower bounds of Moulin mechanisms (confer Table A.2)

	Moulin Lower Bounds		**Acyclic Upper Bounds**	
Problem	BB	EFF	BB	EFF
VertexCover	$\Omega(n^{1/3})$	$\Omega(n^{1/3})$	2	$O(\log n)$
SetCover	$\Omega(n)$	$\Omega(n)$	$O(\log n)$	$O(\log n)$
FacilityLocation	3	$\Omega(\log n)$	1.61	$O(\log n)$
SteinerTree	2	$\Omega(\log^2 n)$	2	$O(\log^2 n)$

B

Incremental and Groves Mechanisms

B.1 Incremental (Sequential Stand Alone) Mechanisms

Incremental (or *sequential stand alone*) mechanisms for binary demand cost sharing process the players in a fixed order and decide to serve a player if he can pay for the pivotal cost of adding him to the set of already served players. His cost share is exactly this pivotal cost. Let C be a cost function and $\pi : [n] \to [n]$ specify an ordering such that $\pi([n]) = [n]$, $\pi(i)$ is the position of player i and $\pi^{-1}(i)$ is the player at position i. The corresponding mechanism $\mathit{MechInc}_{C,\pi}$ is computed by Algorithm B.1:

Algorithm B.1 (computing $\mathit{MechInc}_{C,\pi}(\boldsymbol{b}) = (Q(\boldsymbol{b}), x(\boldsymbol{b}))$).

1: $Q := \emptyset$; $\boldsymbol{x} := 0$
2: **for** $i := \pi^{-1}(1), \ldots, \pi^{-1}(n)$ **do**
3:     **if** $b_i \geq C(Q \cup \{i\}) - C(Q)$ **then**
4:         $x_i := C(Q \cup \{i\}) - C(Q)$; $Q := Q \cup \{i\}$
5: **return** $(Q, \boldsymbol{x})$

It is straightforward to see that for any cost function C and any permutation π $\mathit{MechInc}_{C,\pi}$ meets SP, 1-BB, VP, NPT, and strict CS. In addition, $\mathit{MechInc}_{C,\pi}$ is GSP if the cost function is submodular: the cost share of a player computed for the true valuations can only decrease if more players prior to him in the order bid such as to receive the service. Especially for the first such player this results in a negative utility. Note that it can be easily observed that the cost-shares by $\mathit{MechInc}_{C,\pi}$ are cross-monotonic and due to Corollary C.4, $\mathit{MechInc}_{C,\pi}$ is welfare equivalent to a Moulin mechanism.

If C is supermodular, replacing '$\geq$' in line 3 by '$>$' yields a GSP mechanism: Here, the cost share of a player computed for true valuations can only decrease if preceding players bid in order to be rejected. According to the replacement, in particular the first such player had a positive utility when bidding truthfully.

In general, however, GSP cannot be guaranteed, as Example B.1 illustrates. Furthermore, Example B.2 yields unsatisfying performance with respect to efficiency.

Example B.1. Consider $C : 2^{[3]} \to \mathbb{N}$ with

$$C(S) := \begin{cases} 0 & \text{if } S = \emptyset \\ 2 & \text{if } |S| = 1 \\ 3 & \text{if } |S| = 2 \\ 5 & \text{if } |S| = 3 \ . \end{cases}$$

C is neither sub- nor supermodular. Without loss of generality, look at π with $\pi(i) = i$ for all $i \in [n]$. Let $\boldsymbol{v} := (2,2,2)$. For $\mathit{MechInc}_{C,\pi} = (Q,x)$ as in Algorithm B.1, it is $Q(\boldsymbol{v}) = [3]$ and $x(\boldsymbol{v}) = (2,1,2)$. For $\boldsymbol{b} := (0,2,2)$ we have $Q(\boldsymbol{b}) = \{2,3\}$ and $x(\boldsymbol{b}) = (0,2,1)$. Thus, $K = \{1,3\}$ can successfully form a coalition as $u_1(\boldsymbol{v}, v_1) = u_1(\boldsymbol{b}, v_1) = u_3(\boldsymbol{v}, v_3) = 0$ and $u_3(\boldsymbol{b}, v_3) = 1$. For mechanism $\mathit{MechInc}'_{C,\pi} = (Q', x')$ obtained by replacing '$\geq$' with '$>$', it is $Q'(\boldsymbol{v}) = \emptyset$. For $b := (3,2,2)$ however, we have $Q(\boldsymbol{b}) = \{1,2\}$ and $x(\boldsymbol{b}) = (2,1,0)$. Here, $K = \{1,2\}$, $u_1(\boldsymbol{b}, v_1) = u_1(\boldsymbol{v}, v_2) = u_2(\boldsymbol{v}, v_2) = 0$ and $u_2(\boldsymbol{b}, v_2) = 1$.

Example B.2. Let $C : 2^{[n]} \to \mathbb{N}$ with $C(S) := 1$ for all $S \subseteq [n]$ and consider π with $\pi(i) = i$ for all $i \in [n]$ without loss of generality. Furthermore, consider $\boldsymbol{v} \in \mathbb{R}^n$ with $v_i := 1 - \varepsilon$ for all $i \in [n]$. For $\mathit{MechInc}_{C,\pi} = (Q,x)$ as in Algorithm B.1, the social welfare of the computed solution $Q(\boldsymbol{v}) = \emptyset$ is $SW_C(\emptyset, \boldsymbol{v}) = 0$, while the optimal social welfare is $SW_C([n], \boldsymbol{v}) = n \cdot (1 - \varepsilon) - 1$. For the social cost efficiency measure, we have $SC_C(\emptyset, \boldsymbol{v}) = n \cdot (1 - \varepsilon)$, while the optimal social cost is $SC_C([n], \boldsymbol{v}) = 1$. The same holds for replacing '$\geq$' with '$>$' in Algorithm B.1.

In the following, we show how to adapt incremental mechanisms to general demand cost sharing in Algorithm B.2. For given maximum service levels $L_1, \ldots, L_n$, we let s be a sequence of players such that each player i appears exactly L_i times in the sequence.

Algorithm B.2 (computing $\mathit{MechIncGD}_{C,s}(\boldsymbol{B}) := (q(\boldsymbol{B}), x(\boldsymbol{B}))$).

$\boldsymbol{x} := \boldsymbol{0}$; $\boldsymbol{q} := \boldsymbol{0}$; $i := \mathit{first}(s)$
while $i \neq \mathit{null}$ **do**
    if $b_{i,q_i+1} < C(\boldsymbol{q} + \boldsymbol{e}_i) - C(\boldsymbol{q})$ **then**
        delete i from the whole sequence s
    else
        $q_i := q_i + 1$; $x_i := x_i + C(\boldsymbol{q} + \boldsymbol{e}_i) - C(\boldsymbol{q})$
        delete i from s only at current position
    $i := \mathit{first}(s)$
return $(\boldsymbol{q}, \boldsymbol{x})$

The *sequential stand alone mechanisms* are incremental mechanisms which only allow sequences in which all entries i for a player i occur consecutively. In particular, $\mathit{MechInc}_{C,\pi}$ is a sequential stand alone mechanism.

Moulin [97] investigated incremental mechanisms for general demand by looking at generalizations of sub- and supermodularity. Let

$$\delta_{ij} C(\boldsymbol{q}) := C(\boldsymbol{q} + \boldsymbol{e}_i + \boldsymbol{e}_j) - C(\boldsymbol{q} + \boldsymbol{e}_i) - C(\boldsymbol{q} + \boldsymbol{e}_j) + C(\boldsymbol{q}) \ ,$$

defined only if $q_i \leq L_i - 1$, $q_j \leq L_j - 1$ if $i \neq j$, and $q_i \leq L_i - 2$ if $i = j$.

Under assumption of diminishing marginal returns (confer Assumption 2.2), Moulin [97] showed that incremental cost-sharing mechanisms are GSP for two cases. First, if costs are submodular ($\delta_{ij} C(\boldsymbol{q}) < 0$ for all $i, j, \boldsymbol{q}$ with $i \neq j$) and marginal costs are decreasing ($\delta_{ii} C(\boldsymbol{q}) < 0$ for all $i, \boldsymbol{q}$); second, if costs are supermodular ($\delta_{ij} C(\boldsymbol{q}) > 0$ for all $i, j, \boldsymbol{q}$ with $i \neq j$) and marginal costs are increasing ($\delta_{ii} C(\boldsymbol{q}) > 0$ for all $i, \boldsymbol{q}$).

If costs are submodular and marginal costs are decreasing, he showed a negative result already for 2 players with $L_i \geq 2$ for $i \in \{1,2\}$. In this case, only the two possible sequential stand alone mechanisms yield 1-BB and GSP mechanisms. If costs are supermodular and marginal cost are increasing, he showed that essentially all GSP and 1-BB mechanisms satisfying a certain continuity property (see [97]) are incremental mechanisms.

Example B.3 shows that the social cost efficiency of incremental mechanisms is in general no better than $\sum_{i\in[n]} L_i$.

