Dissertation

Collusion-Resistant Cost-Sharing Mechanisms: Design Techniques, Analyses, Trade-Offs

Florian Schoppmann

Paderborn, 2. Juli 2009

Schriftliche Arbeit zur Erlangung des Grades
Doktor der Naturwissenschaften
an der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

Abstract

How can a system be designed so that autonomous self-interested players behave in a "desirable" way? In this thesis, we study this question in the context of cost-sharing problems, where finitely many players have an unknown valuation for some non-rivalrous but excludable service (e.g., network connectivity). The challenge is to design mechanisms that elicit truthful reports of the players' valuations, determine which set of players Q to serve, and decide how to distribute the incurred service cost $C(Q)$. So in particular, a cost-sharing mechanism has to give players an *incentive* to reveal truthful information. Further constraints for cost-sharing problems include *budget balance* (i.e., recovery of the service cost with the prices charged) and *economic efficiency* (i.e., a reasonable trade-off between the service cost and the excluded players' valuations). Practical applications moreover require that cost-sharing mechanisms are computable in *polynomial time*.

Cost-sharing problems are fundamental in economics and have a broad area of applications; e.g., distributing volume discounts in electronic commerce, sharing the cost of public infrastructure projects, allocating development costs of low-volume built-to-order products, etc. Despite this fundamental nature, general techniques for solving cost-sharing problems are rare. When requiring group-strategyproofness—i.e., collusion resistance in a very strong sense—essentially only one technique has been known, the so-called Moulin mechanisms. Unfortunately, there are several natural cost-sharing problems for which any Moulin mechanism inevitably suffers poor budget balance and economic efficiency.

In this thesis, we devise several alternative techniques for designing cost-sharing mechanisms. We demonstrate the benefits of our novel techniques by applying them to various natural cost-sharing problems where the costs $C(Q)$ are induced by combinatorial optimization problems. Moreover, we provide characterization results that contribute towards understanding the inherent limitations of collusion resistance with respect to the other desirable properties of cost-sharing mechanisms.

Acknowledgments

This thesis would not have been possible without the continuous support of a number of people to whom I wish to express my gratitude here. First and foremost, I want to thank my advisor Burkhard Monien for his support, his steady encouragement, and his insights that have been invaluable to my research. In fact, it also was a course lectured by Burkhard almost four years ago that sparked my interest in algorithmic game theory while still being an undergrad student.

I am furthermore indebted to Yvonne Bleischwitz who used to be the other Ph.D. student in the “cost-sharing team”. I always greatly enjoyed working together with Yvonne, and this feeling has only increased ever since she finished her dissertation and left the group.

Special thanks also go to Martin Gairing and Karsten Tiemann for the fruitful collaboration on the topics of selfish routing. Besides the joint work, Martin has frequently provided me with extremely helpful feedback on various matters. His support reaches back to the time when he was the advisor of my diploma thesis. Together with Karsten I attended more conferences and workshops than with anybody else. It not only was a pleasure to work with him, but also to travel with him to various places in Europe.

During the past three years, I have been lucky to share my office with Tobias Tscheuschner. Discussing research problems with Tobias, as well as the latest news and everything in between, has contributed a lot to making my Ph.D. project delightful and enjoyable.

I am very grateful to Johannes Berendt, Yvonne Bleischwitz, Rainer Feldmann, Martin Gairing, Eva Neels, Jan Rieke, and Ulf-Peter Schroeder for carefully reading preliminary parts of my thesis.

Crucial for my well-being throughout the past years have been my family and my friends. I particularly thank my parents for teaching me to be open-minded and for stimulating a passion for learning at an early age. Finally, I owe my sincerest thanks to Rahel for her exceptional patience throughout the past years. She certainly had to endure the most, and her support has been beyond words.

My research in the past three years was only possible due to a fellowship by the International Graduate School *Dynamic Intelligent Systems* at the University of Paderborn. I wish to thank everybody in the Graduate School team for the support and for organizing the many extra-curricular activities. I enjoyed them a lot. Keep up the good work!

Paderborn, March 2009 *Florian Schoppmann*

Contents

Lists

List of Algorithms

List of Figures

List of Tables

A note on the use of "we" As a matter of style and to acknowledge that many of the results presented in this thesis are based on collaborative work with others, "we" is used throughout the thesis, except in places where the personal voice of the author is expressed.

Chapter 1

Introduction

Present-day information and communication technologies crucially rely on large-scale networks that are created and maintained by a huge number of autonomous players with heterogeneous goals. Evidently, this includes the Internet as the most prominent example, but no less also peer-to-peer, logistics, and social networks. Computer science and research on distributed algorithms in particular have responded to the rapid emergence of these new decentralized networks by a corresponding paradigm shift: Starting roughly a decade ago, computer scientists have increasingly focused on optimization in networks where the self-interested players' behavior cannot be directly controlled [56, 43, 60]. The lack of central control that is encompassed by players' selfishness is instead regarded as an inevitable constraint, similar to the lack of computing power when devising approximation algorithms or the lack of information about the future when designing online algorithms.

Located at the intersection of computer science with mathematics and economics, this rapidly growing field is now routinely called *algorithmic game theory*. Bringing together algorithms and game theory has given rise to several new and important questions that were not rigorously considered before: Can the loss due to selfish behavior be quantified, when compared to a hypothetical optimum? In a more catchy term: What is the *price of anarchy*? What is the complexity to compute stable states, i.e., game-theoretic *equilibria*, or approximations of them? How should systems be designed in order to give players an *incentive* to act in a particular fashion? This thesis is located in the subfield of algorithmic game theory that deals with the last of the previous questions and that we will introduce in the following.

1.1 Algorithmic Mechanism Design

When designing algorithms for traditional optimization problems, computer scientists can often safely assume that their algorithms are run relatively isolated from other parts of the system, sometimes even embedded in a black-box manner. Such an algorithm would be easily interchangeable with any other algorithm that computes feasible solutions, without changing any further aspects of the system beyond the confines of the black box. Correspondingly, the quality of an algorithm has typically been measured only in terms of its computational complexity (occasionally also its space or communication complexity) and its performance according to either worst- or average-case analysis, or—more practically—some reasonable benchmark.

In sharp contrast to devising algorithms as relatively isolated entities, decentralized networks often necessitate algorithms and protocols that work on information received from many different self-interested players. Clearly, these players only reveal what is best for their own benefit. Hence, a system designer has to anticipate that the choice of an algorithm or protocol crucially impacts what information the players will provide. In more drastic words, an algorithm or protocol may have excellent performance—yet, if the players do not adopt it, its implementation will not prove successful.

In economics, research on the implementation of system-wide desirable solutions in the presence of self-interested players is a well-established field called *mechanism design*. In the basic setting, one can imagine that each player has some private information that is relevant to the problem at hand and that has to be elicited, e.g., by *bids*. Now the goal of the system designer is to provide incentives (e.g., monetary transfers) so that revealing *truthful* information is the best strategy players have available.

Auction design is the oldest and arguably the best-known motivation for mechanism design. For instance, a naive way to conduct a sealed-bid auction is to sell the item at stake to the highest bidder, for a price equal to his bid. However, this gives players an incentive to bid less than their maximum willingness to pay—i.e., their true *valuation* for the item. After all, winning only requires bidding strictly more than the second highest bidder. As long as players cannot collude, there is an easy fix from the viewpoint of incentives: In a "second-price" or "Vickrey" auction [75], the winner only has to pay the value of the second highest bid. It is an easy observation that untruthful bidding could not provide any player with a better "net benefit" (i.e., valuation for the item minus payment if the player won the auction, and zero otherwise). Note, however, that Vickrey auctions are not resistant against coordinated manipulation: In particular, the winner would have an increased net benefit if he had convinced his competitors to bid less than their true valuations.

The traditional application of game theory has been to predict outcomes of strategic interaction, i.e., to identify the states of the game where every player is unlikely to change his strategy. Arguably the most important solution concept here is the *Nash equilibrium*, that is a state in which no player can unilaterally improve his utility by changing his strategy, provided that all other players keep theirs unchanged. In mechanism design, which is sometimes referred to as "inverse game theory" [60], the task is often complementary: Design a game so that the state in which every player bids truthfully is *always* a Nash equilibrium, for every possible combination of the true valuations. Hence, since every state of the game could possibly be the truth, this requirement means that revealing truthful information is even a *dominant strategy* for every player, i.e., the best alternative *regardless* of what the other players are doing. Clearly—as long as one can rule out collusion—such a dominant strategy equilibrium is a robust solution concept because no player ever has to reason about his competitors.

Indeed, it contributes much to the importance and elegance of mechanism design that suitable incentives provide the players with essentially effortless decision-making. Obviously, this gives rise to many applications for the continuing increase in electronic commerce and makes mechanism design an invaluable tool, e.g., to support automated negotiations among groups of individuals and businesses.

Algorithmic mechanism design [56] constitutes a synthesis of mechanism design with algorithmic methods for dealing with computationally intensive challenges, with the ultimate goal of comprehending the necessary trade-off between incentives, computational complexity, and approximation guarantees.

1.2 Cost Sharing

In this thesis, we work towards a more thorough understanding of a particular class of algorithmic mechanism design problems, where the cost of a joint project is to be shared between the participants. For illustration and motivation, we give two made-up and arguably simplified examples.

1.2.1 Examples

Inclusion in a Schedule Several logistics companies are interested in operating Sunday flights at a cargo airport that has previously been used only six days a week. Specifically, there are numerous possible flights with different requirements of ground handling, yet only a limited number of terminals that allow processing at different speeds. How can the overall cost for operating the airport infrastructure also on Sundays be shared? This cost is assumed to be roughly proportional to the total time that there is at least one aircraft needing ground handling. Every company has a maximum willingness to contribute to the cost as there are several outside options: They could switch to another day, to another airport, or they might use other means of transport altogether. So a mechanism is sought that elicits truthful bids and then decides which flights are included in the Sunday schedule and at what price (respecting the bids).

There are several challenges: Most naturally, the cost shares need to recover the total cost. Moreover, the schedule should be “efficient” in an economic sense, meaning that for each flight we assume an “external cost” when not including the flight in the schedule. Therefore, a *scheduling problem with rejection* needs to be solved (see also Section 2.3.1): Find a schedule that minimizes the overall schedule length (also called the *makespan*) plus the external costs for the rejected flights. In particular, economic efficiency implies that a trivial solution such as the empty schedule is typically not a good solution.

Network Connectivity The residents of a remote village—long ignored by the telecommunication industry—wish to get out of the communication stone age by providing high-speed Internet connectivity for every home that is interested. The willingness to contribute to the construction and set-up costs is, however, different among the residents. For instance, whereas some people have long been in desperate need of being able to work from home over the Internet, some others do not even own a computer. Thus, the local government is looking for a mechanism that elicits truthful bids and then decides which homes participate and at what price.

Laying cables to provide the selected set of homes with high-speed Internet access can be modeled as a “prize-collecting” *Steiner tree problem* (see also Section 2.3.3):

Find a minimum-cost network that connects all participating homes and the Internet service provider in the next city. Here, “prize-collecting” is the usual term to indicate that, similar as in the previous example, there is an external cost for not including homes in the network.

1.2.2 Cost-Sharing Mechanisms

The above scenarios are examples of *cost-sharing problems*: A non-rivalrous but excludable good (i.e., a service such as inclusion in a schedule or connectivity in a network) is to be made available to $n \in \mathbb{N}$ players at non-negative prices. Each player $i \in \{1, \dots, n\}$ is completely characterized by his *valuation* $v_i \in \mathbb{R}$ for receiving the service. A *cost-sharing mechanism* is sought that elicits truthful reports of each player's valuation and then determines both the set of served players $Q \subseteq \{1, \dots, n\}$ and a distribution of the service cost $C(Q) \in \mathbb{R}_{\geq 0}$.

Cost-sharing problems are fundamental in economics and have a broad area of applications, similar in concept to the previous examples. This includes, e.g., distributing volume discounts in electronic commerce, sharing the cost of public infrastructure projects, allocating development costs of low-volume built-to-order products, etc.; see also Moulin and Shenker [52] and their references. As mentioned before, the pivotal constraint when designing cost-sharing mechanisms is *incentive-compatibility* (also called *truthfulness*), meaning that each player has an incentive to act as desired by the provider of the service. In the case of cost sharing, this is to submit truthful bids to the mechanism. The cost-sharing literature typically requires a particular strong notion of truthfulness in that even collusion (i.e., coordinated wrong-bidding) must never be profitable.

There are three essential goals for the design of truthful cost-sharing mechanisms: The first and most natural constraint is, of course, recovery of the service cost. Together with reasonable bounds on the generated surplus, this property is referred to as (approximate) *budget balance*. As a second goal, a mechanism should satisfy (approximate) *economic efficiency*, i.e., guarantee reasonable bounds on the *social cost* by appropriately trading off the service cost and the excluded players' valuations. Finally, for practical applications, the computational complexity of the mechanism must be reasonable. Not least due to this blend of optimization goals from different perspectives, cost sharing has attracted a great deal of interest also in computer science.

1.3 Design Techniques for Cost-Sharing Mechanisms

Truthfulness in Non-Cooperative Settings In the standard cost-sharing model, a player who is served has a utility equal to his valuation minus his payment. If a player is not served, then he will not be charged and his utility is zero. The basic notion of truthfulness, called *strategyproofness* (SP), requires that no player can improve his utility by false bidding when all other bids are kept fixed. Equivalently, it must hold for all possible combinations of true valuations that the state in which every player i truthfully bids v_i is a Nash equilibrium.

Truthfulness in Cooperative Settings A form of manipulation that SP does not rule out is manipulation by coalitions. Resistance against collusion is, however, especially desirable in settings with a large number of players that the provider of the service might not even know; e.g., in the Internet. Here, players often have the means to coordinate deceit that is impossible to discover. Several concepts of collusion resistance are known in the literature, of which we name only two here: A mechanism is called *group-strategyproof* (GSP) if any defection of a coalition that increases some member's utility inevitably decreases the utility of one of its other members. A weaker notion of collusion resistance is *weak group-strategyproofness* (WGSP) that is fulfilled if any defecting coalition has at least one member whose utility does not strictly improve. Equivalently, with a WGSP mechanism, it must hold for all possible combinations of true valuations that the state in which every player i truthfully bids v_i is a *strong equilibrium* [4].

Efficient Solutions The most universal technique for the design of truthful mechanisms (not just for cost sharing) is the class of *Vickrey-Clark-Groves (VCG) mechanisms* [75, 14, 30]. By always picking a set of players so that the sum of the included players' valuations minus the service cost is maximized and by an appropriate payment scheme such that each player's payment does not directly depend on his own bid, these mechanisms are truthful (SP) and satisfy optimal economic efficiency. In fact, Green and Laffont [29] revealed already in the 1970's that under general assumptions[1], the VCG mechanisms are the only class of SP mechanisms with these properties.

Unfortunately, VCG mechanisms are not resistant against collusion and fail in general to provide any guarantees for cost recovery, even when ignoring computational complexity [52] (see also Section 2.2.1 for details). Hence, there is an intrinsic conflict: In general, truthful mechanisms cannot guarantee exact budget balance and optimal economic efficiency at the same time. In fact, Feigenbaum et al. [24] gave simple cost functions for which not even (relative) *approximations* of both budget balance and economic efficiency can be achieved at the same time—presuming that economic efficiency is measured in terms of the traditional *social welfare* (sum of the included players' valuations minus service cost). Roughgarden and Sundararajan [64] observed that this impossibility is due to an incompatibility with the mixed-sign social welfare and suggested an alternative measure of economic efficiency: *Social cost*, defined as the sum of the excluded players' valuations plus the service cost. This measure is clearly an order-preserving transformation of the social welfare, and the absolute error is always the same under both measures. Roughgarden and Sundararajan proved that measuring economic efficiency in terms of the social cost indeed makes the desired bi-criteria (relative) approximation possible; i.e., there is a large class of mechanisms that guarantee both budget balance and economic efficiency within constant approximation factors $\beta \geq 1$ and $\alpha \geq 1$, abbreviated as β-BB and α-EFF.