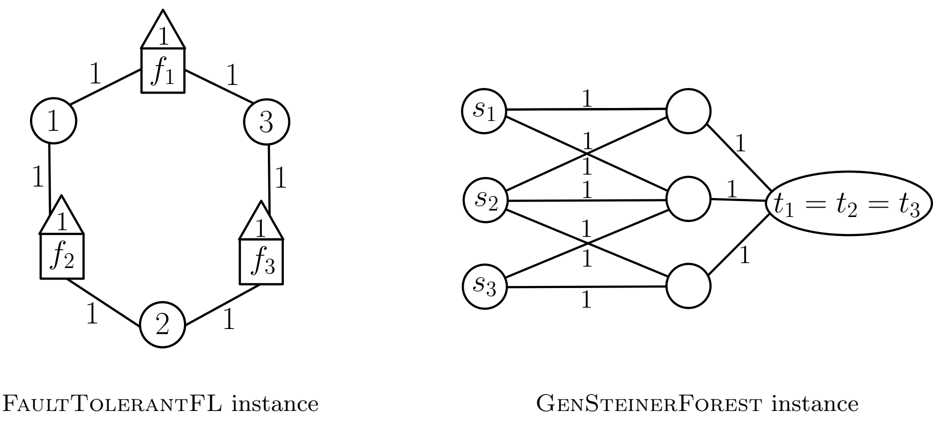
Example B.3. Let $C : \mathcal{A} \to \mathbb{N}$ with $C(\boldsymbol{q}) = 1$ for all $\boldsymbol{q} \in \mathcal{A}$ and, without loss of generality, π with $\pi(i) = i$ for all $i \in [n]$. Furthermore, consider $\boldsymbol{V} \in \mathcal{R}$ with $v_{i,l} := 1 - \varepsilon$ for all $i \in [n]$ and all $l \in [L_i]$. The social cost of the computed solution $q(\boldsymbol{V}) = \boldsymbol{0}$ is $\sum_{i\in[n]}(1-\varepsilon)\cdot L_i$, while the optimal social cost is 1, obtained by allocation $(L_1, \dots, L_n)$.

Incremental Mechanisms are not GSP for our Problems

In general, there are no GSP incremental mechanisms for $(\mathrm{Q}||\mathrm{C}_{\max})$ cost-sharing problems, as already for a $(\mathrm{P2}|p_i{=}1|\mathrm{C}_{\max})$ problem with three players, we get symmetric costs C with $c(1) = c(2) = 1$ and $c(3) = 2$ (which are neither sub- nor supermodular). Now let $\boldsymbol{v} := (1, 2, 1)$. For mechanism *MechInc* $= (Q, x)$ from Algorithm B.1 with input $\boldsymbol{b} := (1, -1, 1)$, it is $Q(\boldsymbol{v}) = [3], x(\boldsymbol{v}) = (1, 0, 1), Q(\boldsymbol{b}) = \{1, 3\}, x(\boldsymbol{b}) = (1, 0, 0)$. Thus, $K = \{2, 3\}$ can successfully form a coalition. For mechanism *MechInc*$'$ $= (Q', x')$ obtained by replacing '$\geq$' with '$>$' and $b' := (2, 2, 1)$, we get that $Q'(\boldsymbol{v}) = (0, 1, 1), x'(\boldsymbol{v}) = (0, 1, 0), Q'(\boldsymbol{b}') = (1, 1, -1)$ and $x'(\boldsymbol{b}') = (1, 0, 0)$. Here, a successful coalition is $\{1, 2\}$.

In addition, the FaultTolerantFL and GenSteinerForest instances in Figure B.1 induce the costs from Example B.1.

Fig. B.1. FaultTolerantFL and GenSteinerForest instances with costs from Example B.1



Binary Demand Sequential Stand Alone Mechanisms are CGSP

We show acyclicity; CGSP then follows from Theorem 6.14.

Lemma B.4. *For every cost function C and every ordering π, $MechInc_{C,\pi}$ is acyclic.*

Proof. Define $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ and $\tau : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ as follows. For $S \subseteq [n]$ and $i \in [n]$, let $\xi_i(S) := C(\{j \in S \mid \pi(j) \leq \pi(i)\}) - C(\{j \in S \mid \pi(j) < \pi(i)\})$ if $i \in S$ and 0 otherwise. Furthermore, $\tau_i(S) := \pi(i)$.

$\square$

B.2 Groves Mechanisms

Applying the celebrated family of Groves mechanisms [25, 63, 131] is essentially the only way to achieve social welfare efficiency and strategyproofness at the same time. As Groves mechanisms are defined for a rather general scenario, we first discuss them in their original setting and afterwards apply them to our cost-sharing model for binary demand.

In the original setting, a choice dependent on players' preferences has to be made from some set of *social alternatives*. An alternative consists of an element $k \in \mathcal{K}$, the set of *project choices*, and a payment $t_i \in \mathbb{R}$ for each player. A negative t_i is interpreted as a bonus given to player i.

The preference of player $i \in [n]$ is modeled by i's *true type* θ_i^* that he privately observes from the set of his possible types Θ_i prior to the choice. Only the true types θ_i^* are private information, everything else is assumed to be common knowledge among the players. We let $\Theta := \Theta_1 \times \ldots \times \Theta_n$ be the space of all possible type profiles $\boldsymbol{\theta} := (\theta_1, \ldots, \theta_n)$. Depending on his type, player i appraises a project choice by a *valuation function* $\nu_i : \mathcal{K} \times \Theta_i \to \mathbb{R}$, and a social alternative $(k, \boldsymbol{t})$ by a quasi-linear utility function $u_i : \mathcal{K} \times \mathbb{R}^n \times \Theta_i \to \mathbb{R}$ with $u_i((k, \boldsymbol{t}), \theta_i) = \nu_i(k, \theta_i) - t_i$.

In order to choose an alternative (k, t) we consider a mechanism $\Gamma := (k, t)$, where $k : \Theta \to \mathcal{K}$ and $t : \Theta \to \mathbb{R}^n$.

According to Definition 2.5, Γ is strategyproof if for every profile vector $\boldsymbol{\theta} \in \Theta$ there is no player i with true valuation $\theta_i^* \in \Theta_i$ such that

$$u_i((k(\theta_i, \boldsymbol{\theta}_{-i}), t(\theta_i, \boldsymbol{\theta}_{-i})), \theta_i^*) > u_i(k((\theta_i^*, \boldsymbol{\theta}), t(\theta_i^*, \boldsymbol{\theta}), \theta_i^*) \ .$$

Furthermore, Γ is called *efficient* if, assuming truthful bids, it always selects a project choice that maximizes $\sum_{i=1}^n \nu_i(k, \theta_i^*)$ over all $k \in \mathcal{K}$. Note that this definition of efficiency seems to deviate from our notion of efficiency from Definitions 2.13 and 2.14. However, we will fit the cost-sharing scenario into this model in such a way that these definitions are equivalent.

Definition B.5. *A mechanism $M = (k, t)$ is called a* Groves mechanism*, if for all inputs $\boldsymbol{\theta} \in \Theta$ and all $k \in \mathcal{K}$ it holds that*

$$\sum_{i=1}^{n} \nu_i(k(\boldsymbol{\theta}), \theta_i) \geq \sum_{i=1}^{n} \nu_i(k, \theta_i) \ \textit{and} \tag{B.1}$$

$$t_i(\boldsymbol{\theta}) = h_i(\boldsymbol{\theta}_{-i}) - \sum_{j \in [n] \setminus \{i\}} \nu_j(k(\boldsymbol{\theta}), \theta_j) \ , \tag{B.2}$$

where h_i is an arbitrary function of $\boldsymbol{\theta}_{-i}$.

By Equation (B.1), Groves mechanisms are efficient. Moreover, it is a simple observation that Groves mechanisms are strategyproof:

Lemma B.6. *Groves mechanisms are strategyproof.*

Proof. Let $\Gamma = (k, t)$ be a Groves mechanism and assume that Γ is not strategyproof. Then there exists a type profile $\boldsymbol{\theta} \in \Theta$, a player $i \in [n]$, and a true type $\theta_i^* \in \Theta_i$ such that

$$u_i((k(\theta_i, \boldsymbol{\theta}_{-i}), t(\theta_i, \boldsymbol{\theta}_{-i})), \theta_i^*) > u_i((k(\theta_i^*, \boldsymbol{\theta}_{-i}), t(\theta_i^*, \boldsymbol{\theta}_{-i})), \theta_i^*) \ .$$

Substituting the payments according to (B.2) an eliminating function $h_{-i}(\boldsymbol{\theta}_{-i})$ on both sides leads to

$$\begin{aligned} &\nu_i(k(\theta_i, \boldsymbol{\theta}_{-i}), \theta_i^*) + \sum_{j \in [n] \setminus \{i\}} \nu_j(k(\theta_i, \boldsymbol{\theta}_{-i}), \theta_j^*) \\ > \ &\nu_i(k(\theta_i^*, \boldsymbol{\theta}_{-i}), \theta_i^*) + \sum_{j \in [n] \setminus \{i\}} \nu_j(k(\theta_i^*, \boldsymbol{\theta}_{-i}), \theta_j^*) \ , \end{aligned}$$

contradicting Equation (B.1). □

Independently, Clarke [25] and Vickrey [131] proposed members of the family of Groves mechanisms for which

$$h_i(\boldsymbol{\theta}_{-i}) = \sum_{j \in [n] \setminus \{i\}} \nu_j(k_{-i}(\boldsymbol{\theta}_{-i}), \theta_j) \ ,$$

where for all $\boldsymbol{\theta}_{-i} \in \Theta_{-i}$ and all $k \in \mathcal{K}$, k_{-i} satisfies

$$\sum_{j \in [n] \setminus \{i\}} \nu_j(k_{-i}(\boldsymbol{\theta}_{-i}), \theta_j) \geq \sum_{j \in [n] \setminus \{i\}} \nu_j(k, \theta_j) \ .$$

Player i's payment reflects the pivotal change of the joint valuation the others have for $k(\boldsymbol{\theta})$ compared to the joint valuation the others have for $k_{-i}(\boldsymbol{\theta}_{-i})$. Hence, next to *Vickrey-Clarke-Groves* mechanisms, these mechanisms are also called *pivotal* mechanisms.