[1] The set of possible bids is equal to the set of possible valuations, and utilities are quasi-linear (see Section 2.1.2).

Cost-Sharing Methods Due to the unsuitability of VCG mechanisms for cost-sharing problems there is great need for other general design techniques. A straightforward idea is to separate a mechanism into two parts: First, compute the set of served players Q depending on the bids. Then, compute the cost shares using a *cost-sharing method* that only depends on Q, and not on the bids. In fact, it is known that *all* GSP mechanisms could be separated this way [51].

Cooperative game theory has long dealt with the question of dividing costs among coalitions. In particular, in the terminology of cooperative game theory, a cost-sharing method is simply a *value* of the *cooperative game* defined by the costs C (see, e.g., Osborne and Rubinstein [58]). Consequently, many families of cost-sharing methods are known in the literature. We consider three in the following:

A very simple cost-sharing method ξ is to let every player pay the *marginal cost* of including him (according to a fixed order of the players). That is, if the players in Q are numbered $1, \dots, |Q|$, every player $i \in Q$ pays $\xi_i(Q) = C(\{1, 2, \dots, i\}) - C(\{1, 2, \dots, i-1\})$. Another well-known cost-sharing method is the *Shapley value* [71]. It assigns to each player $i \in Q$ his *average* marginal cost, over all orders of the players in Q. A third method is the *egalitarian solution* by Dutta and Ray [22], which is motivated by being as egalitarian as possible. The details of the motivation are beyond the scope of this introduction, so we refer to Dutta and Ray's original article for a formal definition and explanations. When the costs C are *submodular* (meaning that the marginal cost of adding a player to a set Q can only decrease as Q gets larger), the egalitarian solution for a set Q can be 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.

GSP Mechanisms Essentially the only technique for designing GSP cost-sharing mechanisms is due to Moulin [51]. Its main ingredient are *cross-monotonic* cost shares $\xi_i(Q)$ that never increase when the set of served players Q gets larger. Given such a cost-sharing method ξ, a *Moulin mechanism* serves the maximal set of players who can afford their corresponding price—due to cross-monotonicity, a unique maximal set always exists. Algorithmically, this set can be found by simulating an iterative ascending auction: At the beginning, all players are included in Q, and each player $i \in Q$ is offered price $\xi_i(Q)$. If there is a player who cannot afford the price offered to him, he is dropped from Q and a new iteration begins. The auction terminates once each of the remaining players can afford the price offered to him.

The main benefit of Moulin mechanisms is that they reduce the design of GSP mechanisms to finding cross-monotonic cost-sharing methods, which are solely responsible for the mechanism's performance. On the negative side, Immorlica et al. [36] and Roughgarden and Sundararajan [64, 65] showed that there are several natural cost-sharing problems for which any Moulin mechanism inevitably suffers from poor budget balance or/and poor economic efficiency.

Immorlica et al. [36] and Penna and Ventre [62] gave another family of cost-sharing mechanisms that are GSP only if voluntary non-participation is not an option players have.[2] In this thesis, however, we very well assume that players may opt not to participate in order to help others. Consequently, these mechanisms are not GSP according to the definition used in this thesis.

The Role of Indifferent Players An immediate implication of SP is that, by unilateral deviation, each player can only influence whether he receives the service, but not his cost share. In detail, for each player $i \in [n]$ and every fixed combination of his competitors' bids, there has to be some *threshold value* θ_i so that i is served for a price of θ_i when bidding strictly more than θ_i, and i is not served when bidding strictly less. We call a player *indifferent* if he bids exactly his threshold value. Note that SP does not imply a particular rule for what to do with an indifferent player.

As a general rule of thumb, the major intricacy for the design of GSP mechanisms with good performance—and thus for finding alternatives to Moulin mechanisms—is the treatment of these indifferent players: Since the utility of an indifferent player is zero regardless of whether he is served or not, his utility is completely unaffected by his own bid. Consequently, this player is prone to manipulate and help others either by enforcing his inclusion with a very large bid or by prompting his exclusion with a very low bid.

More Flexibility by Relaxing GSP Replacing the fairly strong GSP axiom by demanding only WGSP allows for greater flexibility when designing cost-sharing mechanisms. In particular, manipulation by indifferent players is no longer problematic because their utility does not strictly improve. A general technique for the design of WGSP mechanisms, called *acyclic mechanisms*, is due to Mehta et al. [50]. Their mechanisms are generalizations of Moulin mechanism and are likewise computed by simulating iterative ascending auctions. However, for any set of remaining players, there is a specific order in which prices are offered to the players. Now, whenever a player cannot afford this offer, a new iteration is started prematurely. This way, lack of cross-monotonicity can be "concealed" from the players and truthfulness be preserved, while the added versatility of acyclic mechanisms allows for improved budget balance and economic efficiency.

1.4 Contribution

As discussed in the previous sections, cost-sharing problems involve a number of competing goals, and it is known that not all goals are compatible with each other. As a consequence, trade-offs are inevitable. In this thesis, we devise novel design techniques for cost-sharing mechanisms, analyze their performance guarantees, and characterize the compatibility of various desirable properties. In particular, we make intuitively "small" modifications to the assumptions on players' behavior or to certain design goals. It turns out that some of these modifications allow for very improved performance guarantees,

[2] Technically, these mechanisms do not satisfy *strong consumer sovereignty* (see Section 2.1.2).

whereas others—somewhat counterintuitively—do not. For all new techniques, we demonstrate their benefits by applying them to various natural cost-sharing problems where the costs $C(Q)$ are induced by combinatorial optimization problems.

The rest of this section provides a high-level overview of our results. We will give more detailed and formal summaries at the beginning of the later chapters.

1.4.1 Lexicographic Maximization: Beyond Cross-Monotonicity

We propose a new technique for the design of GSP cost-sharing mechanisms that deliberately treats players unequally by maintaining an order of precedence. The main idea is as follows: Given a cost-sharing method ξ, call a set Q *feasible* if all players contained in it can afford their cost share $\xi_i(Q)$. Then, choose the set of players that *lexicographically* maximizes the vector of all players' utilities, over all feasible sets. In contrast, Moulin mechanisms always choose the feasible set for which *all* players' utilities attain their maximum (note here that this maximum is only well-defined for cross-monotonic cost shares). We show that our new technique allows for improved budget balance. However, the trade-off to make is the unequal treatment of the players, which also entails a loss of economic efficiency.

In detail: Costs are called *symmetric* if they only depend on the size of the served players. They are called *subadditive* if the union of two sets of players is never more costly than the sum of the two stand-alone costs. For costs that satisfy both properties, we devise a novel family of cost-sharing mechanisms that we call *symmetric mechanisms* and that achieve provably better budget balance than any Moulin mechanism could guarantee.

We apply our findings to *scheduling problems*, where each player owns exactly one job and the service is inclusion in a schedule, for processing the jobs on parallel machines. The incurred service cost is the maximum completion time of all jobs—called the *makespan*. Therefore, we give a polynomial-time algorithm that not only determines the set of served players Q and a distribution of the service cost $C(Q)$, but also a feasible schedule for the jobs that belong to the players in Q.

Towards understanding the limitations imposed by GSP itself, we establish the following impossibility result: For more than three players, cost-sharing mechanisms that are both GSP and 1-BB do not exist in general, even if the cost function is symmetric. On the other hand, for at most three players and symmetric costs, there is always a GSP and 1-BB mechanism.

1.4.2 Cost Sharing Without Indifferences: To Be or Not to Be (Served)

Our second technique is based on a modified assumption on players' behavior: Specifically, it seems plausible that players prefer being served over not being served, also when they have to pay a price equal to their valuation. In contrast, the "standard" model assumes that players are indifferent between these two outcomes. We call mechanisms that are collusion-resistant under our modified assumptions as "group-strategyproof against service-aware players" (SGSP). It turns out that SGSP greatly increases the flexibility for designing collusion-resistant cost-sharing mechanisms. In particular, trading in GSP for SGSP allows for improved budget balance *and* improved economic efficiency.

In detail: We give a novel family of SGSP mechanisms that we call *egalitarian* due to being reminiscent to the algorithm for computing Dutta and Ray's egalitarian solutions. Our new mechanisms achieve 1-BB for arbitrary costs and additionally $O(\log n)$-EFF for the natural and large class of subadditive costs. Egalitarian mechanisms also fit into the framework of acyclic mechanisms. Thus far, acyclic mechanisms were only known to be WGSP; yet, we prove that they satisfy also the strictly stronger SGSP.

Our results show that for many cost functions defined by natural optimization problems, there are SGSP (and thus WGSP) egalitarian cost-sharing mechanisms that guarantee 1-BB and an economic efficiency known to be asymptotically optimal for truthful and (approximately) budget-balanced cost-sharing mechanisms (see Section 1.5.2). Unfortunately, many of these egalitarian mechanisms are not computable in polynomial time, unless $P = NP$. We therefore devise a framework for coping with the computational complexity, and we identify suitable approximation algorithms. Using this framework, we develop polynomial-time-computable egalitarian mechanisms for sharing the cost induced by various scheduling problems. We show that these mechanisms achieve provably better budget balance *and* economic efficiency than any Moulin mechanism. A comparison of Moulin, symmetric, and egalitarian mechanisms is shown in Table 1.1.

1.4.3 Does Coalition Size Matter?

GSP implies that players have virtually unlimited means to communicate and make binding agreements with all of their competitors. Since GSP is known to impose severe limitations on the other goals in cost sharing, there is hence good reason to seek for a weaker axiom: We study the following question: Does relaxing GSP to resistance only against *coalitions of bounded size* yield a richer set of possible mechanisms? We show that, surprisingly, the answer is essentially "no".

In detail, we prove that if a mechanism has a cost-sharing method and is group-strategyproof against coalitions of size only two ("2-GSP"), then it is also GSP. In other words, 2-GSP and GSP are equivalent if we require that cost shares must only depend on the set of served players, and not directly on the bids. Moreover, we show that even without additional requirements, 2-GSP implies WGSP. Consequently, our results give some justification that GSP may, after all, still be desirable in various scenarios. As another benefit, we believe that our characterizations will facilitate devising and understanding new GSP cost-sharing mechanisms. Finally, we also relate our findings to other concepts of non-manipulability known in the literature.

1.4.4 Generalizing the Model

Up to this point, we have assumed a *binary-demand* cost-sharing model, where players are either served or not served. However, this binary-demand model is not appropriate in every context. This is particularly the case when *fault tolerance* is an issue: For instance, when the service is connectivity to a network, players might have an increased

Table 1.1: Comparison of techniques for designing polynomial-time computable cost-sharing mechanisms, using the example of minimum-makespan problems

Makespan Problem / Technique / Collusion Resistance	BB	EFF	References and Remarks
arbitrary jobs on related machines ($Q\|\|C_{\max}$)			
Moulin mechanisms			
GSP	$2d$	$2d \cdot (1 + H_n)$	[5, 6]
Symmetric mechanisms			
GSP	$\frac{\sqrt{17}+1}{4} \cdot d$	$\Omega(n)$	Theorem 3.4.8, Lemma 3.4.3
Egalitarian mechanisms			
SGSP	2	$4H_n$	Theorem 4.6.6
arbitrary jobs on identical machines ($P\|\|C_{\max}$)			
Moulin mechanisms			
GSP	$\frac{2m}{m+1}$	$\Omega(n)$	[5]
GSP	$\frac{2m-1}{m}$	$\frac{2m-1}{m} \cdot (1 + H_n)$	[10]
Egalitarian mechanisms			
SGSP	$1 + \varepsilon$	$2(1 + \varepsilon) \cdot H_n$	Theorem 4.6.8, running time exponential in $\frac{1}{\varepsilon}$
SGSP	$\frac{4}{3} - \frac{1}{3m}$	$2(\frac{4}{3} - \frac{1}{3m}) \cdot H_n$	Theorem 4.6.4, practical mechanism
identical jobs on related machines ($Q\|p_i = p\|C_{\max}$)			
Egalitarian mechanisms			
SGSP	1	$2H_n$	Section 4.6.4

Note: d denotes the number of different processing requirements, H_n denotes the n-th harmonic number.

utility by having redundant connections—which corresponds to a higher reliability and a higher *quality of service*. The general-demand model accounts for this by allowing players to receive multiple levels of service. Put differently, a general-demand mechanism must choose a *multiset* of served players. We generalize the binary-demand Moulin mechanisms to the first general technique for designing GSP general-demand mechanism.

1.5 Other Related Work

1.5.1 Applications

Besides general design techniques as introduced in Section 1.3, most other work on cost sharing has focused on devising "good" cross-monotonic cost-sharing methods and, more recently, "good" acyclic mechanisms. In these works, costs stem from solutions of combinatorial optimization problems, including the minimum spanning tree [41, 37, 38], Steiner tree [37, 64, 65, 50], fixed tree multicast [23, 24, 3], facility location [59, 46, 65, 36, 50], rent-or-buy-network design [59, 65, 32], Steiner forest [13, 31, 42], edge/vertex/set cover [36, 50], and minimum makespan or minimum sum of completion times when scheduling parallel machines [5, 10, 9, 6, 8, 11]. In this list, three results [50, 8, 11] fit into the framework of acyclic mechanisms, all other results develop Moulin mechanisms with cross-monotonic cost shares.

Sharing Makespan Costs In this thesis, the predominant sample application is the minimum makespan problem, for which several results are known. For sharing the makespan cost of either n identical jobs on m related machines or n arbitrary jobs on m identical machines, $2m/(m+1)$-BB cross-monotonic cost-sharing methods are due to Bleischwitz and Monien [5]. It is shown in the same paper that this is generally the best that can be guaranteed under the constraint of cross-monotonicity. Brenner and Schäfer [10] later modified the cost-sharing methods for identical machines so that economic efficiency is improved from $\Omega(n)$-EFF to $O(\log n)$-EFF. For arbitrary processing requirements and related machines, the best known cross-monotonic cost-sharing methods achieve $2d$-BB, where d is the number of different processing requirements [5]. This is tight up to a factor of 2, since d is a lower bound [5].

1.5.2 Characterizing Collusion-Resistant Cost Sharing

Submodular Costs and Budget Balance A complete characterization of the impact of submodular costs on GSP and 1-BB was given by Moulin [51]: Any GSP mechanism that is 1-BB with regard to submodular costs is a Moulin mechanism, or at least always produces the same utilities as a Moulin mechanism. Conversely, for any submodular cost function, a rich class of cross-monotonic 1-BB cost-sharing methods (and thus Moulin mechanisms) always exists—including the marginal costs, the Shapley value [74], and the egalitarian solution [21]. Interestingly, of all cross-monotonic cost-sharing methods, the Shapley value is characterized by inducing the Moulin mechanism with the best possible economic efficiency guarantee [52, 64]. On the other hand, the egalitarian solution is characterized by maximizing the probability that a given subset S of players can afford the cost shares $\xi(S)$; this result is obtained when assuming that the players' valuations are independent random variables with a common distribution function (under mild restrictions) [54].

Subadditive Costs and Economic Efficiency Dobzinski et al. [19] studied the scenario of a so-called *excludable public good* [51, 17] where $C(S) = 1$ if $S \neq \emptyset$ and $C(\emptyset) = 0$. They showed for any such cost-sharing problem with n players that no SP and β-BB ($\beta \geq 1$) cost-sharing mechanism can guarantee social cost better than $\Omega(\log n)$ times the optimal social cost. The excludable-public-good case is a special instance of many natural cost-sharing problems with subadditive optimal costs. This includes, e.g., makespan, facility location, and rooted Steiner tree problems. Consequently, $\Omega(\log n)$-EFF is a lower bound for all these cost-sharing problems.

Superadditive Costs and Singleton Mechanisms Brenner and Schäfer [11] studied several scheduling cost-sharing problems where the cost of a schedule is defined as the sum of all players' completion times. All problems they considered have superadditive optimal costs (meaning that the union of disjoint sets is always more costly than sum of the stand-alone costs), and the excludable-public-good case case therefore cannot occur. In fact, Brenner and Schäfer [11] gave a simple subclass of acyclic mechanisms, called *singleton mechanisms*, that guarantee 1-BB and *constant*-factor approximations of the social cost (independent of the number of players), i.e., $O(1)$-EFF.

Intuitively, singleton mechanisms can be specified by a complete binary tree with n levels: Every node is labeled with a player, and every path from the source to a leaf contains each player exactly once. Now applying the mechanism corresponds to finding a path from the root to a leaf: Start at the root and initialize Q as the empty set. Now proceed as follows: If the player at the current node can afford his marginal cost $C(Q \cup \{i\}) - C(Q)$, charge him this price, add him to Q, and go to the right successor node. Otherwise, go to the left successor.

Indifference Rules and Cross-Monotonicity Both Moulin and acyclic mechanisms treat indifferent players in an extreme way, in that they are always served. This property is referred to as *upper continuity*. For GSP mechanisms, an interesting characterization of upper continuity is due to Immorlica et al. [36]: If a GSP mechanism is upper-continuous then it has cross-monotonic cost shares. Consequently, it is a Moulin mechanism.

Non-Manipulability Since in many mechanism-design scenarios it is unlikely that players have unlimited means to communicate and make binding agreements with all of their competitors, Serizawa [70] introduced and advocated relaxing GSP to *effective pairwise strategyproofness*. This property is a little weaker than our "2-GSP" because it means that a mechanism only needs to be resistant to pairs of defecting players if their defection was stable (i.e., none of the two players could betray his partner to further increase his utility). However, the models considered by Serizawa do not include the cost-sharing scenario.

Besides the (coalitional) variants of strategyproofness, there are several other concepts of non-manipulability. Satterthwaite and Sonnenschein [66] suggested a property called *(outcome) non-bossiness* (ONB): If a single player changes his bid in a way so that his own outcome does not change, then all other players should also get the same outcome as before (hence, no player can "boss" others around). In an unpublished paper, Shenker [72] proved several results on the relationship between various forms of truthfulness, non-bossiness, and other technical properties. His results are similar but different to our work. With focus only on the cost-sharing model, several other relationships were later studied by Mutuswami [55]. He introduced a variation of ONB called *weak utility non-bossiness* (WUNB), meaning that if a single player changes his bid so that his utility remains the same, then no other player may become better off. Mutuswami [55] showed that SP and ONB together imply WGSP; moreover, SP, ONB, and WUNB together imply GSP. Other variants of non-bossiness were also proposed by Deb and Razzolini [17]. For scenarios when players are capable of side-payments, Schummer [69] studied *bribe-proof* mechanisms, meaning that no player has an incentive to bribe another player into submitting an untruthful bid. For the cost-sharing scenario, Schummer's results imply that collusion-resistance properties that include monetary transfers are too strong (see also Section 5.2.1): They would rule out all but trivial mechanisms where each player's utility is completely independent of the other players' actions.

General-Demand Cost Sharing To the best of our knowledge, general-demand cost sharing has previously only been considered by Moulin [51], Devanur et al. [18], and Mehta et al. [50]. However, these works consider only SP and WGSP mechanisms, respectively.

1.5.3 Outside the Realm of Cost Sharing

The Minimum-Makespan Problem The discipline algorithmic mechanism design was pioneered by Nisan and Ronen [56] roughly a decade ago. One of the optimization problems the authors considered—based on the new idea that part of the input data is held by selfish players—was minimizing the makespan when scheduling unrelated parallel machines: In Nisan and Ronen's work, the machines (and not the jobs as in our cost-sharing applications) are controlled by selfish players and thus have to be given monetary incentives to truthfully reveal the true time needed to process each job. Until today, there is a huge gap between the best known lower and upper bounds on the approximation ratio that "incentive-compatible" algorithms can achieve for this problem. In fact, it is one of the central open problems in algorithmic mechanism design whether "incentive-compatible" approximation is necessarily less powerful than "classical approximation". For a recent survey and further pointers into the literature we refer to Roughgarden [63].

Small Coalitions and Solution Concepts The idea of resistance to coalitions of only bounded size, as described in Section 1.4.3, has been considered also from the perspective of game-theoretic solution concepts: Andelman et al. [2] defined a k-strong equilibrium so that it corresponds in our cost-sharing model to WGSP against coalitions of size at most k. In the paper, the authors obtain results on the *strong price of anarchy* (i.e., the worst-case loss in the system due to selfish behavior of the players) in job scheduling and network creation games, depending on the maximum coalition size k.

1.6 Organization

Chapter 2 gives all technical preliminaries. We formally define cost-sharing problems, the various notions of truthfulness, and all other desirable properties of cost-sharing mechanisms. Afterwards, we give brief definitions of both VCG and Moulin mechanisms and of the combinatorial optimization problems considered in this thesis.

Our main results are given in the remaining chapters of this thesis: In Chapter 3, we introduce our new technique for designing GSP mechanisms, based on the idea of lexicographic maximization of players' utilities. In Chapter 4, we study cost-sharing models without indifferences and develop our new egalitarian mechanisms. Afterwards, we show in Chapter 5 that relaxing GSP to resistance only against coalitions of bounded size does not allow for better mechanisms. Finally, we give generalized Moulin mechanisms for the general-demand cost-sharing model in Chapter 6.

While Section 2.1 is a prerequisite for all what follows, readers with acquaintance of mechanism design and combinatorial optimization may quickly flip through the remaining two sections of Chapter 2. All further chapters may be read independently from each other.

1.7 Prerequisites

Basic knowledge of game-theoretic concepts will help in comprehending explanations and interpretations; however, all formal definitions necessary for our results are contained in this thesis. For a general introduction to game theory from a (micro-)economic point of view, we recommend Mas-Colell et al. [47]. Introductions to various aspects of game theory with a bias towards computer science can be found in the book by Nisan et al. [57]. The essentials of mechanism design are presented in a concise and accessible manner by Parkes [61].

While we likewise define all combinatorial optimization problems that we use in our applications, we assume at least rudimentary acquaintance with the theory of computational intractability and approximation algorithms. A standard reference for NP-completeness is Garey and Johnson [26]. As a reference for approximation algorithms we recommend the book by Hochbaum [33] as it discusses many of the optimization problems studied in the application sections of this thesis.

1.8 Bibliographic Notes

Many of the results presented in this thesis are based on collaborative work with others, and most have appeared in preliminary form in research papers. For this thesis, all joint work has been revised, and all text and all proofs have been written only by me. This thesis includes only results to which I contributed.

Chapter 3 is based on joint work with Yvonne Bleischwitz, Burkhard Monien, and Karsten Tiemann. It has been published in the *Proceedings of the 32nd International Symposium on Mathematical Foundations of Computer Science (MFCS'07)* [9]. Chapter 4 is based on joint work with Yvonne Bleischwitz and Burkhard Monien. It has been published in the *Proceedings of the 3rd International Workshop on Internet and Network Economics (WINE'07)* [8]. Chapter 5 has appeared in the *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE'08)* [67]. A small part of Chapter 3 and all of Chapter 6 are joint work with Yvonne Bleischwitz and have been published in *Information Processing Letters 107(2), 2008* [6] and in the *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)* [7].

Since the focus of this thesis is on cost-sharing mechanisms, I did not include results in other research areas that I also published during my PhD project. These publications deal with the loss due to selfish behavior (the *price of anarchy*) in selfish routing and competitive location scenarios. They appeared, as joint work, in the *Proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS'06)* [1], the *Proceedings of the 2nd International Workshop on Internet and Network Economics (WINE'06)* [49], the *Proceedings of the 3rd International Workshop on Internet and Network Economics (WINE'07)* [25], and the *Proceedings of the 33rd International Symposium on Mathematical Foundations of Computer Science (MFCS'08)* [48].

Chapter 2

Preliminaries

2.1 The Formal Model

2.1.1 Notation

Sets The set of positive integers $\{1, 2, 3, \dots\}$ is denoted by $\mathbb{N}$, the set of non-negative integers by $\mathbb{N}_0$. For $n, m \in \mathbb{Z}$, we write $\{n \dots m\} := \{n, n+1, \dots, m\}$, which is the empty set if $m < n$. Moreover, $[n] := \{1 \dots n\}$. Given a finite set $S \subset \mathbb{N}$ and $i \in \mathbb{N}$, we let $\operatorname{rank}(i, S) := |\{j \in S \mid j \leq i\}|$ be the *rank* of i in S. Moreover, $\mathit{MIN}_k\, S := \{i \in S \mid \operatorname{rank}(i, S) \leq k\}$ is defined as the set of the k smallest elements in S, and $\mathit{MAX}_k\, S$ is likewise defined as the set of the k largest elements in S.

Vectors Given an arbitrary set X and a non-empty index set $I \subset \mathbb{N}$, a *vector* $\boldsymbol{x} \in X^I$ is a family of elements of X indexed by I. It is denoted $\boldsymbol{x} = (x_i)_{i \in I}$. Given a subset $K \subseteq I$, we define the vector $\boldsymbol{x}_K := (x_i)_{i \in K}$ as the subfamily of $\boldsymbol{x}$ indexed by K. Similarly, $\boldsymbol{x}_{-K} := (x_i)_{i \in I \setminus K}$. Given two vectors $\boldsymbol{x} \in X^I$ and $\boldsymbol{z} \in X^K$, we use the usual game-theoretic notation $(\boldsymbol{x}_{-K}, \boldsymbol{z})$ to denote the vector $\boldsymbol{y} \in X^I$ with $y_i = z_i$ for $i \in K$ and $z_i = x_i$ for $i \in I \setminus K$. We say two vectors $\boldsymbol{x}, \boldsymbol{y} \in X^I$ are *K-variants* if $\boldsymbol{x}_{-K} = \boldsymbol{y}_{-K}$.

To simplify notation, we will often omit curly brackets around singleton sets when it is unambiguous to do so. E.g., we will write i-variants instead of $\{i\}$-variants or $K \cup i$ instead of $K \cup \{i\}$. Moreover, we identify X^n with $X^{[n]}$ and apply the previous notation also to vector-valued functions. E.g., if $f : A \to X^n$ is a function, $a \in A$, $f(a) = \boldsymbol{x}$, and $I \subseteq [n]$, then $f_I(a) = \boldsymbol{x}_I$.

Binary Relations If $x_i \leq y_i$ for all $i \in I$, we write $\boldsymbol{x} \leq \boldsymbol{y}$. If $\boldsymbol{x} \leq \boldsymbol{y}$ and $\boldsymbol{x} \neq \boldsymbol{y}$, we write $\boldsymbol{x} < \boldsymbol{y}$. Moreover, if $x_i < y_i$ for all $i \in I$, we write $\boldsymbol{x} \ll \boldsymbol{y}$. If $\boldsymbol{x}$ is lexicographically no larger than $\boldsymbol{y}$, we write $\boldsymbol{x} \preceq \boldsymbol{y}$. Correspondingly, if $\boldsymbol{x} \preceq \boldsymbol{y}$ but $\boldsymbol{x} \neq \boldsymbol{y}$, we write $\boldsymbol{x} \prec \boldsymbol{y}$. The maximum element of $\preceq$ on X is denoted $\operatorname{lex\,max} X$.

Others We denote the n-th *harmonic number* by $H_n := \sum_{i=1}^{n} \frac{1}{i}$.

2.1.2 Cost-Sharing Problems and Mechanisms

A (binary-demand) *cost-sharing problem* with $n \in \mathbb{N}$ players is specified by a *cost function* $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$ that associates all possible sets of served players to the incurred service cost. A set of served players $Q^* \subseteq [n]$ together with a cost distribution $\boldsymbol{x}^* \in \mathbb{R}^n$ is called an *outcome*. We denote player i's true *valuation* for being served by $v_i \in \mathbb{R}$. Unless otherwise stated, we assume *quasi-linear utilities*, i.e., player i's utility for outcome $(Q^*, \boldsymbol{x}^*)$ is $v_i \cdot q_i^* - x_i$ where $q_i^* \in \{0,1\}$, $q_i^* = 1 :\Longleftrightarrow i \in Q^*$.

Definition 2.1.1. *A* cost-sharing mechanism $M = (Q, x)$ *consists of a pair of functions* $Q : \mathbb{R}^n \to 2^{[n]}$ *and* $x : \mathbb{R}^n \to \mathbb{R}^n$ *that associate any bid vector* $\boldsymbol{b}$ *to an outcome* $(Q(\boldsymbol{b}), x(\boldsymbol{b}))$.

Sometimes the set notation will be inconvenient, and we therefore implicitly define $q : \mathbb{R}^n \to \{0,1\}^n$ by $q_i(\boldsymbol{b}) = 1 :\Longleftrightarrow i \in Q(\boldsymbol{b})$. Given a cost-sharing mechanism $M = (Q, x)$, we write $M_i(\boldsymbol{b}) := (q_i(\boldsymbol{b}), x_i(\boldsymbol{b}))$ and define $u_i(\boldsymbol{b} \mid v_i) := v_i \cdot q_i(\boldsymbol{b}) - x_i(\boldsymbol{b})$. When utilities are quasi-linear, $u_i(\boldsymbol{b} \mid v_i)$ is hence player i's utility for outcome $(Q(\boldsymbol{b}), x(\boldsymbol{b}))$. Provided that there is no confusion about the true valuation v_i, we simply write $u_i(\boldsymbol{b})$ instead of $u_i(\boldsymbol{b} \mid v_i)$.[1] We let $M(\boldsymbol{b}) := (M_1(\boldsymbol{b}), \dots, M_n(\boldsymbol{b}))$ and $u(\boldsymbol{b}) := (u_1(\boldsymbol{b}), \dots, u_n(\boldsymbol{b}))$.

Unless otherwise noted, we will always require three standard axiomatic properties in this thesis:

- *No positive transfers* (NPT): Players never get paid, i.e., $x_i(\boldsymbol{b}) \geq 0$.
- *Voluntary participation* (VP): When served, players never pay more than they bid; otherwise, they are charged nothing, i.e., if $i \in Q(\boldsymbol{b})$ then $x_i(\boldsymbol{b}) \leq b_i$, else $x_i(\boldsymbol{b}) = 0$.
- *Consumer sovereignty* (CS): Each player can bid in a way so that he is served, regardless of the other players' bids; i.e., there is a $b^\infty \in \mathbb{R}_{\geq 0}$ such that if $b_i \geq b^\infty$ then $i \in Q(\boldsymbol{b})$.

VP and NPT imply that players may opt not to participate. Technically, these players submit a negative bid. This property in conjunction with CS is sometimes referred to as *strong CS*. It strengthens the collusion-resistance requirements and rules out otherwise implausible and undesirable mechanisms [36].

2.1.3 Non-Manipulability

The basic notion of truthfulness is *strategyproofness* (SP). It requires a mechanism M to guarantee that for all possible valuation vectors $\boldsymbol{v} \in \mathbb{R}^n$, all players $i \in [n]$, and all i-variants $\boldsymbol{b}$ of $\boldsymbol{v}$ it holds that $u_i(\boldsymbol{b}) \leq u_i(\boldsymbol{v})$. In this thesis, as well as in many related works on cost sharing, a stronger notion is required that also ensures resistance against coordinated manipulation.

Definition 2.1.2. *A cost-sharing mechanism M is* group-strategyproof *(GSP) if for all true valuations* $\boldsymbol{v} \in \mathbb{R}^n$ *and all non-empty coalitions* $K \subseteq [n]$ *there is no K-variant* $\boldsymbol{b}$ *of* $\boldsymbol{v}$ *with* $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$.

[1] Since the true valuation v_i is an "optional" argument, we separate it by "|" for better readability.

We say a non-empty coalition $K \subseteq [n]$ is *GSP-successful at* $\boldsymbol{v}$ (or simply a *successful coalition*) if there is some K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ so that the coalition improves by deviating, i.e., $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$. With the corresponding modifications, we will use this terminology also for other kinds of collusion resistance.