A result by Green and Laffont [60] states that essentially *all* efficient and strategyproof mechanisms have to compute the payments as in (B.2), under the premise that the set of possible types for each player is sufficiently rich. In other words, Groves mechanisms characterize the *only* mechanisms that are efficient and strategyproof. Unfortunately, the famous Green and Laffont impossibility theorem [61] rules out the existence of Groves mechanisms that simultaneously meet the budget-balance condition of $\sum_{i=1}^n t_i(\boldsymbol{\theta}) = 0$ for all $\boldsymbol{\theta} \in \Theta$.

Groves (Marginal Cost) Mechanisms for Cost Sharing

In order to relate Groves mechanisms to the cost-sharing scenario, observe that the set of possible types Θ_i for player i is given by $\mathbb{R}$, and that the set of project choices $\mathcal{K}$ is given by $2^{[n]}$. The true type θ_i^* of a player i is simply his true valuation v_i, the actual input $\boldsymbol{\theta}$ is the bid vector $\boldsymbol{b}$. For $S \subseteq [n]$, we let $\nu_i(S, b_i) := b_i$ if $i \in S$ and 0 otherwise. In order to relate the efficiency condition implied by (B.1) to our notion of efficiency in Definitions 2.13 and 2.14, we have to introduce the service provider as an additional player, referred to as 'player 0'. For $S \subseteq [n]$, we define the valuation function of the provider to be $\nu_0(S, b_0) := -C(S)$ (independent of his dummy bid b_0). Then, for any vector $\boldsymbol{v}$ of true valuations,

$$\begin{aligned} \max_{S \subseteq [n]} \left(\sum_{i=0}^n \nu_i(S, v_i) \right) &= \max_{S \subseteq [n]} \left(\sum_{i \in S} v_i - C(S) \right) \\ &= \max_{S \subseteq \{i \in [n] | v_i \geq 0\}} \left(\sum_{i \in S} v_i - C(S) \right) \ , \end{aligned}$$

implying that efficiency in this model is equivalent to efficiency (1-EFF) in the cost-sharing model. According to (B.1), the function Q becomes

$$Q(\boldsymbol{b}) := \text{lexicographic largest set in} \\ \big\{S \subseteq [n] \mid SW_C(S, \boldsymbol{b}) \geq SW_C(S', \boldsymbol{b}) \text{ for all } S' \subseteq [n]\big\}$$

Naturally, we would have to define the payments $x(\boldsymbol{b})$ in such a way that they meet (B.2) to account for SP and ensure that $x_0(\boldsymbol{b}) = -C(Q(\boldsymbol{b}))$, as this is the bonus the provider has to receive in order to cover his cost. Furthermore, we have to ensure that the constructed Groves mechanism meets VP, NPT, and strict CS. However, even though we can achieve all of these requirements, 1-BB requires $x_0(\boldsymbol{b}) + \sum_{i=1}^n x_i(\boldsymbol{b}) = \sum_{i=1}^n x_i(\boldsymbol{b}) - C(Q(\boldsymbol{b})) = 0$, which by the Green and Laffont impossibility theorem is infeasible. Still, for the sake of illustration, we define adequate corresponding payments and give examples with poor budget-balance.

We define the payment of the provider by letting

$$h_{-0}(\boldsymbol{b}_{-0}) := \sum_{j \in [n]_0 \setminus \{0\}} \nu_j(Q(\boldsymbol{b}), b_j) - C(Q(\boldsymbol{b})) \ .$$

Note that Q's choice is independent of the dummy bid b_0, thus $Q(\boldsymbol{b}) = Q(\boldsymbol{b}_{-0})$.

We define the payments for players 1 to n according to the pivotal mechanism. Let Q_{-i} be the function corresponding to k_{-i}, i.e.,

$$Q_{-i}(\boldsymbol{b}_{-i}) := \text{lexicographic largest set in} \\ \big\{S \subseteq [n] \setminus \{i\} \mid SW_C(S, \boldsymbol{b}) \geq SW_C(S', \boldsymbol{b}) \text{ for all } S' \subseteq [n] \setminus \{i\}\big\} \ .$$

Furthermore, let $S^*_{-i} := Q_{-i}(\boldsymbol{b}_{-i})$, and $S^* := Q(\boldsymbol{b})$. Then define

$$\begin{aligned} x_i(\boldsymbol{b}) &:= \sum_{j \in [n]_0 \setminus \{i\}} \nu_j(Q_{-i}(\boldsymbol{b}_{-i}), b_j) - \sum_{j \in [n]_0 \setminus \{i\}} \nu_j(Q(\boldsymbol{b}), b_j) \\ &= \sum_{j \in S^*_{-i}} b_j - C(S^*_{-i}) - \sum_{j \in S^* \setminus \{i\}} b_j + C(S^*) \\ &\leq \sum_{j \in S^*} b_j - C(S^*) - \sum_{j \in S^* \setminus \{i\}} b_j + C(S^*) \\ &= b_i \ . \end{aligned}$$

Obviously, NPT holds due to $x_i(\boldsymbol{b}) \geq 0$ for all $i \in [n]$. VP follows from the fact that for all $i \in [n]$, $x_i(\boldsymbol{b}) \leq b_i$ and $x_i(\boldsymbol{b}) = 0$ if $i \notin S^*$ (in this case, $S^*_{-i} = S^*$). For any bid that is strictly larger than $\max_{S \subseteq [n] \setminus \{i\}} \{C(S \cup \{i\}) - C(S)\}$, $i \in [n]$ will be in the set maximizing social welfare. Furthermore, a negative bid will always result in not being in the set maximizing social welfare. Thus, this Groves mechanism meets strict CS. Moulin and Shenker [99] refer to this specific Groves mechanism as the *marginal cost mechanism.*

Example B.7 shows that in general the provider cannot hope for a reasonable cost recovery:

Example B.7. For $M > 0$, define $C(S) := M \cdot |S|$ for all $S \subsetneq [n]$ and $C([n]) := M \cdot (n-1)$. Let the true valuations be $v_i := M$ for all $i \in [n]$. Social welfare efficiency requires to serve all players at cost $M \cdot (n-1)$, as for all $S \subsetneq [n]$, $SW_C(S, \boldsymbol{v}) = 0$ and $SW_C([n], \boldsymbol{v}) = M$. However, using the pivotal mechanism, every player pays 0.

On the other hand, Example B.8 shows that in general the players are not guaranteed to not being overcharged:

Example B.8. For a sufficiently large $M > 0$, let $C(S) := \varepsilon$ for all $S \subsetneq [n]$ and $C([n]) := M$. Furthermore, let the true valuations be $v_i := M + \varepsilon$ for all $i \in [n]$. It holds that for all $S \subsetneq [n]$ that $SW_C(S, \boldsymbol{v}) = (n-1) \cdot (M+\varepsilon) - \varepsilon < (n-1) \cdot M + n \cdot \varepsilon$ and $SW_C([n], \boldsymbol{v}) = (n-1) \cdot M + n \cdot \varepsilon$. The set $[n]$ maximizes social welfare. However, the payments computed by the pivotal mechanism are $M - \varepsilon$ for each player.

Besides not being able to guarantee reasonable budget-balance approximations, Groves mechanisms are vulnerable to collusion. For Groves mechanisms, Green and Laffont [62] considered a model of coalition forming in which a coalition forms if players can strictly improve the *sum* of their utilities (and may then distribute their surplus via side-payments). They show that averting this kind of coalition forming is impossible for Groves mechanisms. On the positive side, they can show that for a fixed coalition the expected gain to cheating compared to telling the truth decreases with the number of players in the population. Example B.9 shows that even WGSP cannot be guaranteed by Groves mechanisms in general:

Example B.9. Let $n = 2$ and for all $\emptyset \neq S \subseteq [2]$, let $C(S) := 3$. In addition, $\boldsymbol{v} := (2, 2)$. Maximizing social welfare efficiency requires to serve both players. The payments of the pivotal mechanisms are 1 for each player. However, for $\boldsymbol{b} := (3, 3)$, both players are served for a cost-share of 0.

Once we restrict attention to submodular costs, Moulin and Shenker [99] showed that marginal cost mechanisms are essentially the only NPT, VP, and strict CS mechanism from the class of Groves mechanisms. Furthermore, they showed that submodularity ensures that the overall cost share never exceeds the service cost, i.e., the scenario from Example B.8 does not occur. In addition, Moulin and Shenker [99] discussed that even for submodular costs, Groves mechanisms are in general not group-strategyproof.

Presumably due to their bad performance with respect to budget-balance, there are only few works that consider Groves mechanisms (and especially the marginal cost mechanism) within cost-sharing scenarios (see, e.g., [4, 36, 37, 99]). For instance, marginal cost mechanisms have been considered for MULTICAST cost-sharing problems with fixed transmission trees inducing submodular costs. Feigenbaum et al. [37] investigated the distributed computation of the marginal cost mechanism and showed that it can be computed by sending 2 messages per link in the multicast tree. Contrary they showed that computing the Shapley value for these problems requires a quadratic total number of messages.

C

Characterization of GSP Mechanisms for Submodular Costs

We have discussed at the beginning of Chapter 4 that there always exists 1-BB and cross-monotonic cost-sharing methods (and thus 1-BB and GSP (Moulin) mechanisms) if cost are submodular. In this chapter, we introduce Theorem C.3 from Moulin [97] that completes the characterization of 1-BB and GSP under submodular costs, showing that every 1-BB and GSP mechanism for submodular costs is equivalent to a Moulin mechanism.

For the proof of Theorem C.3, we need the notion of *strong sets* given in Definition C.2. We also remind the reader of the result by Moulin [97], showing that for a GSP mechanism, cost-shares are uniquely defined by the set of served agents:

Theorem C.1. [97] *Let $M = (Q, x)$ be a GSP mechanism. Then, for any two vectors $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$, it holds that $x(\boldsymbol{b}) = x(\boldsymbol{b}')$.*

Proof. Let $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n$ with $Q(\boldsymbol{b}) = Q(\boldsymbol{b}')$. First consider vector $\boldsymbol{b}$. Fix an arbitrary $S \subseteq [n]$ and define $\boldsymbol{d}$ as:

$$d_i := \begin{cases} b_i^+ & \text{for all } i \in S \cap Q(\boldsymbol{b}) \\ -1 & \text{for all } i \in S \setminus Q(\boldsymbol{b}) \\ b_i & \text{for all } i \notin S \end{cases}$$

Moulin shows that it holds that

$$\text{for all } i \in S : i \in Q(\boldsymbol{b}) \Leftrightarrow i \in Q(\boldsymbol{d}) \text{ and } x_i(\boldsymbol{b}) = x_i(\boldsymbol{d}) \tag{C.1}$$

$$\text{for all } i \notin S : u_i(\boldsymbol{b}, b_i) = u_i(\boldsymbol{d}, b_i) \ . \tag{C.2}$$

Then, considering $\boldsymbol{b}'$ instead of $\boldsymbol{b}$ yields the same vector $\boldsymbol{d}$ and (C.1) – (C.2) hold with $\boldsymbol{b}$ replaced by $\boldsymbol{b}'$. For $S := [n]$, we get that $x_i(\boldsymbol{b}) = x_i(\boldsymbol{d}) = x_i(\boldsymbol{b}')$ for all $i \in [n]$, thus $x(\boldsymbol{b}) = x(\boldsymbol{b}')$.
We omit the proof of (C.1) and (C.2) as we have shown a more general result in Theorem 6.4 in Chapter 6, restricting GSP to CGSP and strict CS to CS.

Clearly, every GSP mechanism $M = (Q, x)$ *induces* a unique cost-sharing method ξ, by setting $\xi(S) := x(\boldsymbol{b})$ where $b_i < 0$ if $i \notin S$ and $b_i = b_i^+$ if $i \in S$.

Definition C.2. *A set $S \subseteq [n]$ is a* strong set *for a cost-sharing method ξ and a vector $\boldsymbol{b} \in \mathbb{R}^n$ if there are no sets $K, T \subseteq [n]$ such that*

$$\textit{for all } i \notin K : i \in T \Leftrightarrow i \in S \tag{C.3}$$

$$\textit{for all } i \in K : b_i \cdot \mathsf{in}_i(T) - \xi_i(T) \geq b_i \cdot \mathsf{in}_i(S) - \xi_i(S) \ , \tag{C.4}$$

with at least one strict inequality.

Theorem C.3. [97] *For any GSP mechanism M that is 1-BB with respect to submodular costs there exists a cross-monotonic cost-sharing method ξ such that Moulin_ξ is welfare equivalent to M.*

Proof. Let $M = (Q, x)$. Due to Theorem C.1 we may define a cost-sharing method ξ by setting $\xi(S) := x(\boldsymbol{b})$ for all $S \subseteq [n]$, where $b_i < 0$ if $i \notin S$ and $b_i = b_i^+$ if $i \in S$. In particular, $x(\boldsymbol{b}) = \xi(Q(\boldsymbol{b}))$ for all $\boldsymbol{b} \in \mathbb{R}^n$.

1. First, we show that for every $\boldsymbol{b} \in \mathbb{R}^n$, $Q(\boldsymbol{b})$ is a strong set for ξ and $\boldsymbol{b}$.
   Fix $\boldsymbol{b} \in \mathbb{R}^n$ and assume that $Q(\boldsymbol{b})$ is not strong. Thus, there exist subsets $K, T \subseteq [n]$ meeting
   $$\text{for all } i \notin K : i \in T \Leftrightarrow i \in Q(\boldsymbol{b}) \tag{C.5}$$
   $$\text{for all } i \in K : b_i \cdot \mathsf{in}_i(T) - \xi_i(T) \geq b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) \;, \tag{C.6}$$
   $$\text{with at least one strict inequality.}$$
   Let $\boldsymbol{d} := ((\mathbf{-1})_{-T}, \boldsymbol{b}_T^+)$. By (C.1), (C.2) of Theorem C.1 (where $S = [n] \setminus K$) and (C.5) it holds that
   $$\text{for all } i \in [n] : u_i(\boldsymbol{b}, b_i) = u_i((\boldsymbol{d}_{-K}, \boldsymbol{b}_K), b_i) \;. \tag{C.7}$$
   By (C.6) and (C.7), for all $i \in K$:
   $$\begin{aligned} u_i(\boldsymbol{d}, b_i) &= b_i \cdot \mathsf{in}_i(Q(\boldsymbol{d})) - \xi_i(Q(\boldsymbol{d})) \\ &= b_i \cdot \mathsf{in}_i(T) - \xi_i(T) \\ &\geq b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) \\ &= u_i(\boldsymbol{b}, b_i) \\ &= u_i((\boldsymbol{d}_{-K}, \boldsymbol{b}_K), b_i) \;. \end{aligned}$$
   with at least one strict inequality, contradicting GSP for true valuations b_i for all $i \in K$.
2. Second, we show that for every bid vector $\boldsymbol{b} \in \mathbb{R}^n$ and any two strong sets $S, S' \subseteq [n]$ for ξ and $\boldsymbol{b}$ it holds that
   $$\text{for all } i \in [n] : \; b_i \cdot \mathsf{in}_i(S) - \xi_i(S) = b_i \cdot \mathsf{in}_i(S') - \xi_i(S') \;. \tag{C.8}$$
   We show (C.8) by induction on $\kappa(\boldsymbol{b}) := |\{i \in [n] \mid b_i = b_i^+ \text{ or } b_i = -1\}|$.
   - Consider $\boldsymbol{b} \in \mathbb{R}^n$ with $\kappa(\boldsymbol{b}) = n$. By (1.), $Q(\boldsymbol{b})$ is a strong set for ξ and $\boldsymbol{b}$. We show now that $Q(\boldsymbol{b})$ is the *unique* strong set for ξ and $\boldsymbol{b}$.
     Assume that $S \neq Q(\boldsymbol{b})$ is another strong set.
     – If there exists $i \in S$ with $i \notin Q(\boldsymbol{b})$, it is $b_i = -1 < 0$. Let $K := \{i\}$ and $T := S \setminus \{i\}$. Then $b_i \cdot \mathsf{in}_i(T) - \xi_i(T) = 0 > b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$ contradicts S being strong.
     – If there exists $i \in Q(\boldsymbol{b})$ with $i \notin S$, $b_i = b_i^+$. Define sets $K := \{i\}$ and $T := S \cup \{i\}$. Now, $b_i \cdot \mathsf{in}_i(T) - \xi_i(T) > 0 = b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$ yields again a contradiction to S being strong. (Note to ensure that $b_i \cdot \mathsf{in}_i(T) - \xi_i(T) > 0$ we can simply assume that b_i^+ is large enough.)
   - Induction hypothesis: (C.8) holds for all $\boldsymbol{b} \in \mathbb{R}^n$ with $\kappa(\boldsymbol{b}) > k$.

- Now choose $\boldsymbol{b}$ with $\kappa(\boldsymbol{b}) = k$. Without loss of generality, assume $b_i \in \{-1, b_i^+\}$ for all $i \in [k]$. We know by (1.) that $Q(\boldsymbol{b})$ is a strong set for ξ and $\boldsymbol{b}$. Consider another strong set S for ξ and $\boldsymbol{b}$. We show

$$\text{for all } i \in [n] : \; b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) = b_i \cdot \mathsf{in}_i(S) - \xi_i(S) \; . \tag{C.9}$$

Due to the choice of $\boldsymbol{b}$, it is $\mathsf{in}_i(Q(\boldsymbol{b})) = \mathsf{in}_i(S)$ for all $i \in [k]$. Otherwise, we can obtain a contradiction to S being strong, by the same reasoning as in the induction base case.

For an arbitrary $j \in \{k+1, \ldots, n\}$, we define a new vector $\boldsymbol{d}$ as

$$\boldsymbol{d} := \begin{cases} (b_j^+, \boldsymbol{b}_{-j}) & \text{if } j \in S \\ (-1, \boldsymbol{b}_{-j}) & \text{otherwise.} \end{cases} \tag{C.10}$$

We show that S is also strong for ξ and $\boldsymbol{d}$, i.e., there are no sets $K, T \subseteq [n]$ such that

$$\text{for all } i \notin K : i \in T \Leftrightarrow i \in S \tag{C.11}$$

$$\text{for all } i \in K : d_i \cdot \mathsf{in}_i(T) - \xi_i(T) \geq d_i \cdot \mathsf{in}_i(S) - \xi_i(S) \tag{C.12}$$

$$\text{with at least one strict inequality}$$

If there are T, K with $j \notin K$ that meet the conditions, we get a contradiction to S being strong for ξ and $\boldsymbol{b}$, since $d_i = b_i$ for all $i \neq j$. Now assume that there are T, K with $j \in K$ that meet the conditions. If j strictly improves, i.e., $d_j \cdot \mathsf{in}_j(T) - \xi_j(T) > d_j \cdot \mathsf{in}_j(S) - \xi_j(S)$, it follows that $d_j = b_j^+$, $j \in S \cap T$ and $\xi_j(T) < \xi_j(S)$. This again yields a contradiction to S being strong for ξ and $\boldsymbol{b}$. Thus, S is also strong for ξ and $\boldsymbol{d}$. By (1.), $Q(\boldsymbol{d})$ is another strong set for ξ and $\boldsymbol{d}$, and by induction assumption

$$\text{for all } i \in [n] : \; d_i \cdot \mathsf{in}_i(Q(\boldsymbol{d})) - \xi_i(Q(\boldsymbol{d})) = d_i \cdot \mathsf{in}_i(S) - \xi_i(S) \; . \tag{C.13}$$