A *weaker* notion of collusion resistance is obtained by *strengthening* the requirements for successful coalitions:

Definition 2.1.3. *A cost-sharing mechanism M is* weakly group-strategyproof *(WGSP) if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all non-empty coalitions $K \subseteq [n]$ there is no K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \gg u_K(\boldsymbol{v})$.*

Besides the coalitional variants of strategyproofness, there are several other concepts of non-manipulability. In this thesis, we consider a property introduced by Satterthwaite and Sonnenschein [66]: If a single player changes his bid in a way so that his own outcome does not change, then all other players should also get the same outcome as before.

Definition 2.1.4 (Satterthwaite and Sonnenschein [66]). *A cost-sharing mechanism M is (outcome)* non-bossy *(ONB) if for all players $i \in [n]$ and all i-variants $\boldsymbol{b}, \boldsymbol{b}' \in \mathbb{R}^n$ it holds that $M_i(\boldsymbol{b}) \neq M_i(\boldsymbol{b}')$ or $M(\boldsymbol{b}) = M(\boldsymbol{b}')$.*

Another notion of non-bossiness was later introduced by Mutuswami [55].

Definition 2.1.5 (Mutuswami [55]). *A cost-sharing mechanism M is* weakly utility non-bossy *(WUNB) if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$, all players $i \in [n]$, and all i-variants $\boldsymbol{b}$ of $\boldsymbol{v}$ it holds that $u_i(\boldsymbol{b}) \neq u_i(\boldsymbol{v})$ or $u_{-i}(\boldsymbol{b}) \leq u_{-i}(\boldsymbol{v})$.*

2.1.4 Cost-Sharing Methods

A straightforward idea for devising cost-sharing mechanisms is to separate a mechanism into two parts: First, compute the set of served players depending on the bids. Then, compute the cost shares only depending on the set of served players. In fact, it is known that *all* GSP mechanisms *have to* work this way. This has been originally observed by Moulin [51]. Later on, we will give a more general statement in Theorem 4.2.9.

Definition 2.1.6. *A* cost-sharing method *is a function $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ that associates each set of players to a cost distribution, where for all $S \subseteq [n]$ and all $i \notin S$ it holds that $\xi_i(S) = 0$.*

Definition 2.1.7. *A cost-sharing mechanism $M = (Q, x)$ is* separable *if there exists a cost-sharing method ξ so that $x = \xi \circ Q$, i.e., for all $\boldsymbol{b} \in \mathbb{R}^n : x(\boldsymbol{b}) = \xi(Q(\boldsymbol{b}))$.*

Given a cost-sharing method ξ and a bid vector $\boldsymbol{b}$, we say a set of players $S \subseteq [n]$ is $\boldsymbol{b}$*-feasible* with regard to ξ if for all $i \in S$ it holds that $\xi_i(S) \leq b_i$. Trivially, the empty set is always $\boldsymbol{b}$-feasible.

2.1.5 Dealing with Indifferences

A general rule of thumb for the design of truthful mechanisms is that a player's payment must not depend directly on his own bid. In particular, the following simple proposition is well-known and a standard fact (see, e.g., Deb and Razzolini [16]).

Proposition 2.1.8 (Threshold Property). *A cost-sharing mechanism $M = (Q, x)$ is SP if and only if the following holds: For all $i \in [n]$ and all $\boldsymbol{b}_{-i} \in \mathbb{R}^{[n]\setminus i}$, there is a non-negative* threshold value *$\theta_i(\boldsymbol{b}_{-i})$ so that if $b_i > \theta_i(\boldsymbol{b}_{-i})$ then $i \in Q(\boldsymbol{b})$, if $b_i < \theta_i(\boldsymbol{b}_{-i})$ then $i \notin Q(\boldsymbol{b})$, and if $i \in Q(\boldsymbol{b})$ then $x_i(\boldsymbol{b}) = \theta_i(\boldsymbol{b}_{-i})$.*

We call a player i *indifferent at* $\boldsymbol{b}$ if $b_i = \theta_i(\boldsymbol{b}_{-i})$, i.e., player i bids exactly his threshold value. Clearly, the threshold property leaves open how to handle indifferent player. Two extreme options are to always serve players who bid their respective threshold value or to always reject them. Formally, these two extremes are captured by the notions upper- and lower-continuity.

Definition 2.1.9. *A cost-sharing mechanism $M = (Q, x)$ is called* upper-continuous *if for all players i and all bid vectors $\boldsymbol{b}$ the following holds: If $i \in Q(\boldsymbol{b}_{-i}, z)$ for all $z > b_i$ then also $i \in Q(\boldsymbol{b})$. Likewise, M is called* lower-continuous *if the following holds: If $i \notin Q(\boldsymbol{b}_{-i}, z)$ for all $z < b_i$ then also $i \notin Q(\boldsymbol{b})$.*

We remark that an alternative property called *upper semi-continuity* is sometimes defined in the literature (e.g., by Deb and Razzolini [16]), meaning that $q_i(\boldsymbol{b}_{-i}, \cdot)$ is always upper semi-continuous. That is, if $q_i(\boldsymbol{b}) = 0$ then $\exists \varepsilon > 0 : \forall z \in (b_i - \varepsilon, b_i + \varepsilon) : q(\boldsymbol{b}_{-i}, z) = 0$. For mechanisms that fulfill the threshold property, upper semi-continuity and upper continuity are clearly equivalent.

2.1.6 Budget Balance and Economic Efficiency

In typical applications, cost functions are implicitly defined by combinatorial optimization problems, i.e., $C(S)$ is the value of a minimum-cost solution for the problem instance that corresponds to the set of served players S. Due to the NP-hardness of many natural problems, usually only approximations with cost $C'(S) \geq C(S)$ can be computed in polynomial time, unless P = NP. Still, the budget of the mechanism should be reasonably balanced:

Definition 2.1.10. *A mechanism $M = (Q, x)$ is β*-budget-balanced (β-BB) with regard to actual costs C' and optimal costs C *if for all bid vectors $\boldsymbol{b}$ it holds that*

$$C'(Q(\boldsymbol{b})) \leq \sum_{i=1}^{n} x_i(\boldsymbol{b}) \leq \beta \cdot C(Q(\boldsymbol{b})),$$

where $\beta \geq 1$ is a constant (independent of $\boldsymbol{b}$).

We define β-BB in the corresponding way also for cost-sharing methods. Since the definition of budget balance is meaningless otherwise, we will always assume and only consider problems with $C(\emptyset) = 0$.

For economic efficiency, the service cost and the rejected players' valuations should be traded off as good as possible. A measure for this trade-off is the *social cost* function $SC : 2^{[n]} \to \mathbb{R}_{\geq 0}$. Given actual costs C' and true valuations $\boldsymbol{v}$, social costs are defined by $SC(S) := C'(S) + \sum_{i \notin S} \max\{v_i, 0\}$.

Definition 2.1.11. *A mechanism $M = (Q, x)$ is α*-efficient (α-EFF) with regard to actual costs C' and optimal costs C *if for all true valuations $\boldsymbol{v}$ it holds that*

$$SC(Q(\boldsymbol{v})) \leq \alpha \cdot \min_{P \subseteq [n]} \left\{ C(P) + \sum_{i \notin P} \max\{v_i, 0\} \right\} ,$$

where $\alpha \geq 1$ is a constant (independent of $\boldsymbol{v}$).

Note here that there are two potential sources for loss of economic efficiency: First, the selected set S may be suboptimal; and second, the actual cost $C'(S)$ may be too high.

In the sections where polynomial-time computability is not an issue, we implicitly assume that the actual costs C' coincide with the optimal costs C. When there is no confusion, we will usually only write β-BB and α-EFF (and omit the "with regard to").

2.1.7 Special Cost Functions

In many cases, costs exhibit a special structure that can be exploited when designing cost-sharing mechanisms. In this thesis, we discuss costs with the following properties:

- *Symmetric costs*: Costs depend only on the number of served players. That is, for any two sets $S, T \subseteq [n]$ with $|S| = |T| : C(S) = C(T)$.

- *Subadditive costs*: The cost of the union of two sets is never more than the sum of the stand-alone costs. That is, for any two sets $S, T \subseteq [n] : C(S \cup T) \leq C(S) + C(T)$.

- *Superadditive costs*: The cost of the union of two disjoint sets is never less than the sum of the stand-alone costs. That is, for any two sets $S, T \subseteq [n], S \cap T = \emptyset :$ $C(S \cup T) \geq C(S) + C(T)$.

- *Submodular costs*: The marginal costs of adding players to some set S are non-increasing in the size of S. That is, for all players $i \in [n]$ and any two sets $S \subseteq T \subseteq [n] : C(T \cup \{i\}) - C(T) \leq C(S \cup \{i\}) - C(S)$. It can be shown that this condition is equivalent to that for all $S, T \subseteq [n] : C(S \cup T) + C(S \cap T) \leq C(S) + C(T)$.

- *Supermodular costs*: Marginal costs are non-decreasing, i.e., for any two sets $S \subseteq T : C(T \cup \{i\}) - C(T) \geq C(S \cup \{i\}) - C(S)$. Equivalently, for all $S, T \subseteq [n]$: $C(S \cup T) + C(S \cap T) \geq C(S) + C(T)$.

Note that subadditivity seems very natural in cost-sharing scenarios as it conveys the idea of synergies between players. On the other hand, superadditivity may be seen as the result of congestion. Clearly, sub- and supermodularity are special cases of sub- and superadditivity, respectively.

2.2 Previous Design Techniques

For completeness, we give formal definitions of Vickrey-Clarke-Groves (VCG) mechanisms [75, 14, 30] and Moulin mechanisms [51, 52] in this section. The acyclic-mechanism framework by Mehta et al. [50] will be needed only in Chapter 4. We therefore postpone its formal introduction to Section 4.4, where we give a slight generalization of the original algorithm for computing the outcome of these mechanisms.

2.2.1 Vickrey-Clarke-Groves Mechanisms

Recall that VCG mechanisms are the unique class of 1-EFF mechanisms [29]. They are, of course, a general technique in mechanism design and in no way restricted to cost-sharing problems. We give a definition using our cost-sharing notation here.

Definition 2.2.1. *A cost-sharing mechanism $M = (Q, x)$ is a* VCG mechanism *with respect to non-decreasing costs C if for all bid vectors $\boldsymbol{b}$ and players $i \in [n]$ it holds that*

$$\begin{aligned} Q(\boldsymbol{b}) &\in \arg\max_{T \subseteq [n]} \left\{ \sum_{j \in T} b_j - C(T) \right\} \\ x_i(\boldsymbol{b}) &= C(Q(\boldsymbol{b})) - \sum_{j \in Q(\boldsymbol{b}) \setminus \{i\}} b_j + h_i(\boldsymbol{b}_{-i}), \end{aligned}$$

where $h_i : \mathbb{R}^{[n] \setminus i} \to \mathbb{R}$ is a function independent of b_i.

By definition, if all players bid truthfully, a VCG mechanism selects a set of players with maximum social welfare (sum of the included players' valuations minus service cost). Equivalently, it picks a set with minimum social cost.

We give a short explanation why VCG mechanisms are SP: Consider an arbitrary player $i \in [n]$. Note that his cost share $x_i(\boldsymbol{b})$ and thus also his utility do not directly depend on his own bid b_i, but only on the set of served players $Q(\boldsymbol{b})$ and the other players' bids $\boldsymbol{b}_{-i}$. Moreover, the function h_i is irrelevant for incentive considerations, so we might as well assume $h_i \equiv 0$. Let now $\boldsymbol{v}$ contain the true valuations. Player i's cost share $x_i(\boldsymbol{v})$ is defined in a way such that his utility $u_i(\boldsymbol{v}) = v_i \cdot q_i(\boldsymbol{v}) + x_i(\boldsymbol{v})$ is exactly equal to the maximum social welfare. Hence, it holds for any i-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ that

$$u_i(\boldsymbol{b}) = v_i \cdot q_i(\boldsymbol{b}) + \sum_{j \in Q(\boldsymbol{b}) \setminus \{i\}} b_j - C(Q(\boldsymbol{b})) = \sum_{j \in Q(\boldsymbol{b})} v_j - C(Q(\boldsymbol{b})) \leq u_i(\boldsymbol{v}),$$

where the last inequality holds because, by Definition 2.2.1, the set of players $Q(\boldsymbol{v})$ that the mechanisms chooses for input $\boldsymbol{v}$ has optimal social welfare. Consequently:

Proposition 2.2.2. *VCG mechanisms are SP and 1-EFF.*

The functions h_i can be chosen so that the respective VCG mechanism satisfies NPT, VP, and CS [52]. Unfortunately, however, VCG mechanisms do not provide any guarantees for cost recovery: Consider the following simple example in the context of cost sharing: Let there be two players, and consider the excludable-public-good case, i.e., $C(S) = 1$ if $S \neq \emptyset$ and $C(\emptyset) = 0$. If $b_1 = 0$ and $b_2 = 1$ it holds that $x_1(\boldsymbol{b}) = 0$ due to NPT and VP, and thus $h_1(b_2) = 0$. Similarly, $h_2(b_1) = 0$ if $b_1 = 1$. Consequently, if $b_1 = b_2 = 1$ we have $Q(\boldsymbol{b}) = \{1, 2\}$ but no cost is recovered.

Besides not being budget-balanced, one can also find simple examples where VCG mechanisms are not resistant against collusion (see, e.g., Moulin and Shenker [52]).

2.2.2 Moulin Mechanisms

In the following, we formally introduce Moulin mechanisms, which are the most universal technique for the design of GSP cost-sharing mechanisms. Their main characteristic is having cross-monotonic cost shares.

Definition 2.2.3. *A cost-sharing method ξ is* cross-monotonic *if for all sets A, B with $A \subseteq B$ and all players $i \in A$ it holds that $\xi_i(A) \geq \xi_i(B)$.*

Given cross-monotonic cost shares, Moulin mechanisms are characterized by serving the largest feasible set of players. We remark here that the original definition by Moulin [51] and by Moulin and Shenker [52] was different and essentially algorithmic[2] (see Algorithm 2.1). Yet we will see very soon that both definitions are equivalent.

Definition 2.2.4. *A cost-sharing mechanism $M = (Q, x)$ is called a* Moulin mechanism *if it has a cross-monotonic cost-sharing method ξ and for every bid vector $\boldsymbol{b}$ it holds that $Q(\boldsymbol{b})$ is the largest $\boldsymbol{b}$-feasible set with regard to ξ.*

Recall here that there is a unique largest $\boldsymbol{b}$-feasible set because ξ is cross-monotonic: Suppose both A and B were largest $\boldsymbol{b}$-feasible sets, then also $A \cup B$ is $\boldsymbol{b}$-feasible due to cross-monotonicity. By assumption, it must then hold that $A = B$. Consequently, Moulin mechanisms are well-defined. Equivalently to Definition 2.2.4, a mechanism $M = (Q, x)$ is a Moulin mechanism if and only if there is a cost-sharing method ξ so that for all $\boldsymbol{b} \in \mathbb{R}^n$ it holds that $Q(\boldsymbol{b}) = \max\{S \subseteq [n] \mid \forall i \in S : b_i \geq \xi_i(S)\}$ and $x(\boldsymbol{b}) = \xi(Q(\boldsymbol{b}))$. It turns out that this non-constructive definition allows for a much simpler proof of GSP than the one given by Moulin and Shenker [52]:

Theorem 2.2.5. *Moulin mechanisms are GSP.*

Proof. Let $\boldsymbol{v}$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$. By assumption and due to VP, it then holds that $Q(\boldsymbol{b})$

[2] To be exact, Moulin and Shenker defined $Q(\boldsymbol{b})$ as the limit of the decreasing set sequence $Q_0 := [n]$, $Q_{j+1} := \{i \in Q_j \mid b_i \geq \xi_i(Q_j)\}$.

is $\boldsymbol{v}$-feasible. Consequently, due to cross-monotonicity, also $Q(\boldsymbol{v}) \cup Q(\boldsymbol{b})$ is $\boldsymbol{v}$-feasible. Since $Q(\boldsymbol{v})$ is the largest $\boldsymbol{v}$-feasible set by definition of Moulin mechanisms, this implies $Q(\boldsymbol{b}) \subseteq Q(\boldsymbol{v})$. Therefore, again by cross-monotonicity, $u_i(\boldsymbol{b}) \le u_i(\boldsymbol{v})$ for all $i \in Q(\boldsymbol{v})$ and thus $u_K(\boldsymbol{b}) = u_K(\boldsymbol{v})$. □

The outcome of Moulin mechanisms can be efficiently computed in a straightforward manner:

```
Input:   cross-monotonic cost-sharing method $\xi$, bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output:  set of players $Q \in 2^{[n]}$, cost distribution $\boldsymbol{x} \in \mathbb{R}^n_{\ge 0}$
1: $Q := [n]$
2: while $\exists i \in Q$: $b_i < \xi_i(Q)$ do
3:     $Q := \{i \in Q \mid b_i \ge \xi_i(Q)\}$
4: $\boldsymbol{x} := \xi(Q)$
```

Algorithm 2.1: Moulin mechanisms

Lemma 2.2.6. *Let ξ be a cross-monotonic cost-sharing method. Then, for all bid vectors $\boldsymbol{b}$, Algorithm 2.1 computes the outcome of the respective Moulin mechanism.*

Proof. Denote by S the set chosen by the Moulin mechanism, and by $(Q^*, \boldsymbol{x}^*)$ the outcome returned by Algorithm 2.1.