By definition of $\boldsymbol{d}$, $j \in Q(\boldsymbol{d}) \Leftrightarrow j \in S$. By (C.13), $\xi_j(Q(\boldsymbol{d})) = \xi_j(S)$ and therefore $b_j \cdot \mathsf{in}_j(Q(\boldsymbol{d})) - \xi_j(Q(\boldsymbol{d})) = b_j \cdot \mathsf{in}_j(S) - \xi_j(S)$. Furthermore, if it holds that $b_j \cdot \mathsf{in}_j(Q(\boldsymbol{b})) - \xi_j(Q(\boldsymbol{b})) < b_j \cdot \mathsf{in}_j(Q(\boldsymbol{d})) - \xi_j(Q(\boldsymbol{d}))$ then it follows that $u_j(\boldsymbol{b}, b_j) < u_j(\boldsymbol{d}, b_j)$, contradicting SP with true valuation b_j. Since our choice for j was arbitrary from the set $\{k+1, \ldots, n\}$, it holds for *all* $j \in \{k+1, \ldots, n\}$ that

$$b_j \cdot \mathsf{in}_j(Q(\boldsymbol{b})) - \xi_j(Q(\boldsymbol{b})) \geq b_j \cdot \mathsf{in}_j(S) - \xi_j(S) \; . \tag{C.14}$$

Now assume that there exists $j \in \{k+1, \ldots, n\}$ such that (C.14) holds with a strict inequality. With $T = Q(\boldsymbol{b})$ and $K = \{k+1, \ldots, n\}$ we obtain a contradiction to S being strong for ξ and $\boldsymbol{b}$. Thus (C.14) holds with equality for all $j \in \{k+1, \ldots, n\}$. We are left with showing that (C.9) holds for all $i \in [k]$.

If there exists $i \in [k]$ with $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) < b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$, we get by (C.13) that $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) < b_i \cdot \mathsf{in}_i(Q(\boldsymbol{d})) - \xi_i Q(\boldsymbol{d}))$, for an arbitrary $j \in \{k+1, \ldots, n\}$ and $\boldsymbol{d}$ as defined in C.10. Thus, $u_i(\boldsymbol{b}, b_i) < u_i(\boldsymbol{d}, b_i)$ and $u_j(\boldsymbol{b}, b_j) = b_j \cdot \mathsf{in}_j(Q(\boldsymbol{b})) - \xi_j(Q(\boldsymbol{b})) = b_j \cdot \mathsf{in}_j(Q(\boldsymbol{d})) - \xi_j(Q(\boldsymbol{d})) = u_j(\boldsymbol{d}, b_j)$. With true valuations b_i, b_j, we obtain a contradiction to GSP.

Thus, for all $i \in [k]$, $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) \geq b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$, and for all $i \in \{k+1, \ldots, n\}$, $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) = b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$. There cannot be an $i \in [k]$ with $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) > b_i \cdot \mathsf{in}_i(S) - \xi_i(S)$, since $K := [n]$ and $T := Q(\boldsymbol{b})$ yield a contradiction to S being strong for ξ and $\boldsymbol{b}$.

3. Third, we show that for all $i, j \in [n]$ with $i \neq j$ and all sets $S \subseteq [n] \setminus \{i, j\}$, at least one of the conditions (C.15) – (C.17) holds.

$$\xi_i(S \cup \{i,j\}) = \xi_i(S \cup \{i\}) \tag{C.15}$$
$$\xi_j(S \cup \{i,j\}) = \xi_j(S \cup \{j\}) \tag{C.16}$$
$$\xi_i(S \cup \{i,j\}) < \xi_i(S \cup \{i\}) \text{ and } \xi_j(S \cup \{i,j\}) < \xi_j(S \cup \{j\}) \ . \tag{C.17}$$

Assume that there is a set $S \subseteq [n] \setminus \{i, j\}$ such that none of the conditions (C.15) – (C.17) holds for S. Then one of the following has to hold:

$$\xi_i(S \cup \{i,j\}) < \xi_i(S \cup \{i\}) \text{ and } \xi_j(S \cup \{i,j\}) > \xi_j(S \cup \{j\}) \tag{C.18}$$
$$\xi_i(S \cup \{i,j\}) > \xi_i(S \cup \{i\}) \text{ and } \xi_j(S \cup \{i,j\}) < \xi_j(S \cup \{j\}) \tag{C.19}$$
$$\xi_i(S \cup \{i,j\}) > \xi_i(S \cup \{i\}) \text{ and } \xi_j(S \cup \{i,j\}) > \xi_j(S \cup \{j\}) \tag{C.20}$$

Assume that (C.18) holds. Define $\boldsymbol{d}$ by $d_k := b_k^+$ for all $k \in S$, $d_k := -1$ for all $k \notin S \cup \{i, j\}$, and choose d_i, d_j such that $\xi_i(S \cup \{i,j\}) < d_i < \xi_i(S \cup \{i\})$ and $\xi_j(S \cup \{i,j\}) > d_j > \xi_j(S \cup \{j\})$. By (1.) we know that $Q(\boldsymbol{d})$ is strong for ξ and $\boldsymbol{d}$.

- If $i \in Q(\boldsymbol{d})$ and $j \in Q(\boldsymbol{d})$, then with $K := \{j\}$ and $T := Q(\boldsymbol{d}) \setminus \{j\}$, we have for all $k \neq j$ that $k \in T \Leftrightarrow k \in Q(\boldsymbol{d})$ and

  $$d_j \cdot \mathsf{in}(j, T) - \xi_j(T) = 0 > d_j - \xi_j(S \cup \{i,j\}) = d_j \cdot \mathsf{in}(j, Q(\boldsymbol{d})) - \xi_j(Q(\boldsymbol{d})) \ .$$

  However, this contradicts that $Q(\boldsymbol{d})$ is strong for ξ and $\boldsymbol{d}$.
- If $i \in Q(\boldsymbol{d})$ and $j \notin Q(\boldsymbol{d})$, then with $K := \{i\}$ and $T := Q(\boldsymbol{d}) \setminus \{i\}$, we have for all $k \neq i$ that $k \in T \Leftrightarrow k \in Q(\boldsymbol{d})$ and

  $$d_i \cdot \mathsf{in}_i(T) - \xi_i(T) = 0 > d_i - \xi_i(S \cup \{i\}) = d_i \cdot \mathsf{in}_i(Q(\boldsymbol{d})) - \xi_i(Q(\boldsymbol{d})) \ .$$

  This poses a contradiction to $Q(\boldsymbol{d})$ being strong for ξ and $\boldsymbol{d}$. The case $i \notin Q(\boldsymbol{d})$ and $j \in Q(\boldsymbol{d})$ analogously leads to this contradiction.
- If $i \notin Q(\boldsymbol{d})$ and $j \notin Q(\boldsymbol{d})$ then with $K := \{i, j\}$ and $T := Q(\boldsymbol{d}) \cup \{i, j\}$, we have for all $k \notin \{i, j\}$ that $k \in T \Leftrightarrow k \in Q(\boldsymbol{d})$ and

  $$d_i \cdot \mathsf{in}_i(T) - \xi_i(T) = d_i - \xi_i(S \cup \{i,j\}) > 0 = d_i \cdot \mathsf{in}_i(Q(\boldsymbol{d})) - \xi_i(Q(\boldsymbol{d})) \ ,$$

  analogously for j. Again, this is a contradiction to $Q(\boldsymbol{d})$ being strong for ξ and $\boldsymbol{d}$.

As a consequence (C.18) cannot hold. By exchanging the roles of i and j, also (C.19) cannot hold. Thus, we are left to investigate (C.20). Define $\boldsymbol{d}$ for all $k \notin \{i, j\}$ as before and let d_i, d_j be such that $\xi_i(S \cup \{i,j\}) > d_i > \xi_i(S \cup \{i\})$ and $\xi_j(S \cup \{i,j\}) > d_j > \xi_j(S \cup \{j\})$. It is not hard to verify that $S \cup \{i\}$ and $S \cup \{j\}$ are both strong for ξ and $\boldsymbol{d}$. However,

$$d_i \cdot \mathsf{in}_i(S \cup \{i\}) - \xi_i(S \cup \{i\}) > 0 = d_i \cdot \mathsf{in}_i(S \cup \{j\}) - \xi_i(S \cup \{j\}) \ ,$$

a contradiction to (C.8) in (2.).
Thus, at least one of the conditions (C.15) – (C.17) holds.

4. Forth, we show that ξ is cross-monotonic, by induction on n.

- Let $n = 2$. We first show that for all $S \subseteq [n]$ with $1 \in S$, it holds that $\xi_1(S \cup \{2\}) \leq \xi_1(S)$. In case that $S = \{1, 2\}$, equality holds. Let $S = \{1\}$. Due to (C.15)–(C.17), either $\xi_1(\{1,2\}) = \xi_1(\{1\})$, or $\xi_2(\{1,2\}) = \xi_2(\{2\})$, or $\xi_1(\{1,2\}) < \xi_1(\{1\})$ and $\xi_2(\{1,2\}) < \xi_2(\{2\})$. The only case to consider is $\xi_2(\{1,2\}) = \xi_2(\{2\})$. We utility the submodularity of C and 1-BB of M to obtain
$$\begin{aligned} &C(\{1,2\}) - C(\{1\}) - C(\{2\}) + C(\emptyset) = 0 \\ \Rightarrow\ &\xi_1(\{1,2\}) + \xi_2(\{1,2\}) - \xi_1(\{1\}) - \xi_2(\{2\}) = 0 \\ \Rightarrow\ &\xi_1(\{1,2\}) = \xi_1(\{1\}) \ . \end{aligned}$$
The fact that $\xi_2(S \cup \{1\}) \leq \xi_2(S)$ for all $S \subseteq [n]$ with $2 \in S$, holds analogously.
- Induction hypothesis: ξ is cross-monotonic for all $k < n$.
- Assume without loss of generality that $\xi_2([n]) > \xi_2([n] \setminus \{1\})$. We will obtain a contradiction to the submodularity of C, thereby proving cross-monotonicity. First assume that there is a $j \in [n] \setminus \{1, 2\}$ with $\xi_j([n]) < \xi_j([n] \setminus \{1\})$. Define true valuations $\boldsymbol{v}$ by setting $v_1 := \xi_1([n])$ and $v_i := b_i^+$ for all $i \in [n] \setminus \{1\}$. It is $Q(\boldsymbol{v}) \in \{[n], [n] \setminus \{1\}\}$. If $Q(\boldsymbol{v}) = [n]$, then $u_1(((-\mathbf{1})_1, \boldsymbol{v}_{-1}), v_1) = 0 = u_1(\boldsymbol{v}, v_1)$ and $u_2(((-\mathbf{1})_1, \boldsymbol{v}_{-1}), v_2) > u_2(\boldsymbol{v}, v_2)$ contradicts GSP. If $Q(\boldsymbol{v}) = [n] \setminus \{1\}$, then $u_1((b_1^+, \boldsymbol{v}_{-1}), v_1) = 0 = u_1(\boldsymbol{v}, v_1)$ and $u_j((b_1^+, \boldsymbol{v}_{-1}), v_j) > u_j(\boldsymbol{v}, v_j)$, a contradiction to GSP as well. Thus, for all $j \in [n] \setminus \{1, 2\}$, $\xi_j([n]) \geq \xi_j([n] \setminus \{1\})$ leading to
$$C([n] \setminus \{1\}) = \sum_{i=2}^{n} \xi_i([n] \setminus \{1\}) < \sum_{i=2}^{n} \xi_i([n]) = C([n]) - \xi_1([n]) \ . \qquad \text{(C.21)}$$
Since one of the cases (C.15) – (C.17) has to hold and we assume that $\xi_2([n]) > \xi_2([n] \setminus \{1\})$, it follows that $\xi_1([n]) = \xi_1([n] \setminus \{2\})$. Then by induction assumption, 1-BB, and (C.21),
$$\begin{aligned} C([n] \setminus \{2\}) &= \xi_1([n] \setminus \{2\}) + \sum_{i=3}^{n} \xi_i([n] \setminus \{2\}) \\ &\leq \xi_1([n] \setminus \{2\}) + \sum_{i=3}^{n} \xi_i([n] \setminus \{1, 2\}) \\ &= \xi_1([n]) + C([n] \setminus \{1, 2\}) \\ &< C([n]) - C([n] \setminus \{1\}) + C([n] \setminus \{1, 2\}) \ , \end{aligned}$$
contradicting submodularity of C. Thus, ξ is cross-monotonic.

5. Fifth and last, consider $Moulin_\xi = (Q', \xi)$. Since ξ is cross-monotonic by (4.), $Moulin_\xi$ is GSP because of Theorem 4.1. As (1.) only utilized GSP of M, we can conclude that for every bid vector $\boldsymbol{b} \in \mathbb{R}^n$ not only $Q(\boldsymbol{b})$ but also $Q'(\boldsymbol{b})$ is a strong set for ξ and $\boldsymbol{b}$. Then by (2.), $b_i \cdot \mathsf{in}_i(Q(\boldsymbol{b})) - \xi_i(Q(\boldsymbol{b})) = b_i \cdot \mathsf{in}_i(Q'(\boldsymbol{b})) - \xi_i(Q'(\boldsymbol{b}))$ for all $i \in [n]$, i.e., M and $Moulin_\xi$ are welfare equivalent.

□

Corollary C.4. [97] *For any GSP mechanism M that employs cross-monotonic cost shares specified by ξ, $Moulin_\xi$ is welfare equivalent to M.*

Proof. The corollary simply follows from the proof of Theorem C.3 as submodularity and 1-BB are only exploited for showing that cost-shares of a GSP and 1-BB mechanism are cross-monotonic.

D

Further Optimization Problems

- SpanningTree:
  - *Input*: set of players $[n]$, complete undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, vector $\boldsymbol{s} \in V^n$ representing a node s_i for each player i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: subtree T that connects all nodes in $\{s_i\}_{i \in S}$ and only contains edges with both endpoints in $\{s_i\}_{i \in S}$.
  - *Objective*: minimize the weight of T.
- SteinerTree:
  - *Input*: set of players $[n]$, connected undirected graph $G = (V, E)$ with metric edge weights $w_e \in N$, vector $\boldsymbol{s} \in V^n$ representing a node s_i for each player i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: subtree T that connects all nodes in $\{s_i\}_{i \in S}$.
  - *Objective*: minimize the weight of T
- PriceCollectingSteinerForest:
  - *Input*: set of players $[n]$, undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, vectors $\boldsymbol{s} \in V^n$ and $\boldsymbol{t} \in V^n$ represening pairs (s_i, t_i) of nodes for each player i, penalty vector $\boldsymbol{p} \in \mathbb{N}^n$, subset of players $S \subseteq [n]$.
  - *Feasible solution*: forest F and subset $Q \subseteq [n]$ such that for all $i \in S$, s_i and t_i are either connected by F or $i \in Q$.
  - *Objective*: minimize the weight of F plus the penalties of the players in Q, i.e. of the not-connected pairs in S.
- ConnectedFacilityLocation:
  - *Input*: set of players $[n]$, set of facilities F, opening costs $o_f \in \mathbb{N}$ for each facility, a metric $d : ([n] \cup F) \times ([n] \cup F) \to \mathbb{N}$ specifying the distances between all pairs of players and facilities, a parameter $M \geq 1$, subset of players S.
  - *Feasible solution*: set of facilities F and a Steiner tree T connecting all facilities in F.
  - *Objective*: minimize the sum of the opening costs of F, the connection costs of players in S to a closest open facility in F, and M times the weight of T.
- TravelingSalesman
  - *Input*: set of players $[n]$, complete undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, vector $\boldsymbol{s} \in V^n$ representing a node s_i for each player i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: hamiltonian cycle H for the nodes in $\{s_i\}_{i \in S}$, i.e., a cycle that visits all nodes in $\{s_i\}_{i \in S}$ exactly once, returns to the starting vertex, and only contains edges with both endpoints in $\{s_i\}_{i \in S}$.
  - *Objective*: minimize the weight of H.

- VertexCover:
  - *Input*: set of players $[n]$, undirected and unweighted graph $G = (V, E)$, vector $\boldsymbol{s} \in E^n$ representing an edge s_i for each agent i, subset of players $S \subseteq [n]$
  - *Feasible solution*: set of vertices Z such that each edge in $\{s_i\}_{i \in S}$ has at least one endpoint in Z.
  - *Objective*: minimize the cardinality of Z.
- SetCover:
  - *Input*: set of players $[n]$, set of elements U, set $\{U_1, \ldots, U_k\}$ of subsets of U with $\bigcup_i U_i = U$, vector $\boldsymbol{s} \in U^n$ representing an element $s_i \in U$ for each player i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: subset U' of $\{U_1, \ldots, U_k\}$ such that $\bigcup_{S \in U'} S \supseteq \{s_i\}_{i \in S}$.
  - *Objective*: minimize the number of sets in U'.
- EdgeCover:
  - *Input*: set of players $[n]$, undirected and unweighted graph $G = (V, E)$, vector $\boldsymbol{s} \in V^n$ representing an vertex s_i for each agent i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: set of edges F such that each vertex in $\{s_i\}_{i \in S}$ is adjacent to an edge in F.
  - *Objective*: minimize the cardinality of F.
- SingleSourceRentOrBuy:
  - *Input*: set of players $[n]$, connected undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, root vertex r, vector $\boldsymbol{s} \in V^n$ representing a node s_i for each agent i, parameter $M \geq 1$, subset of players $S \subseteq [n]$.
  - *Feasible solution*: set E_B of *bought* edges, set E_R of *rented* edges, such that for all $i \in S$, there is a path from i to r using only edges in $E_B \cup E_R$. Specify these paths by P.
  - *Cost*: The cost of $e \in E_B$ is $M \cdot w_e$. An $e \in E_R$ costs $w_e \cdot \lambda(P, e)$, where $\lambda(P, e)$ denotes the number of paths in P traversing e.
  - *Objective*: minimize the cost.
- MulticommodityRentOrBuy:
  - *Input*: set of players $[n]$, connected undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, vectors $\boldsymbol{s} \in V^n$, $\boldsymbol{t} \in V^n$ and $\boldsymbol{f} \in \mathbb{N}^n$ represening pairs (s_i, t_i) of nodes and a flow requirement f_i for each agent i, parameter $M \geq 1$, subset of players $S \subseteq [n]$.
  - *Feasible solution*: set E_B of *bought* edges, set E_R of *rented* edges such that for all $i \in S$, we can route flow f_i from s_i to t_i using only edges in $E_B \cup E_R$. Specify this flow by F.
  - *Cost*: The cost of a bought edge $e \in E_B$ is $M \cdot w_e$. A rented edges $e \in E_R$ costs $w_e \cdot \lambda(F, e)$, where $\lambda(F, e)$ denotes the total flow in F traversing e.
  - *Objective*: minimize the cost.
- Multicast:
  - *Input*: set of players $[n]$, connected undirected graph $G = (V, E)$ with metric edge weights $w_e \in \mathbb{N}$, root vertex r, vector $\boldsymbol{s} \in V^n$ representing a node s_i for each agent i, subset of players $S \subseteq [n]$.
  - *Feasible solution*: Steiner tree T rooted at r connecting all the nodes in $\{s_i\}_{i \in S}$.
  - *Objective*: minimize the weight of T.
  - *Note*: It is often assumed that a fixed tree for all players $[n]$ is given and that the cost for serving $S \subseteq [n]$ is simply the cost of the smallest subtree containing S. Thus, there is essentially no optimization problem and the focus lies on computing specific cost-shares (also in a distributed way).