We show that no player dropped by Algorithm 2.1 can be contained in S. In detail, we verify by induction that the invariant $S \subseteq Q$ holds throughout the algorithm. Clearly, this holds before the first iteration of the while-loop. Therefore, consider an arbitrary iteration and fix all variable values immediately before line 3. Suppose $S \subseteq Q$ holds. Now assume, by way of contradiction, that there is a player $i \in S$ who will be dropped in the current iteration, i.e., $b_i < \xi_i(Q)$. Then $b_i < \xi_i(Q) \le \xi_i(S)$, where the last inequality holds due to cross-monotonicity. This is a contradiction to the fact that S is $\boldsymbol{b}$-feasible. Hence, any player removed in the current iteration cannot be contained in S, and the invariant continues to hold also after line 3.

Now, since the exit condition of the while-loop was fulfilled in the last iteration, Q^* is clearly $\boldsymbol{b}$-feasible. Moreover, $S \subseteq Q^*$, and S is the largest $\boldsymbol{b}$-feasible set. Consequently, $S = Q^*$. □

2.3 Optimization Problems

2.3.1 Scheduling Problems

Problem Definition Machine scheduling is one of the fundamental areas in combinatorial optimization, with a huge number of specific problem types (see, e.g., Brucker [12]). We therefore define in detail only the problem of minimizing the maximum completion time when scheduling n jobs on m *parallel related* machines. This problem is the predominant application considered in this thesis.

Each job i is characterized by its *processing requirement* $p_i \in \mathbb{N}$. Similarly, each machine j is characterized by its *speed* $s_j \in \mathbb{N}$, which is the amount of processing that machine j can finish within one unit of time. A schedule maps, in a non-overlapping way, each job i to some machine j and a starting time t. The *completion time* of job i is then $C_i := t + p_i/s_j$. The objective is to minimize the maximum completion time of any job, i.e., to minimize $C_{\max} := \max_{i\in[n]} C_i$. A different term for $C_{\max}$ is the *makespan*. In makespan problems, the precise schedule is often not needed and we consider only an allocation $\boldsymbol{a} \in [m]^n$ of the jobs to machines. Then, assuming that jobs are processed without gaps, $C_{\max} = \max_{j\in[m]} \sum_{i\in[n]|a_i=j} p_i/s_j$. Special cases we also consider are *identical jobs* (all p_i are equal) and *identical machines* (all s_j are equal).

In the context of cost sharing, each player corresponds to a job. The (optimal) service cost $C(S)$ is defined as the optimal value of the objective function for scheduling only the jobs in set S. For instance, if the objective is to minimize the makespan, then

$$C(S) := \min_{\boldsymbol{a}\in[m]^S} \left\{ \max_{j\in[m]} \frac{\sum_{i\in S|a_i=j} p_i}{s_j} \right\} .$$

We remark that the social cost $SC(S) = C'(S) + \sum_{i\notin S} \max\{v_i, 0\}$ is the objective function of the respective scheduling problem *with rejection* where $\max\{v_i, 0\}$ is the external cost (typically called "penalty" in the optimization literature) for not serving player $i \in [n]$.

The Three-Field Classification Scheme In order to distinguish between the various variants of scheduling problems, we make use of the three-field notation $\alpha|\beta|\gamma$ introduced by Graham et al. [28]: The field α represents the machine environment: E.g., "1" denotes a single machine, "P" identical parallel machines, "Q" related parallel machines, and an optional $m \in \mathbb{N}_{\geq 2}$ after P or Q denotes that the number of machines is m and thus a constant. The field β defines job characteristics: $p_i = p$ means that all processing requirements are identical, r_i means that the jobs may have release times (earliest starting time), and "pmtn" indicates that preemption is allowed. Finally, the field γ refers to the objective function. E.g, $C_{\max}$ is the makespan, $\sum C_i$ is the sum of completion times, and $\sum w_i C_i$ is the weighted sum of completion times.

Algorithms Even $P2||C_{\max}$ is an NP-hard problem [45]. A simple approximation algorithm for $Q||C_{\max}$ is the *longest processing time first* (LPT) algorithm [27], which assigns jobs in the order of decreasing processing requirements. Every job is allocated to the machine where its completion time is minimal (taking only into account the jobs have been assigned already). The approximation ratio of LPT is upper bounded by $1 + \sqrt{3}/3 \approx 1.58$ in the general case [44] and is exactly $\frac{4}{3} - \frac{1}{m}$ in the special case of identical machines (i.e., $P||C_{\max}$) [27]. If jobs are identical ($Q|p_i = p|C_{\max}$), LPT produces optimal schedules [12]. Both for $P||C_{\max}$ and $Q||C_{\max}$, *polynomial-time approximation schemes* (PTAS) have been devised by Hochbaum and Shmoys [34, 35].

2.3.2 Bin Packing

Problem Definition In a bin packing problem, we are given $n \in \mathbb{N}$ items with rational-valued item sizes $\varsigma_1, \dots, \varsigma_n \in (0, 1]$. The task is to pack them into a minimum number of unit-sized bins. That is, we seek to partition $[n]$ into a minimum number of subsets $B_1, \dots, B_m$ so that for each $j \in [m]$ it holds that $\sum_{i \in B_j} \varsigma_i \leq 1$. In the cost-sharing variant, each player corresponds to an item and the (optimal) service cost $C(S)$ is determined by the minimum number of bins needed for packing the items in S.

Algorithms With an easy reduction from 2-Partition [26], it is straightforward to see that bin packing is NP-hard to approximate within a factor of less than $\frac{3}{2}$. A simple approximation algorithm for bin packing is *next fit decreasing* (NFD), which assigns items in the order of decreasing size. Every item is put into the same bin as the preceding one if it fits; otherwise it is put into a new bin. Clearly, NFD can be implemented to run in time $O(n \cdot \log n)$. Its approximation guarantee is $2 \cdot \text{opt} - 1$; see, e.g., Hochbaum [33]. A better approximation algorithm is *first fit decreasing* (FFD), which also assigns items in the order of decreasing sizes but puts every item into the first bin where it fits. Using a balanced tree as data structure, it can be implemented to run in time $O(n \cdot \log n)$. The tight bound of FFD is $\frac{11}{9} \cdot \text{opt} + \frac{6}{9}$, [20]. It will prove useful later on that in the special case where all item sizes are a power of 2, FFD produces optimal solutions [15].

2.3.3 Network Problems

Most other problems considered in the cost-sharing literature are network problems (compare Section 1.5.1). While we occasionally refer to such problems, we will not elaborate on them in our application sections. We only give two examples:

Steiner Tree We are given a connected undirected graph $G = (V, E)$ and a distinct root node $r \in V$. Associated with each edge $e \in E$ is a non-negative edge cost. The task is to find a subgraph of G with minimum total edge cost so that for each $v \in V$ there is a path from v to r. In the cost-sharing variant, each player corresponds to a node and $C(S)$ is the cost of an optimal Steiner tree for the nodes in S (and r).

Metric Facility Location We are given a bipartite graph $G = (V, E)$. The partitioning of V is given by $V = N \cup F$, where N is the set of consumers and F is the set of facilities. Each facility $v \in F$ has opening costs f_v, and each edge $e \in E$ is associated with a connection cost c_e. The triangle inequality is fulfilled. The task is to find a set of facilities to be opened and a mapping of all consumers to an open facility, under the objective to minimize the sum of opening costs plus connection costs. In the cost-sharing variant, each player corresponds to a consumer and $C(S)$ is the cost of an optimal solution including only the consumers from S.

Chapter 3

Lexicographic Maximization: Beyond Cross-Monotonicity

3.1 Overview of Contribution

We devise a novel technique for the design of GSP mechanisms based on the following idea: Given a cost-sharing method ξ, a set S is called feasible if every player $i \in S$ can afford his cost share $\xi_i(S)$. Now choose the set of players that *lexicographically* maximizes the vector of all players' utilities, over all feasible sets. In contrast, Moulin mechanisms always choose the feasible set for which *all* players' utilities attain their maximum—which is only well-defined for cross-monotonic cost shares.

As an integral part of our new technique, we identify a property of cost-sharing methods that is sufficient to induce GSP mechanisms under lexicographic maximization. Cost-sharing methods that satisfy this property are called *valid*. Moreover, the resulting cost-sharing mechanisms are called *symmetric mechanisms*. For cost functions that are both symmetric and subadditive, we give a family of valid cost-sharing methods—and hence GSP symmetric mechanisms—that guarantee $(\sqrt{17}+1)/4$-BB. Interestingly, these cost-sharing methods assign at most two different cost shares, for any set of served players. Nevertheless, the aforementioned budget-balance factor is the best that *any* valid cost-sharing method can generally guarantee. Our result is a significant improvement over cross-monotonic methods (and thus Moulin mechanisms), which can only guarantee 2-BB (see Section 1.5.1). All computation needed for our new technique can be carried out efficiently: For the case of subadditive symmetric costs, we give algorithms both for computing the cost-sharing method and for computing the outcome of the mechanism itself. The total running time is $O(n^2)$.

As an application of our findings, we look at sharing the makespan cost when scheduling jobs on related machines ($Q||C_{\max}$). Since the makespan cost function is not symmetric if jobs are non-identical, we group all jobs by their processing requirements and then use symmetric mechanisms separately for each group. Altogether, this yields a GSP and $(d \cdot (\sqrt{17}+1)/4)$-BB mechanism, where d is the number of different processing requirements. Note that this result beats the previously best-known budget balance of $2d$ (see again Section 1.5.1). Together with a solution for the scheduling instance, the outcome of the mechanism can be computed in time $O(n^2 + n \cdot \log m)$. Hence, while symmetric costs might seem to be of limited practical interest, our technique can still be used in settings without symmetric costs. Unfortunately, though, the better budget balance compared to Moulin mechanisms comes at a price: We give an example with identical jobs on identical machines where our mechanisms achieve only $\Omega(n)$-EFF, whereas Brenner and Schäfer [10] gave Moulin mechanisms with $O(\log n)$-EFF. Nevertheless, we regard symmetric mechanisms as an important systematic first step for finding GSP mechanisms that perform better than Moulin mechanisms.

Unfortunately, we have to leave open an *exact characterization* of when cost-sharing methods induce GSP mechanisms under lexicographic maximization. Yet, we discuss a possible direction towards finding properties that are not based on symmetric costs.

Towards understanding the limitations imposed by GSP itself, we study the impact of symmetry of costs on 1-BB and GSP. An impossibility result can be shown already for this restricted class of cost functions: Even for just 4 players, we prove that symmetry of costs is not sufficient for the existence of a GSP and 1-BB mechanism. This is an exact bound, because for the case of only 3 players we do give a family of 1-BB and GSP mechanisms.

3.2 Symmetric Mechanisms

In this section, we develop our novel technique for designing GSP cost-sharing mechanisms. As an intermediate step, we define *precedence mechanisms*. The main idea (formalized below) is to always choose the set of players that lexicographically maximizes the utility vector. In contrast, Moulin mechanisms always perform a maximization of all components in the utility vector, which is only possible in the case of cross-monotonic cost shares. We remark that the name precedence mechanisms refers to the following fact: As a necessity for lexicographic maximization, the mechanism serves the players according to an order of precedence that is determined by the players' indices.

Given a fixed cost-sharing method ξ, we define the *benefit distribution* with respect to ξ as the function $P^{\xi} : \mathbb{R}^n \times 2^{[n]} \to \mathbb{R}^n$, where $P^{\xi}(\boldsymbol{v}, S) := (v_i \cdot s_i - \xi_i(S))_{i\in[n]}$. Here, $\boldsymbol{s} \in \{0,1\}^n$ is the service-allocation vector corresponding to S. Obviously, if the true valuation vector is $\boldsymbol{v}$ and a mechanism with cost-sharing method ξ serves the player set S, then $P^{\xi}(\boldsymbol{v}, S)$ is the vector of all players' utilities. By definition, $P^{\xi}(\boldsymbol{b}, S)$ is non-negative if and only if S is $\boldsymbol{b}$-feasible. We denote the set of all $\boldsymbol{b}$-feasible player sets by $\mathscr{F}(\boldsymbol{b})$. To simplify notation, we omit the superscript ξ when there is no confusion about ξ.

Now recall that the main ingredient of a Moulin mechanism $M = (Q, x)$ is a cross-monotonic cost-sharing method ξ, and M serves the maximal set S of players i who can afford their corresponding $\xi_i(S)$—due to cross-monotonicity, a unique maximal set always exists. Formally, $Q(\boldsymbol{b}) = \max \mathscr{F}(\boldsymbol{b})$, i.e., $Q(\boldsymbol{b})$ is always the greatest element of $\mathscr{F}(\boldsymbol{b})$ ordered by $\subseteq$. Due to cross-monotonicity, the largest feasible set is also a utility maximizer. Thus, an equivalent formalization of Moulin mechanisms using the above definition of P is $Q(\boldsymbol{b}) = \max(\arg\max_{S\in\mathscr{F}(\boldsymbol{b})}\{P(\boldsymbol{b}, S)\})$. The crucial idea is now the following generalization.

Definition 3.2.1. *A mechanism $M = (Q, x)$ with cost-sharing method ξ is called a* precedence mechanism *if, for all bid vectors $\boldsymbol{b}$, it chooses a set of players S so that the benefit distribution $P(\boldsymbol{b}, S)$ is lexicographically maximal over all $\boldsymbol{b}$-feasible sets S. Equivalently, $Q(\boldsymbol{b}) \in \arg\operatorname{lex}\max_{S\in\mathscr{F}(\boldsymbol{b})}\{P(\boldsymbol{b}, S)\}$.*

For historic reasons [9], we take the lexicographic order with *reversed significance* in Section 3.2: For instance, $(1,2) \succ (2,1)$. Likewise for sets, $\{2,4\} \succ \{1,2,3\}$.

We remark that in this generality, precedence mechanisms only satisfy NPT and VP, but not necessarily CS: As an example, let $n = 2$, $\xi_2(\{2\}) = 1$, and $\xi_2(\{1,2\}) > 1$. Now, if player 2 bids more than 1, player 1 will not be served, regardless of his bid. Also note that a precedence mechanism is not uniquely defined by its cost-sharing method because, in general, $\arg\operatorname{lex}\max_{S \in \mathscr{F}(\boldsymbol{b})}\{P(\boldsymbol{b}, S)\}$ contains more than one feasible set.