References

[1] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem in networks. *SIAM Journal on Computing*, 24(3):445–456, 1995.

[2] N. Andelman, Y. Azar, and M. Sorani. Truthful approximation mechanisms for scheduling selfish related machines. In *Proceedings of the 22nd Int. Symposium on Theoretical Aspects of Computer Science*, pages 69–82, 2005.

[3] A. Archer. Mechanisms for discrete optimization with rational agents, 2004. PhD Thesis, Cornell University.

[4] A. Archer, J. Feigenbaum, A. Krishnamurthy, and R. Sami. Approximation and collusion in multicast cost sharing. *Games and Economic Behaviour*, 47(1):36–71, 2004.

[5] A. Archer and É. Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42th IEEE Symposium on Foundations of Computer Science*, pages 482–491, 2001.

[6] V. Auletta, R. De Prisco, P. Penna, G. Persiano, and C. Ventre. New constructions of mechanisms with verification. In *Proceedings of the 33rd Int. Colloquium on Automata, Languages and Programming*, pages 596–607, 2006.

[7] V. Auletta, R. De Prisco, P. Penna, and P. Persiano. The power of verification for one-parameter agents. In *Proceedings of the 31st Int. Colloquium on Automata, Languages and Programming*, pages 171–182, 2004.

[8] R. Aumann. *Lectures on Game Theory.* Westview Press, 1989.

[9] M. Bern and P. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(1):171–176, 1989.

[10] Y. Bleischwitz and B. Monien. Fair cost-sharing methods for scheduling jobs on parallel machines. In *Proceedings of the 6th Italian Conference on Algorithms and Complexity*, pages 175–186, 2006. To appear in Journal of Discrete Algorithms.

[11] Y. Bleischwitz, B. Monien, and F. Schoppmann. To be or not to be (served). In *Proceedings of the 3rd Int. Workshop on Internet and Network Economics*, pages 515–528, 2007.

[12] Y. Bleischwitz, B. Monien, F. Schoppmann, and K. Tiemann. The power of two prices: Beyond cross-monotonicity. In *Proceedings of the 32th Int. Symposium on Mathematical Foundations of Computer Science*, pages 657–668, 2007.

[13] Y. Bleischwitz and F. Schoppmann. Group-strategyproof cost sharing for metric fault tolerant facility location. *To appear in Proceedings of the 1st Int. Symposium on Algorithmic Game Theory*, 2008.

[14] Y. Bleischwitz and F. Schoppmann. New efficiency results for makespan scheduling. *To appear in Information Processing Letters*, 2008.

[15] O. Bondareva. Some applications of linear programming to cooperative games. *Problemy Kibernetiki*, 10:119–139, 1963.

[16] J. Brenner and G. Schäfer. Cost sharing methods for makespan and completion time scheduling. In *Proceedings of the 24th Int. Symposium on Theoretical Aspects of Computer Science*, pages 670–681, 2007.

[17] P. Bruckner. *Scheduling Algorithms.* Springer, 1995.

[18] J. Byrka. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. In *Proceedings of the 10th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 29–43, 2007.

[19] S. Chawla, T. Roughgarden, and M. Sundararajan. Optimal cost-sharing mechanisms for Steiner forest problems. In *Proceedings of the 2nd Int. Workshop on Internet and Network Economics*, pages 112–123, 2006.

[20] B. Chen. Tighter bounds for MULTIFIT scheduling on uniform processors. *Discrete Mathematics*, 31(3):227–260, 1991.

[21] R.T. Chien. Synthesis of a communication net. *IBM Journal of Research and Development*, 3:311–320, 1960.

[22] G. Christodoulou, E. Koutsoupias, and A. Vidali. A lower bound for scheduling mechanisms. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, pages 1163–1170, 2007.

[23] F. A. Chudak. Improved approximation algorithms for uncapacitated facility location. *Integer Programming and Combinatorial Optimization*, pages 180–194, 1998.

[24] Y. Chun. The solidarity axiom for quasi-linear social choice problems. *Social Choice and Welfare*, 3:297–310, 1986.

[25] E. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971.

[26] E. G. Coffman Jr., M. R. Garey, and D. S. Johnson. An application of bin-packing to multiprocessor scheduling. *Journal on Computing*, 1(7):1–17, 1978.

[27] E. G. Coffman Jr., M. R. Garey, and D. S. Johnson. Bin packing with divisible item sizes. *Journal of Complexity*, 3(4):406–428, 1987.

[28] E. G. Coffman Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. *Chapter 2 of 'Approximation Algorithms for Hard Problems', D. Hochbaum (editor)*, 1996.

[29] A. Czumaj. Selfish routing on the internet. *Chapter 42 of 'Handbook of Scheduling: Algorithms, Models, and Performance Analysis', J. Leung (editor)*, 2004.

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

[31] N. R. Devanur, M. Mihail, and V. V. Vazirani. Strategyproof cost-sharing mechanisms for set cover and facility location games. *Decision Support Systems*, 39(1):11–22, 2005.

[32] G. Dósa. The tight bound of first fit decreasing bin-packing algorithm is FFD(I) $\leq$ (11/9) OPT(I) + 6/9. In *Proceedings of the 1st Int. Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, pages 1–1, 2007.

[33] B. Dutta. The egalitarian solution and reduced game properties in convex games. *Int. Journal of Game Theory*, 19(2):153–169, 1990.

[34] B. Dutta and D. Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57(3):615–635, 1989.

[35] F.Y. Edgeworth. *Mathematical Psychics.* Kegan Paul Publishers, 1881.

[36] J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Hardness results for multicast cost sharing. *Theoretical Computer Science*, 304(1–3):215–236, 2003.

[37] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, 2001.

[38] W. Fernandez de la Vega and G.S. Lueker. Bin packing can be solved within 1 + epsilon in linear time. *Combinatorica*, 1(4):349–355, 1981.

[39] G.P. Ferrante, F. Sorrentino, and C. Ventre. Improvements for truthful mechanisms with verifiable one-parameter selfish agents. In *Proceedings of the 3rd Workshop on Approximation and Online Algorithms*, pages 147–160, 2006.

[40] S. Fischer and B. Vöcking. On the structure and complexity of worst-case equilibria. In *Proceedings of the 1st Workshop on Internet and Network Economics*, pages 151–160, 2005.

[41] S. Fischer and B. Vöcking. On the structure and complexity of worst-case equilibria. *Theoretical Computer Science*, 378(2):165–174, 2007.

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

[43] D.K. Friesen. Tighter bounds for the multifit processor scheduling algorithm. *SIAM Journal on Computing*, 13(1):170–181, 1984.

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

[45] D.K. Friesen and M.A. Langston. Bounds for multifit scheduling on uniform processors. *SIAM Journal on Computing*, 12(1):60–70, 1983.

[46] D.K. Friesen and M.A. Langston. Evaluation of a MULTIFIT based scheduling algorithm. *Journal of Algorithms*, 7(1):35–59, 1986.

[47] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. Computing Nash equilibria for scheduling on restricted parallel links. In *Proceedings of the 36th ACM Symposium on Theory of Computing*, pages 613–622, 2004.

[48] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. Spirakis. Extreme Nash equilibria. In *Proceedings of the 8th Italian Conference on Theoretical Computer Science*, pages 1–20, 2003.

[49] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. Spirakis. Structure and complexity of extreme Nash equilibria. *Theoretical Computer Science*, 343(1–2):133–157, 2005.

[50] M. Gairing, T. Lücking, B. Monien, and K. Tiemann. Nash equilibria, the price of anarchy and the fully mixed Nash equilibrium conjecture. In *Proceedings of the 32nd Int. Colloquium on Automata, Languages and Programming*, pages 51–65, 2005.

[51] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[52] M.R. Garey and D.S. Johnson. 'Strong' np-completeness results: Motivation, examples, and implications. *Journal of the ACM*, 25(3):499–508, 1978.

[53] D. Gillies. Solutions to general non-zero-sum games. *Annals of Mathematical Studies, 40*, pages 47–85, 1969.

[54] M. X. Goemans and M. Skutella. Cooperative facility location games. *Journal of Algorithms*, 50(2):194–214, 2004.

[55] M.X. Goemans and D. Bertsimas. Survivable networks, linear programming relaxations and the parsimonious property. *Mathematical Programming*, 60(1–3):145–166, 1993.

[56] M.X. Goemans and D.P. Williamson. A general approximation technique for constrained forest problems. In *Proceedings of the 3rd ACM-SIAM Symposium on Discrete Algorithms*, pages 307–315, 1992.

[57] R.E. Gomory and T.C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[58] R. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.

[59] R. Graham, E. Lawler, J. Lenstra, and A. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[60] J. Green and J. Laffont. Characterizations of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica*, 45(2):427–438, 1977.