3.2.1 Symmetric Cost-Sharing Methods and Mechanisms

We define in this section a condition on cost-sharing methods that will turn out to be sufficient to induce precedence mechanisms that satisfy also CS and that are GSP. We first need:

Definition 3.2.2. *A cost-sharing method ξ is* symmetric *if for all $S, T \subseteq [n]$ with $|S| = |T|$ and all $i \in S$, $j \in T$ with* $\operatorname{rank}(i,S) = \operatorname{rank}(j,T)$ *it holds that $\xi_i(S) = \xi_j(T)$.*

Clearly, a symmetric cost-sharing method ξ is completely defined by $\xi([1]), \dots, \xi([n])$. In order to increase readability, we will therefore slightly abuse notation and simply write $\xi_\lambda(p)$ to denote $\xi_\lambda([p])$. Hence, for any arbitrary $S \subseteq [n]$ and $i \in S$, it holds that $\xi_i(S) = \xi_{\operatorname{rank}(i,S)}(|S|)$.

Given a fixed ξ, we always implicitly define two vectors $\boldsymbol{l} \in \mathbb{R}^n_{\geq 0}$ and $\boldsymbol{d} \in [n]^n$ as follows. For each cardinality $p \in [n]$, we say $l_p := \xi_p(p)$ is the ***l**ow cost share* and $d_p := |\{\lambda \in [p] \mid \xi_\lambda(p) > l_p\}|$ is the number of ***d**isadvantaged players* who pay more than the low cost share. Every cost share $\xi_\lambda(p) > l_p$ is called a ***h**igh cost share*.

We call a contiguous range $\{y \dots z\} \subseteq [n]$ of cardinalities with $d_y = 0$, $d_{z+1} = 0$ (or $z = n$), and $d_\lambda > 0$ for $\lambda \in \{y+1 \dots z\}$ a *segment*. Note that $d_1 = 0$ by definition. In order to improve readability, we stick to the convention of denoting ranks (in sets) by λ, μ, ν, players by i, j, k, and cardinalities by $p, r, s, \dots$.

Definition 3.2.3. *A symmetric cost-sharing method ξ is* valid *if for every segment $\{y \dots z\}$ and every cardinality $p \in \{y \dots z\}$ the following holds:*

V1) Cost shares are always non-increasing in the rank, i.e., $\xi_1(p) \geq \dots \geq \xi_p(p) = l_p$.

V2) Within a segment, low cost shares are constant. In general, they are non-increasing in the cardinality, i.e., $\forall r \in \{y \dots z\} : l_r = l_p$ and $\forall r \in \{z+1 \dots n\} : l_r \leq l_p$.

V3) Within a segment, high cost shares are non-increasing in the cardinality. That is, $\forall r \in \{p \dots z\} : \xi_{[d_p]}(r) \leq \xi_{[d_p]}(p)$.

V4) The number of players paying a low cost share is non-decreasing in the cardinality, i.e., $d_p \leq d_{p-1} + 1$. Moreover, if a high cost share has strictly decreased, all higher-precedence ranks are set to the low cost share; i.e., $\xi_\lambda(p) < \xi_\lambda(p-1) \Longrightarrow d_p \leq \lambda$.

Example 3.2.4. *We give two valid symmetric cost-sharing methods for illustration:*

p	l_p	d_p	$\xi_1(p),\dots,\xi_p(p)$
1	1	0	1
2	1	1	5, 1
3	1	2	5, 4, 1
4	1	3	5, 4, 2, 1
5	1	2	5, 3, 1, 1, 1

p	l_p	d_p	$\xi_1(p),\dots,\xi_p(p)$
1	1	0	2
2	1	1	4, 2
3	1	1	3, 2, 2
4	1	2	3, 3, 2, 2
5	1	0	1, 1, 1, 1, 1

The symmetric cost-sharing method on the left side consists only of the single segment $\{1\dots5\}$, whereas the right method has the two segments $\{1\dots4\}$, $\{5\}$. If $S=\{2,4\}$, then $\xi(S)=(0,5,0,1,0)$ for the left method and $\xi(S)=(0,4,0,2,0)$ for the right method.

Definition 3.2.5. *Let ξ be a valid symmetric cost-sharing method. The precedence mechanism uniquely defined by $Q(\boldsymbol{b}) := \operatorname{lex\,max}(\arg\operatorname{lex\,max}_{S\in\mathscr{F}(\boldsymbol{b})}\{P(\boldsymbol{b},S)\})$ is called a* symmetric mechanism.

Roughly speaking, valid symmetric cost-sharing methods satisfy a property reminiscent to a "half-sided" version of cross-monotonicity:

Lemma 3.2.6. *Let ξ be a valid symmetric cost-sharing method. Inclusion of a lower-precedence player never makes a higher-precedence player worse off, i.e., for every cardinality $p\in[n-1]$ and every rank $\lambda\in[p]$ it holds that $\xi_\lambda(p)\ge\xi_{\lambda+1}(p+1)$.*

Proof. Let $p\in[n-1]$ and $\lambda\in[p]$. If $\lambda\le d_p$, then $\xi_{\lambda+1}(p+1)\le\xi_\lambda(p+1)\le\xi_\lambda(p)$ due to (V3) and (V1). If $\lambda>d_p$, then $\lambda+1>d_{p+1}$ due to (V4) and $\xi_{\lambda+1}(p+1)=l_{p+1}\le l_p=\xi_\lambda(p)$ due to (V2). □

The previous property guarantees that symmetric mechanisms indeed fulfill our requirements for cost-sharing mechanisms:

Lemma 3.2.7. *Symmetric mechanisms satisfy NPT, VP, and CS.*

Proof. CS is the only non-obvious property to show: Let $\boldsymbol{b}$ be a bid vector, $i\in[n]$ be a player, and $S\subseteq[n]\setminus i$ be an arbitrary set of players that does not contain i. Suppose $b_i>\max_{p\in[n]}\xi_1(p)$. We show that a symmetric mechanism would not choose S. There are two cases:

- Case $S=\emptyset$ or $i<\min S$:

  Define $S':=S\cup i$. Then $|S'|=|S|+1$ and for all $j\in S'\setminus i$ it holds that $\operatorname{rank}(j,S')=\operatorname{rank}(j,S)+1$. Therefore, $\xi_j(S')\le\xi_j(S)$ due to Lemma 3.2.6.

- Otherwise:

  Define $k:=\max(S\cap[i])$ and $S':=(S\setminus k)\cup i$. Then $|S'|=|S|$ and for all $j\in S'\setminus i$ it holds that $\operatorname{rank}(j,S')=\operatorname{rank}(S)$. Therefore, $\xi_j(S')=\xi_j(S)$ because ξ is a symmetric cost-sharing method.

In both cases, we have $\xi_i(S') < b_i$ and for all $j \in S' \setminus i$ that $\xi_j(S') \le \xi_j(S) \le b_j$. Hence, S' is $\boldsymbol{b}$-feasible and $P(\boldsymbol{b}, S') \succ P(\boldsymbol{b}, S)$. This completes the proof. □

Theorem 3.2.8. *Symmetric mechanisms are GSP.*

Proof. Let $M = (Q, x)$ be a symmetric mechanism. The proof of GSP is by way of contradiction: Let $\boldsymbol{v}$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \ge u_K(\boldsymbol{v})$. Suppose there is a player $i \in K$ with $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$. We will show that the mechanism would not have chosen $Q(\boldsymbol{v})$ for input $\boldsymbol{v}$ then.

Denote $S := Q(\boldsymbol{v})$, $l := l_{|S|}$, and $T := Q(\boldsymbol{b})$. Note that T is also $\boldsymbol{v}$-feasible. Moreover, let $\{y \dots z\} \ni |S|$ be the segment that $|S|$ is in. We partition S into $S' := S \cap [i]$ and $S'' := S \cap \{i+1 \dots n\}$. Likewise, we partition T into $T' := T \cap [i]$ and $T'' := T \cap \{i+1 \dots n\}$. Finally, let $s' := |S'|$, $s'' := |S''|$, $t' := |T'|$, $t'' := |T''|$, and define

$$p := \min\left\{ r \in \{t' + s'' \dots |T \cup S''|\} \;\middle|\; \xi_{[t']}(r) \le \xi_{[t']}(|T|) \right\}. \tag{3.2.9}$$

For any cardinality $r \in \{t' + s'' \dots |T \cup S''|\}$, define

$$R_r := T' \cup S'' \cup MIN_{r - t' - s''}(T'' \setminus S'').$$

Note that T', S'', and $MIN_{r-t'-s''}(T'' \setminus S'')$ are pairwise disjoint. Moreover, the definition ensures that $i \in R_r$ and $|R_r| = r$.

We show in the rest of the proof that for input $\boldsymbol{v}$ a precedence mechanisms would rather have chosen R_p instead of S. We start by verifying that p is well-defined and $p \in \{y \dots z\}$:

i) $|T| \le z$, $v_i > \xi_i(T) \ge l$, and $|T \cup S''| \le z$

   Let $\{y' \dots z'\} \ni |T|$ be the segment that $|T|$ is in. By way of contradiction, assume that $|T| > z$. Then, all players in S and T have a true valuation of at least $l_{y'}$, so $R_{y'}$ is $\boldsymbol{v}$-feasible and $P(\boldsymbol{v}, R_{y'}) \succ P(\boldsymbol{v}, S)$. A contradiction. Consequently, $|T| \le z$ and we also have $v_i > \xi_i(T) \ge l$.

   Now, all players in $T \cup S$ have a true valuation of at least $l \ge l_{z+1}$. So if $|T \cup S''| > z$ then R_{z+1} would be $\boldsymbol{v}$-feasible and $P(\boldsymbol{v}, R_{z+1}) \succ P(\boldsymbol{v}, S)$, a contradiction.

ii) $y \le s''$ and $y \le |T|$

   Define $S''_+ := \{j \in S'' \mid v_j > l\}$. By way of contradiction, assume $y > |S''_+|$. Define $R' := S''_+ \cup i \cup MAX_{y - |S''_+| - 1}(S \setminus (S''_+ \cup i))$. Then $|R'| = y$, $i \in R'$, and R' is $\boldsymbol{v}$-feasible, but $P(\boldsymbol{v}, R') \succ P(\boldsymbol{v}, S)$. This is a contradiction. Hence, $y \le |S''_+| \le s''$.

   By way of contradiction, assume that $y > |T|$. Note that for all $j \in S''_+ \setminus T$ it holds that $u_j(\boldsymbol{b}) = 0 < u_j(\boldsymbol{v})$, so $j \notin K$ and $b_i = v_i$. Define $R'' := T \cup MAX_{y - |T|}(S''_+ \setminus T)$. Then $|R''| = y$ because $y - |T| \le |S''_+ \setminus T|$. Moreover, R'' is $\boldsymbol{b}$-feasible, but $P(\boldsymbol{b}, R'') \succ P(\boldsymbol{b}, T)$. A contradiction.

iii) p is well-defined and $p \in \{y \dots z\}$

Since $y \leq |T| \leq |T \cup S''| \leq z$ and due to (V3), we have that $\xi_{[t']}(|T \cup S''|) \leq \xi_{[t']}(|T|)$. Moreover, also $y \leq s'' \leq t' + s'' \leq |T \cup S''|$, so p is well-defined and $p \in \{y \dots z\}$.

Finally, we show that R_p is $\boldsymbol{v}$-feasible. By (3.2.9), it holds for all players $k \in T'$ that $\xi_k(R_p) \leq \xi_k(T) \leq v_k$. Therefore, fix an arbitrary player $k \in R_p \setminus T'$. There are three cases:

- Case $p < |S|$:

  Since $t' + s'' \leq p$, this implies $1 \leq t' \leq p - s'' < |S| - s'' = s'$.

  If $i \notin S$ and $v_i > \xi_{s'}(|S|)$ then $R' := (S \setminus \max S') \cup i$ is $\boldsymbol{v}$-feasible but $P(\boldsymbol{v}, R') \succ P(\boldsymbol{v}, S)$; hence, $i \notin S$ implies $\xi_i(T) < v_i \leq \xi_{s'}(|S|)$. On the other hand, if $i \in S$, then $\xi_i(T) < \xi_i(S) = \xi_{s'}(|S|)$.

  In both cases, we get

  $$\begin{aligned} \xi_{t'}(p) &\leq \xi_{t'}(|T|) && \text{due to (3.2.9)} \\ &= \xi_i(T) < \xi_{s'}(|S|) && \text{as explained} \\ &\leq \xi_{t'}(|S|) && \text{due to (V1)}. \end{aligned}$$

  Now due to (V3), this is only possible if $t' > d_p$. Consequently, $\xi_k(R_p) = l$.

- Case $p \geq |S|$ and $p = t' + s''$:

  Then $R_p \setminus T' = S''$; so $\xi_k(R_p) \leq \xi_k(S)$ because of $p \geq |S|$ and Lemma 3.2.6.

- Case $p \geq |S|$ and $p > t' + s''$:

  Then $\xi_{[t']}(p) < \xi_{[t']}(p-1)$ by (3.2.9) and therefore $\xi_k(R_p) = l$ due to (V4).

Hence, R_p is $\boldsymbol{v}$-feasible. Now, $P(\boldsymbol{v}, R_p) \succ P(\boldsymbol{v}, S)$ because $\xi_k(R_p) \leq \xi_k(S)$ for all $k \in S''$ and $\xi_i(R_p) \leq \xi_i(T)$. This is a contradiction and completes the proof. □

3.2.2 Computing the Outcome of Symmetric Mechanisms

In the rest of this section, we show that the outcome of symmetric mechanisms can be efficiently computed by Algorithm 3.1. We start by giving some intuition. Afterwards, we provide a formal verification of correctness.

Algorithm 3.1 takes an optimistic approach in that it starts with the full player set $Q = [n]$ and then removes players as long as Q is not $\boldsymbol{b}$-feasible. We divide the algorithm into three phases. The first phase consists of lines 1–2, and finds the largest set Q so that all players in Q can afford $l_{|Q|}$, i.e., the low cost share corresponding to cardinality $|Q|$. This phase is reminiscent to a Moulin mechanism—in particular, if high cost shares would never be used (i.e., all d_r's were 0), then the symmetric cost-sharing method is cross-monotonic, and phase 1 works *exactly* as the corresponding Moulin mechanism.

```
Input:  valid symmetric cost-sharing method $\xi$, bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output: set of players $Q \in 2^{[n]}$, cost distribution $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$
1: $Q := [n]$
2: while $\exists i \in Q : b_i < l_{|Q|}$ do $Q := \{i \in Q \mid b_i \geq l_{|Q|}\}$
3: $I := \{i \in Q \mid b_i = l_{|Q|}\}$; $H := \emptyset$; $p := \max(\{r \in [|Q|] \mid d_r = 0\} \cup \{0\})$
4: while $p < |Q \setminus I|$ do
5:     $j := \min(Q \setminus H)$; $I := I \setminus j$
6:     if $b_j \geq \xi_j(Q)$ then $H := H \cup j$; $p := \max\{r \in [|Q|] \mid d_r = |H|\}$
7:     else $Q := Q \setminus j$
8: $Q := Q \setminus MIN_{|Q|-p}\, I$; $\boldsymbol{x} := \xi(Q)$
```

Algorithm 3.1: Symmetric mechanisms

In the second phase, from line 3–7, least-precedence players are offered a high cost share if including them for the low share $l_{|Q|}$ would require dropping higher-precedence players who are not contained in I. Note here that the set I contains all remaining players who are indifferent to being served for $l_{|Q|}$, and H contains all players i already assigned a high cost share $\xi_i(Q) > l_{|Q|}$. Moreover, p contains the maximum number of players in Q that could be served when only the players in H pay a high cost share. Player j is of least precedence among all remaining players that have not yet been assigned a high cost share.

The third phase consists only of line 8: Here, indifferent players are dropped for the benefit of players with a lower precedence.

Lemma 3.2.10. *Suppose Algorithm 3.1 is given a valid symmetric cost-sharing method and an arbitrary bid vector as input. The following holds:*

i) *In every iteration of the while loop in the second phase, either p increases or $|Q|$ decreases; but not both within the same iteration.*

ii) *The algorithm can be implemented to terminate after at most $O(n^2)$ steps.*

iii) *Let $\{y \ldots z\}$ be the segment that $|Q|$ is in after the first phase. Then, $|Q|$ stays within this segment until the algorithm terminates.*

iv) *In line 6, if $b_j \geq \xi_j(Q)$ is fulfilled, then $\xi_j(Q)$ is the price that player j will be charged at the end of the algorithm. In particular, in the third phase, all players in H are charged a high cost share, and all players in $Q \setminus H$ are charged the low cost share $l_{|Q|}$.*

Proof. i) We show the claim together with the following invariant, which holds during and immediately after the second phase: "$\forall r \in \{p+1 \ldots |Q|\} : |H| < d_r$". Clearly, the invariant holds immediately after line 3, so consider an arbitrary iteration and assume that the invariant holds immediately before line 5. Let the updated variable values at the end of the same iteration (i.e., immediately after line 7) be indicated by a star (*). One of the following two cases happens:

- In line 6, a player is added to H, i.e., $|H^*| = |H| + 1$. Since $p < |Q|$, $d_{|Q|} > |H|$, and due to (V4), there must be a cardinality $r \in \{p+1 \dots |Q|\}$ with $d_r = |H^*|$. It follows that $p^* > p$ and $\forall r \in \{p^*+1 \dots |Q|\} : |H^*| < d_r$.
- In line 7, a player is removed from Q, i.e., $|Q^*| < |Q|$.

Clearly, the invariant continues to hold in both cases.

ii) Every iteration of the while loop in the first phase decreases $|Q|$ by at least 1. Similarly, every iteration of the while loop in the second phase decreases $|Q \setminus H|$ by 1. Hence, the algorithm terminates. By sorting players' bids, the first phase can be implemented to take at most $O(n \cdot \log n)$ steps. The second phase takes $O(n^2)$ steps and the third phase $O(n)$ steps.

iii) In the second phase, in line 3, p is initialized with $p := y$. During the rest of the algorithm, p is non-decreasing, $|Q|$ is non-increasing, and $p \leq |Q|$ is an invariant throughout the algorithm. Together with the invariant from (i), this implies that always $|Q| \in \{y \dots z\}$.

iv) Consider an arbitrary iteration, immediately after line 7. By the invariant from (i), it holds for all $r \in \{p+1 \dots |Q|\}$ that $|H| < d_r$, and thus by (V3) and (V4) also that $\xi_{[|H|]}(r) = \xi_{[|H|]}(|Q|)$. Moreover, after line 8, it holds that $|Q| = p$. By the previous assignment in line 3 or 6, $d_p = |H|$. This completes the proof. □

Theorem 3.2.11. *Let ξ be a valid symmetric cost-sharing method. Then, for all bid vectors $\boldsymbol{b}$, Algorithm 3.1 computes the outcome of the respective symmetric mechanism.*

Proof. Consider an arbitrary input ξ and $\boldsymbol{b}$. Denote by S the set chosen by the respective symmetric mechanism, and by $(Q^*, \boldsymbol{x}^*)$ the outcome returned by Algorithm 3.1. Note first that no player dropped in the first phase can be contained in S as otherwise S would not be $\boldsymbol{b}$-feasible. Hence, the loop invariant $S \subseteq Q$ holds immediately before line 3. Define $l := l_{|Q|}$.

Now consider the second phase. We show that, again, no player dropped here can be contained in S. Therefore, consider an iteration of the while loop and fix all variable values immediately before line 5. Assume that the loop invariant $S \subseteq Q$ holds. Note that $|S|$ and $|Q|$ are in the same segment, due to Lemma 3.2.10 (iii). Let $j := \min(Q \setminus H)$ be the player who is either dropped or added to H in this iteration. It holds that $l \leq b_j < \xi_j(Q)$ because $\operatorname{rank}(j, Q) = |H| + 1 = d_p + 1 \leq d_{|Q|}$. We only need to consider the case that j is dropped in line 7, i.e., $j \notin Q^*$; otherwise, Q will not be changed in this iteration and the loop invariant $S \subseteq Q$ continues to hold.

By way of contradiction, assume $j \in S$ and thus $\xi_j(S) < \xi_j(Q)$. This implies $\xi_j(S) = l$ because otherwise $\operatorname{rank}(j, S) \leq d_{|S|}$ and therefore $\xi_j(S) = \xi_{\operatorname{rank}(j,S)}(|S|) \geq \xi_{\operatorname{rank}(j,S)}(|Q|) \geq \xi_{\operatorname{rank}(j,Q)}(|Q|) = \xi_j(Q)$. Here the inequalities are due to (V3) and (V1). Since $S \subseteq Q$ and $\xi_j(S) = l < \xi_j(Q)$, it must hold that $d_{|S|} \leq |H|$. Therefore, we have $|S| \leq p = \max\{r \in [|Q|] \mid d_r = |H|\}$. Since $p < |Q \setminus I|$, there are $r > p - |H|$ players $k \in Q \cap \{j+1 \dots n\}$ with $b_k > l$. Note here that (V4) further implies

$r > p - |H| \geq |S| - d_{|S|} \geq |S \cap \{j \dots n\}|$. Among these r players, at least one player k is not included in S. Now, $R := (S \setminus j) \cup k$ is $\boldsymbol{b}$-feasible and $P(\boldsymbol{b}, R) \succ P(\boldsymbol{b}, S)$. A contradiction.

Finally, consider phase 3 and fix the variable values immediately before line 8. Sufficiently many players k with $b_k = l$ are removed so that *all* players $k \in Q \setminus H$ with $b_k > l$ receive the service for l. Since $S \subseteq Q$, this implies $P(\boldsymbol{b}, Q^*) \succeq P(\boldsymbol{b}, S)$. It remains to be shown that Q^* is lexicographically maximal. This holds because only the least necessary number of indifferent players $|Q| - p$ are removed and these players are of least precedence within I. □

3.3 Symmetric Mechanisms for Symmetric Subadditive Costs

In this section, we devise an efficient algorithm for computing $\frac{\sqrt{17}+1}{4} \approx 1.28$-BB valid symmetric cost-sharing methods for arbitrary symmetric subadditive costs $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$. For simplicity of notation, we will specify the costs C by a function (or array) $c : [n] \to \mathbb{R}_{\geq 0}$. That is, for any non-empty set S we have $C(S) = c(|S|)$.

The valid symmetric cost-sharing methods we propose are of a particularly simple structure in that for any set of served players, at most two different cost shares are assigned. Therefore, the cost-sharing methods can be specified in a succinct way.

Definition 3.3.1. *A* two-price cost-sharing form *(2P-CSF) is a 4-tuple $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ with $n \in \mathbb{N}$, $\boldsymbol{d} \in [n]^n$, and $\boldsymbol{h}, \boldsymbol{l} \in \mathbb{R}^n_{\geq 0}$ that specifies a symmetric cost-sharing method ξ as follows: For every cardinality $p \in [n]$ and every rank $\lambda \in [p]$, define $\xi_\lambda(p) := h_p$ if $\lambda \leq d_p$ and $\xi_\lambda(p) := l_p$ otherwise.*

Corresponding to the previous section, we denote by l_p the low cost share and by h_p the high cost share used for cardinality $p \in [n]$. Moreover, d_p specifies the number of disadvantaged players who pay the high cost share. Note that only the least-precedence players pay the high cost share. We say a 2P-CSF is *valid* if it specifies a valid symmetric cost-sharing method.

3.3.1 Computing Two-Price Cost-Sharing Forms

We compute 2P-CSFs with Algorithm 3.2. The intuition is as follows: The minimum average per-player cost up to the respective cardinality is used as the low cost share. More precisely, we also multiply with the budget-balance factor to leave some flexibility when the average cost increases for a larger cardinality. Now, if assigning all players the low cost share recovers the total cost, no player has to pay a high cost share.

Otherwise, the general idea is to let the least-precedence player pay the rest. However, the high cost share cannot increase within a segment (V3), and therefore it might have to be assigned to two players. Now, as long as there is more than one disadvantaged player, the high cost share cannot change (V4). Hence, special care is needed to ensure that the high cost share is neither too large nor too small.

Note that "∞" is used as a placeholder for any "sufficiently large" value in order to simplify the presentation (a value strictly larger than $\beta \cdot c(s)$ is sufficient).

```
Input:   function $c : [n] \to \mathbb{R}_{\geq 0}$ that specifies symmetric costs $C$,
         BB parameter $\beta \geq \frac{\sqrt{17}+1}{4}$
Output:  2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$
 1: $d_1 := 0$; $h_1 := \infty$; $l_1 := \beta \cdot c(1)$; $s := 1$
 2: for $p := 2, \dots, n$ do
 3:     if $\beta \cdot \frac{c(p)}{p} \leq l_s$ then $d_p := 0$; $h_p := \infty$; $l_p := \beta \cdot \frac{c(p)}{p}$; $s := p$
 4:     else
 5:         $d_p := 1$; $l_p := l_{p-1}$; $h_p := \min\{\beta \cdot c(p) - (p-1) \cdot l_p, h_{p-1}\}$
 6:         if $p \cdot l_p \geq c(p)$ then
 7:             $d_p := 0$; $h_p := \infty$
 8:         else if $h_p + (p-1) \cdot l_p < c(p)$ then
 9:             $d_p := 2$
10:         else if $2 \cdot c(s) > h_p + (p-1) \cdot l_p > (\beta^2 - \beta) \cdot c(s) + (p-1) \cdot l_p \geq c(p)$ then
11:             $h_p := (\beta^2 - \beta) \cdot c(s)$
```

Algorithm 3.2: Two-price cost-sharing forms

Lemma 3.3.2. *Suppose Algorithm 3.2 is given symmetric subadditive costs $c : [n] \to \mathbb{R}_{\geq 0}$ as input. Then, the output 2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ is valid and the algorithm terminates after $O(n)$ steps.*

Proof. Consider an arbitrary cardinality $p \in [n]$. Clearly, $d_p \in \{0, 1, 2\}$ and $d_1 = 0$.

(V1) $l_p < h_p$

This can be verified by induction. The *base case* $p = 1$ holds as $h_1 = \infty > \beta \cdot c(1) = l_1$. For the *induction step* $p - 1 \to p$, assume that $l_{p-1} < h_{p-1}$. Since there is nothing to show if $h_p = \infty$, assume $h_p < \infty$. Then the condition in line 3 evaluated to false for cardinality p. Let $s := \max\{r \in [p-1] \mid \beta \cdot \frac{c(r)}{r} = l_p\}$ be the last cardinality previous to p for which the lower cost share was set in line 1 or line 3. Now, $\frac{c(p)}{p} > \frac{c(s)}{s} = \frac{l_s}{\beta}$ and $l_p = l_{p-1} = \cdots = l_s$. In line 5, h_p was set for the first time, either to $h_{p-1} > l_{p-1} = l_p$ or to $\beta \cdot c(p) - (p-1) \cdot l_p > l_p$. If h_p was set again in line 11, then $p \cdot l_p < c(p)$ because line 6 evaluated to false and $(\beta^2 - \beta) \cdot c(s) + (p-1) \cdot l_p \geq c(p)$ because line 10 evaluated to true. Hence, $h_p = (\beta^2 - \beta) \cdot c(s) \geq c(p) - (p-1) \cdot l_p > l_p$.

(V2) $l_p \leq l_{p-1}$ and $(l_p < l_{p-1} \Longrightarrow d_p = 0)$

This holds since $l_p \neq l_{p-1}$ only if l_p was set in line 3.

(V3) $h_p > h_{p-1} \Longrightarrow d_p = 0$

This holds since line 5 ensures that from $d_p > 0$ it follows that $h_p \leq h_{p-1}$. Note here that in line 11, the value of h_p was not increased due to the condition in line 10.

(V4) $d_p \leq d_{p-1} + 1$ and $(h_p < h_{p-1} \Longrightarrow d_p \leq 1)$

This is fulfilled trivially if $d_p \leq 1$. So assume $d_p = 2$. Then $h_p < c(p) - (p-1) \cdot l_k$ because line 8 evaluated to true in iteration p and thus $h_p = \min\{\beta \cdot c(p) - (p-1) \cdot l_p, h_{p-1}\} = h_{p-1}$. Now assume $d_p > d_{p-1} + 1$, i.e., $d_{p-1} = 0$. Then, $h_p = h_{p-1} = \infty$, a contradiction to $h_p < c(p) - (p-1) \cdot l_p$.

It is trivial to see that the running time of the algorithm is $O(n)$. □

Theorem 3.3.3. *Given symmetric, subadditive, and non-decreasing costs $c : [n] \to \mathbb{R}_{\geq 0}$ and an arbitrary parameter $\beta \geq \frac{\sqrt{17}+1}{4}$, Algorithm 3.2 efficiently computes a valid and β-BB 2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$.*

Proof. Due to Lemma 3.3.2, it only remains to be shown that $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ is β-BB. Let $\gamma : [n] \to \mathbb{R}_{\geq 0}$, $\gamma(p) := d_p \cdot h_p + (p - d_p) \cdot l_p$, be the *recovered cost*. Consider now an arbitrary cardinality $p \in [n]$. Like in iteration p of the algorithm, let $s := \max\{r \in [p] \mid \beta \cdot \frac{c(r)}{r} = l_p\}$ be the last cardinality previous or equal to p for which the lower cost share was set in lines 1 or 3.

- Case $d_p = 0$:

  For the upper bound, note that $\gamma(p) = p \cdot l_s \leq p \cdot \beta \cdot \frac{c(p)}{p} = \beta \cdot c(p)$. The lower bound $\gamma(p) \geq c(p)$ holds by line 6.

- Case $d_p = 1$:

  Since the value of h_p is never increased in line 11, it holds due to line 5 that $h_p \leq \beta \cdot c(p) - (p-1) \cdot l_p$ and thus $\gamma(p) = h_p + (p-1) \cdot l_p \leq \beta \cdot c(p)$. Furthermore, since the condition in line 8 evaluated to false and since the value of h_p is only decreased in line 11 to $(\beta^2 - \beta) \cdot c(s)$ if line 10 evaluated to true, it holds that $\gamma(p) \geq c(p)$.

Before considering the case $d_p = 2$, we need two general observations: First, we show

$$h_p \geq (\beta^2 - \beta) \cdot c(s). \tag{3.3.4}$$

Obviously, (3.3.4) holds when $h_p = \infty$. Therefore assume $h_p < \infty$, i.e., $d_p > 0$ and $s < p$. By line 6 we then also have $c(p) > p \cdot l_p > \beta \cdot c(s)$. Consequently, $\beta \cdot c(p) - (p-1) \cdot l_p > \beta \cdot c(p) - c(p) + l_p > (\beta - 1) \cdot c(p) > (\beta^2 - \beta) \cdot c(s)$. Hence, by line 5, if $h_{p-1} \geq (\beta^2 - \beta) \cdot c(s)$, then also $h_p \geq (\beta^2 - \beta) \cdot c(s)$. Since p was chosen arbitrarily, this proves (3.3.4).

Next, we show

$$c(p) \leq 2 \cdot c(s). \tag{3.3.5}$$

Let $t := \min(\{r \in \{p+1 \ldots n\} \mid \beta \cdot \frac{c(r)}{r} \leq l_p\} \cup \{n+1\})$ be the next cardinality after p for which the lower cost share was set in lines 1 or 3 (or $t = n+1$ if s is the largest such cardinality). Then $s \leq p < t \leq 2s$. Otherwise, if $t > 2s$, then t would not be minimal due to $\frac{c(2s)}{2s} \leq \frac{2 \cdot c(s)}{2s} = \frac{c(s)}{s}$ because of subadditivity. Since c is also non-decreasing, $c(p) \leq c(t) \leq c(s) + c(t-s) \leq 2 \cdot c(s)$.