[61] J. Green and J. Laffont. *Incentives in public decision-making.* 1979.

[62] J. Green and J. Laffont. On coalition incentive compatibility. *Review of Economic Studies*, 46(143):243–254, 1979.

[63] T. Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[64] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, pages 649–657, 1998.

[65] S. Guha, A. Meyerson, and K. Munagala. Improved algorithms for fault tolerant facility location. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 636–641, 2001.

[66] A. Gupta, J. Könemann, S. Leonardi, R. Ravi, and G. Schäfer. An efficient cost-sharing mechanism for the prize-collecting Steiner forest problem. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[67] A. Gupta, A. Srinivasan, and É. Tardos. Cost-sharing mechanisms for network design. In *Proceedings of the 7th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 139–152, 2004.

[68] D. Hochbaum and D. Shmoys. Using dual approximation algorithms for scheduling problems: theoretical and practical results. *Journal of the ACM*, 34(1):144–162, 1987.

[69] D. Hochbaum and D. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988.

[70] T. Hokari. Population monotonic solutions on convex games. *Int. Journal of Game Theory*, 29(3):327–338, 2000.

[71] L. Hurwicz. Optimality and informal efficiency in resource allocation processes. *Mathematical Methods in the Social Sciences*, 1960.

[72] L. Hurwicz. On informationally decentralized systems. In *Decision and Organization.* North-Holland, 1972.

[73] N. Immorlica, M. Mahdian, and V. Mirrokni. Limitations of cross-monotonic cost sharing schemes. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 602–611, 2005.

[74] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[75] K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problems. In *Proceedings of the 34th ACM Symposium on the Theory of Computing*, pages 731–740, 2002.

[76] K. Jain and V. Vazirani. Applications of approximate algorithms to cooperative games. In *Proceedings of the 33th ACM Symposium on Theory of Computing*, pages 364–372, 2001.

[77] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the Association for Computing Machinery (ACM)*, 48(2):274–296, 2001.

[78] K. Jain and V.V. Vazirani. An approximation algorithm for the fault tolerant metric facility location problem. *Algorithmica*, 38(3):433–439, 2003.

[79] N. Karmarkar and R.M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Proceedings of the 23rd Symposium on Foundations of Computer Science*, pages 312–320, 1982.

[80] R.M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972.

[81] K. Kent and D. Skorin-Kapov. Population monotonic cost allocation on MSTs. In *Operational Research Proceedings KOI*, pages 43–48, 1996.

[82] J. Könemann, S. Leonardi, and G. Schäfer. A group-strategyproof mechanism for Steiner forests. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 612–619, 2005.

[83] J. Könemann, S. Leonardi, G. Schäfer, and S. van Zwam. From primal-dual to cost shares and back: A stronger LP relaxation for the Steiner forest problem. In *Proceedings of the 32th Int. Colloquium on Automata, Languages, and Programming*, pages 930–942, 2005.

[84] E. Koutsoupias, M. Mavronicolas, and P.G. Spirakis. Approximate equilibria and ball fusion. *Theory of Computing Systems*, 36(6):683–693, 2003.

[85] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Int. Symposium on Theoretical Aspects of Computer Science*, pages 404–413, 1999.

[86] S. Leonardi and G. Schäfer. Cross-monotonic cost-sharing methods for connected facility location games. In *Proceedings of the ACM Conference on Electronic Commerce*, pages 224–243, 2004.

[87] J. Leung, editor. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis.* CRC Press, 2006.

[88] P. Lu and C. Yu. An improved randomized truthful mechanism for scheduling unrelated machines. In *25th Int. Symposium on Theoretical Aspects of Computer Science*, pages 527–538, 2008.

[89] M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani. A greedy facility location algorithm analyzed using dual-fitting. In *Proceedings of the 4th Workshop on Approx. Alg. for Combinatorial Optimization Problems*, pages 127–137, 2001.

[90] M. Mahdian, Y. Ye, and J. Zhang. Improved approximation algorithms for metric facility location problems. In *Proceedings of the 5th Int. Workshop on Approx. Algorithms for Combinatorial Optimization*, pages 229–242, 2002.

[91] E. Maskin. Nash equilibrium and welfare optimality. *Review of Economic Studies*, 66:23–38, 1999. presented at Economic Society in Paris, June 1977.

[92] N. Megiddo. Cost allocation for Steiner trees. *Networks*, 8:1–6, 1978.

[93] K. Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters*, 27(3):125–128, 1988.

[94] A. Mehta, T. Roughgarden, and M. Sundararajan. Beyond Moulin mechanisms. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, pages 1–10, 2007. full version: http://theory.stanford.edu/ tim/papers/bmm.pdf.

[95] H. Moulin. Fair division under joint ownership: Recent results and open problems, 1989. Mimeo, Universitat Autonòma de Barcelona.

[96] H. Moulin. *Cooperative Microeconomics*. Princeton University Press, 1995.

[97] H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999.

[98] H. Moulin. Axiomatic cost and surplus sharing. *Handbook of Social Choice and Welfare*, pages 289–357, 2002.

[99] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001.

[100] Frank D. Murgolo. Anomalous behavior in bin packing algorithms. *Discrete Applied Mathematics*, 21(3):229–243, October 1988.

[101] S. Mutuswami. Strategyproof mechanism for cost sharing, 1997. working paper, Indian statistical institute.

[102] R. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[103] R. Myerson. *Game Theory - Analysis of Conflict*. Harvard University Press, 1991.

[104] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behaviour*, 35(1–2):166–196, 2001.

[105] M. Osborne and A. Rubinstein. *A course in game theory*. MIT Press, 1994.

[106] M. Pál and É. Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science*, pages 584–593, 2003.

[107] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[108] P. Penna and C. Ventre. More powerful and simpler cost-sharing methods (when cross-monotonicity is the wrong way). In *Proceedings of the 2nd Workshop on Approximation and Online Algorithms*, pages 97–110, 2005.

[109] P. Penna and C. Ventre. The algorithmic structure of group strategyproof budget-balanced cost-sharing mechanisms. In *Proceedings of the 23rd Int. Symposium on Theoretical Aspects of Computer Science*, pages 337–348, 2006.

[110] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice Hall, 2002.

[111] R. Reischuk. *Komplexitätstheorie*. Teubner, 1999.

[112] E. Rosenthal. Monotonicity of the core and value in dynamic cooperative games. *Int. Journal of Game Theory*, 19(1):45–57, 1990.

[113] T. Roughgarden and M. Sundararajan. Approximately efficient cost-sharing mechanisms. *arXiv report*, 2006. http://www.arxiv.org/pdf/cs.GT/0606127.

[114] T. Roughgarden and M. Sundararajan. New trade-offs in cost-sharing mechanisms. In *Proceedings of the 38th ACM Symposium on Theory of Computing*, pages 79–88, 2006.

[115] T. Roughgarden and M. Sundararajan. Optimal efficiency guarantees for network design mechanisms. In *Proceedings of the 12th Conference on Integer Programming and Combinatorial Optimization*, pages 469–483, 2007.

[116] L. Shapley. A value for n-person games. *Contributions to the theory of games*, pages 31–40, 1953.

[117] L. S. Shapley. On balanced sets and cores. *Naval Research Logistics Quarterly*, 14:453–460, 1967.

[118] L.S. Shapley. Cores of convex games. *Int. Journal of Game Theory*, 1:11–26, 1971.

[119] D.B. Shmoys, C. Swamy, and R. Levi. Fault-tolerant facility location. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 735–736, 2003.

[120] D.B. Shmoys, É. Tardos, and K.I. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th ACM Symposium on the Theory of Computing*, pages 265–274, 1997.

[121] A. Smith. *An Inquiry into the Nature and Causes of the Wealth of Nations*. Methuen and Co., Ltd., London, 1776.

[122] Y. Sprumont. Population monotonic allocation schemes for cooperative games with transferable utility. *Games and Economic Behavior*, 2(4):378–394, 1990.

[123] M. Sviridenko. An improved approximation algorithm for the metric uncapacitated facility location problem. In *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization*, pages 240–257, 2002.

[124] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, pages 573–577, 1980.

[125] W. Thomson. The fair division of a fixed supply among a growing population. *Mathematics of Operations Research*, 8(3):319–326, 1983.

[126] W. Thomson. Problems of fair division and the egalitarian solution. *Journal of Economic Theory*, 31(2):211–226, 1983.

[127] W. Thomson. Population-monotonic allocation rules. *Social Choice, Welfare and Ethics*, pages 79–124, 1995.

[128] V. Vazirani. The Steiner tree problem and its generalizations. *Approximation Algorithms for Combinatorial Optimization*, pages 33–38, 1998.

[129] V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[130] C. Ventre. Mechanisms with verification for any finite domain. In *Proceedings of the 2nd Int. Workshop on Internet and Network Economics*, pages 37–49, 2006.

[131] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.

[132] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behaviour*. Princeton University Press, 1944.

[133] J. Vygen. Approximation algorithms for facility location problems (lecture notes), Research Institute for Discrete Mathematics, Bonn, 2005.

[134] I. Wegener. *Komplexitätstheorie*. Springer, 2003.

[135] P. Winter. Steiner problem in networks: A survey. *Networks*, 17:129–167, 1985.

[136] P. Winter. Generalized Steiner problem in outerplanar graphs. *BIT*, 25:485–496, 1987.

[137] H. Young. Cost allocation. *Handbook of Game Theory with economic applications*, pages 1193–1235, 1994.

[138] M. Yue. On the exact upper bound for the MULTIFIT processor scheduling algorithm. *Annals of Operations Research*, 24(1–4):233–259, 1990.