- Case $d_p = 2$:

  Define $p' := \max\{r \in [p] \mid d_r = 1\}$, and let $y := \max\{r \in [p] \mid d_p = 0\}$ be the start of the segment that p is in. Then, $s \le y < p' < p$, $d_{y+1} = 1$, $d_{p'+1} = 2$, and $h_{y+1} \ge h_p = h_{p'}$. We show

  $$2 \cdot c(s) > h_{y+1} + y \cdot l_{y+1} \,. \tag{3.3.6}$$

  By way of contradiction, assume $2 \cdot c(s) \le h_{y+1} + y \cdot l_{y+1}$. Then, $c(p'+1) \le c(p) \le 2 \cdot c(s) \le h_{y+1} + y \cdot l_{y+1}$ due to (3.3.5). Observe now that the value $h_r + (r-1) \cdot l_r$ is increasing in $r \in \{y+1 \dots p'+1\}$, because lines 3, 6, 8, and 10 evaluated to false for iterations $y+1, \dots, p'$. This is a contradiction to $d_{p'+1} = 2$ and to the fact that line 8 must have evaluated to true in iteration $p'+1$. Therefore, (3.3.6) holds.

  As a next step, we show

  $$h_p = (\beta^2 - \beta) \cdot c(s) \,. \tag{3.3.7}$$

  Recall (3.3.4) and assume, by way of contradiction, that $h_p > (\beta^2 - \beta) \cdot c(s)$. Since $h_{y+1} \ge h_p$, line 10 must have evaluated to false for cardinality $(y+1)$. Since the other two inequalities of line 10 are fulfilled according to (3.3.6) and by our assumption $h_p > (\beta^2 - \beta) \cdot c(s)$, it must hold that $(\beta^2 - \beta) \cdot c(s) + y \cdot l_{y+1} < c(y+1)$. This implies $\beta^2 \cdot c(s) = \beta^2 \cdot c(s) - \beta \cdot c(s) + s \cdot l_{y+1} \le (\beta^2 - \beta) \cdot c(s) + y \cdot l_{y+1} < c(y+1)$. Moreover, by definition of y and due to line 5, we have $h_{y+1} = \beta \cdot c(y+1) - y \cdot l_{y+1}$. Then, however, $2 \cdot c(s) < \beta^3 \cdot c(s) < \beta \cdot c(y+1) = h_{y+1} + y \cdot l_{y+1}$. This is a contradiction to (3.3.6), which then proves (3.3.7)

  Now, we get as lower bound

  $$\begin{aligned} \gamma(p) &= 2h_p + (p-2) \cdot l_p \\ &= 2h_p + (p-2) \cdot \beta \cdot \frac{c(s)}{s} && \text{due to the definition of } s \\ &\ge (2\beta^2 - \beta) \cdot c(s) && \text{due to (3.3.7) and } p-2 \ge s \\ &\ge 2 \cdot c(s) && \text{due to } \beta \ge \tfrac{\sqrt{17}+1}{4} \\ &\ge c(p) && \text{due to (3.3.5)} \,. \end{aligned}$$

  For the upper bound, note that $\beta \cdot c(s) < p \cdot l_p < c(p)$ due to line 6. Hence,

  $$\begin{aligned} \gamma(p) &= 2h_p + (p-2) \cdot l_p \\ &= h_p - l_p + h_p + (p-1) \cdot l_p \\ &< h_p + c(p) && \text{due to line 8} \\ &= (\beta - 1) \cdot \beta \cdot c(s) + c(p) && \text{due to (3.3.7)} \\ &< (\beta - 1) \cdot c(p) + c(p) && \text{as explained} \\ &= \beta \cdot c(p) \,. \end{aligned}$$

  □

3.3.2 Lower Bound on the Performance of Symmetric Mechanisms

We show that $\frac{\sqrt{17}+1}{4}$-BB is in general the best that can be achieved by valid symmetric cost-sharing methods.

Theorem 3.3.8. *For all $\varepsilon > 0$, there is a symmetric, subadditive, and non-decreasing cost function c for which no valid $\left(\frac{\sqrt{17}+1}{4} - \varepsilon\right)$-BB symmetric cost-sharing method exists.*

Proof. Fix $\beta := \frac{\sqrt{17}+1}{4}$, let $0 < \varepsilon \leq \beta - 1$, and set $\alpha := \beta - \varepsilon$. Additionally, let $p, l \in \mathbb{N}$ with $l > \log_\beta \frac{\beta-1}{\varepsilon}$ and $p > \frac{(l+1)\cdot\alpha}{\varepsilon} = \frac{(l+1)\cdot\beta}{\varepsilon} - (l+1)$. Set $r := p + l + 1$ and $n := r + 1$ and consider function c defined below:

k	$1 \cdots p$	$p+1$	$p+2$	$\cdots$	$p+l$	r	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 and non-decreasing. By way of contradiction, assume there is a valid symmetric cost-sharing method ξ which is α-BB. For all $s \in [n]$, we define $\gamma(s) := d_s \cdot h_s + (s - d_s) \cdot l_s$. Moreover, let $h_s := \xi_1(s)$ be the largest cost share used for cardinality s.

The idea is the following: It can be shown that $d_r \geq 1$ and $d_n = d_r + 1$. Let $y := \max\{s \in [r-1] \mid d_s = 0\}$ be the start of the segment that cardinality n is in. Then $y < r$ and $h_n \leq h_{y+1}$ due to (V3). By case analysis, $h_{y+1} \leq \alpha \cdot c(y+1) - c(y) < \beta^2 - \beta$. Thus $\gamma(n) \leq h_n + \alpha \cdot \beta < 2 \cdot \beta^2 - \beta = 2 = c(n)$, a contradiction to α-BB. In detail, we can show the following:

- $d_r \geq 1$: Otherwise, $d_r = 0$ and we would obtain a contradiction to α-BB:

$$\gamma(r) = r \cdot l_r \leq r \cdot l_p \leq r \cdot \frac{\alpha}{p} = \alpha \cdot \left(1 + \frac{l+1}{p}\right) < \alpha \cdot \left(1 + \frac{\varepsilon}{\alpha}\right) = \beta = c(r).$$

- $d_n = d_r + 1$: Otherwise, $d_n \leq d_r$ and we again obtain a contradiction to α-BB:

$$\begin{aligned}\gamma(n) &\leq \gamma(r) + l_n \leq \alpha \cdot \beta + l_n < \beta^2 + \frac{\alpha}{p} \\ &< \beta^2 + \frac{\varepsilon}{l+1} \leq \beta^2 + \frac{\varepsilon}{2} \leq \beta^2 + \frac{\beta-1}{2} < 2.\end{aligned}$$

- Bounds on h_{y+1}: Due to $\alpha \cdot c(y+1) \geq \gamma(y+1) = h_{y+1} + \gamma(y) \geq h_{y+1} + c(y)$, it holds that $h_{y+1} \leq \alpha \cdot c(y+1) - c(y)$. There are three cases:
  - Case $y \in [p-1]$: Then

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

– Case $y \in \{p \dots p+l-1\}$: Let $s := y+1-p$. Then

$$\begin{aligned} h_{y+1} &\leq \alpha \cdot c(y+1) - c(y) \\ &= \alpha \cdot c(p+s) - c(p+s-1) \\ &= \alpha \cdot \left(\beta - \frac{\beta-1}{\beta^s}\right) - \left(\beta - \frac{\beta-1}{\beta^{s-1}}\right) \\ &< \beta^2 - \frac{\beta-1}{\beta^{s-1}} - \left(\beta - \frac{\beta-1}{\beta^{s-1}}\right) \\ &= \beta^2 - \beta\,. \end{aligned}$$

– Case $y = p+l = r-1$: Then

$$\begin{aligned} h_{y+1} &\leq \alpha \cdot c(r) - c(r-1) = \alpha \cdot \beta - \left(\beta - \frac{\beta-1}{\beta^l}\right) \\ &< \alpha \cdot \beta - (\beta - \varepsilon) = \alpha \cdot (\beta - 1) < \beta^2 - \beta\,. \end{aligned}$$ □

3.3.3 What Can Be Achieved with One Price?

The previous result on *two-price* cost-sharing forms gives rise to the question what can be achieved using *only one* price. In the following, we say a symmetric cost-sharing method ξ is *β-uniform* with regard to costs c if for every cardinality $p \in [n]$ and every rank $\lambda \in [p]$ it holds that $\xi_\lambda(p) := \beta \cdot \min_{r \in [p]}\{\frac{c(r)}{r}\}$. Clearly, the definition ensures that ξ is cross-monotonic. Note that if c is submodular, then $\frac{c(r)}{r}$ is non-increasing in r, and the 1-uniform cost-sharing method is also 1-BB. Moreover, it coincides with the Shapley value [71] and the egalitarian solution [22].

Lemma 3.3.9. *For every symmetric, subadditive, and non-decreasing cost function c, the 2-uniform cost-sharing method ξ is cross-monotonic and 2-BB.*

Proof. Let $\mu : [n] \to \mathbb{R}_{\geq 0}$ so that $\mu(p)$ is the unique cost share for cardinality p. Fix now an arbitrary $p \in [n]$. The upper bound $p \cdot \mu(p) \leq 2 \cdot c(p)$ holds by definition. Now, since for all cardinalities $r \leq \frac{n}{2}$ we have $\frac{c(2r)}{2r} \leq \frac{2 \cdot c(r)}{2r} = \frac{c(r)}{r}$ due to subadditivity, we get that $\mu(p) = 2 \cdot \frac{c(r)}{r}$ for some $r \in \{\lceil \frac{p}{2} \rceil \dots p\}$. Furthermore, since $2 \cdot c(\lceil \frac{p}{2} \rceil) \geq c(p)$, we have that $p \cdot \mu(p) = p \cdot 2 \cdot \frac{c(r)}{r} \geq 2 \cdot c(\lceil \frac{p}{2} \rceil) \geq c(p)$, which proves the lower bound. □

In the following, we show that 2-BB is in fact a lower bound for all GSP “one-price” cost-sharing mechanisms.

Lemma 3.3.10. *For any $\varepsilon > 0$, there is a symmetric, subadditive, and non-decreasing cost function for which no GSP and $(2-\varepsilon)$-BB mechanism exists that always assigns all served players the same price.*

Proof. Let $\varepsilon \in (0,1]$ and $n \in \mathbb{N}$ with $n > \frac{2}{\varepsilon}$. Consider the cost function $c(p) := 1$ for all $p \in [n-1]$ and $c(n) := 2$. By way of contradiction, assume there is a GSP mechanism $M = (Q, x)$ that is $(2-\varepsilon)$-BB and charges all players equally. Since M is GSP, it is separable (see Moulin [51] or Theorem 4.2.9). Denote the equal cost share used for player set S by $\mu(S)$.

We first observe that for each $i \in [n] : \mu([n] \setminus i) < \mu([n])$. Otherwise, there is a player i so that $\mu([n] \setminus i) \geq \frac{c(n)}{n} = \frac{2}{n}$ and then $(n-1) \cdot \mu([n] \setminus i) \geq 2 - \frac{2}{n} > 2 - \varepsilon$, which is a contradiction to $(2-\varepsilon)$-BB.

Now assume the true valuations $\boldsymbol{v}$ are given by $v_1 = v_2 = \mu([n])$ and $v_i := b^\infty$ for all other players i. It has to hold that $Q(\boldsymbol{v}_{-1}, b^\infty) = [n] \setminus 2$. Otherwise, if $Q(\boldsymbol{v}_{-1}, b^\infty) = [n]$, player 2 could help all other players by bidding -1. Correspondingly, $Q(\boldsymbol{v}_{-2}, b^\infty) = [n] \setminus 1$.

Now there are four possibilities for $Q(\boldsymbol{v})$: If $Q(\boldsymbol{v}) = [n]$, $Q(\boldsymbol{v}) = [n] \setminus 1$, or $Q(\boldsymbol{v}) = [n] \setminus \{1,2\}$, then player 1 could improve by bidding b^∞. Correspondingly, if $Q(\boldsymbol{v}) = [n] \setminus 2$, then player 2 could improve by bidding b^∞. A contradiction to GSP. □

3.4 Applications

In this section, we consider the problem of sharing the *makespan cost* when scheduling n jobs on m parallel machines, where there is a one-to-one correspondence between players and jobs. Recall that the cost $C(S)$ is defined as the maximum completion time in an optimal schedule for the jobs in S (see Section 2.3.1).

It is a simple observation that C is a subadditive function. Moreover, if jobs are identical, then C is symmetric and $C(S)$ can be computed in time $O(n \cdot \log m)$ using the LPT (longest processing time first) algorithm [27].

If jobs are not identical, C is not symmetric anymore and an optimal schedule is in general NP-hard to compute. However, to keep finding a solution computationally tractable, we want algorithms to be polynomial-time computable in the size of the scheduling instance plus the players' bids. We therefore need to resort to approximation algorithms. For any algorithm ALG that computes feasible schedules, we define $C_{\text{ALG}}(S)$ as the makespan of the schedule that ALG computes for the jobs in S. The objective is now to find mechanisms that are β-BB with regard to the actual cost C_{ALG} and the optimal cost C.

3.4.1 Makespan Minimization with Identical Jobs

Theorem 3.4.1. *For sharing the makespan cost when scheduling n identical jobs on m related machines ($Q|p_i = p|C_{\max}$), there is always a valid $\frac{\sqrt{17}+1}{4}$-BB 2P-CSF. The outcome of the corresponding symmetric mechanism, together with a schedule for the served players, can be computed in time $O(n^2 + n \cdot \log m)$.*

Proof. Since the optimal costs are symmetric, subadditive, and non-decreasing, it is sufficient to apply Theorem 3.3.3. Consider now the running time: $c(1), \ldots, c(n)$ can be computed by a single run of LPT, which takes time $O(n \cdot \log m)$. Afterwards, computing the corresponding 2P-CSF with Algorithm 3.2 can be done in time $O(n)$. Finally, computing the outcome of the symmetric mechanism with Algorithm 3.1 takes time $O(n^2)$. Summing up yields the desired result. $\square$

Lemma 3.4.2. *For sharing the makespan cost when scheduling n identical jobs on m identical machines ($P|p_i = p|C_{\max}$), there is always a 1-BB 2P-CSF that can be computed in time $O(n)$.*

Proof. Define the 2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ as follows: For cardinality $p \in [n]$, define $\sigma \in \mathbb{N}, \tau \in [m-1]_0$ by $p = \sigma \cdot m + \tau$.

- If $p \in [m]$, let $d_p := 0$, $h_p := \infty$, and $l_p := \frac{1}{p}$.
- If $p > m$ and $\tau = 0$, let $d_p := 0$, $h_p := \infty$, and $l_p := \frac{1}{m}$.
- Otherwise, let $d_p := 1$, $h_p := 1 - \frac{\tau}{m}$, and $l_p := \frac{1}{m}$.

It can easily be verified that this 2P-CSF is valid and 1-BB. We remark that the same 2P-CSF can be obtained by applying Algorithm 3.2 and dividing all cost shares by $\frac{\sqrt{17}+1}{4}$. $\square$

Lemma 3.4.3. *There is a family of scheduling instances with n identical jobs and m identical machines, so that any symmetric mechanism driven by the 2P-CSF computed by Algorithm 3.2 (or as in Lemma 3.4.2) cannot guarantee an economic efficiency better than $\Theta(n)$.*

Proof. Let $n = 2m$. Hence, the cost function C is specified by $c(p) = 1$ for cardinalities $p \in [m]$ and $c(p) = 2$ for $p \in \{m+1 \dots 2m\}$. Let $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ as in Lemma 3.4.2 (as noted before, this is the same 2P-CSF as the one computed by Algorithm 3.2 but divided by $\frac{\sqrt{17}+1}{4}$). We have:

p	1	2	$\dots$	m	$m+1$	$m+2$	$\dots$	$2m$
$c(p)$	1	1	$\dots$	1	2	2	$\dots$	2
$\xi(p)$	1	$\frac{1}{2}, \frac{1}{2}$	$\dots$	$\frac{1}{m}, \dots$	$1, \frac{1}{m}, \dots$	$\frac{m-1}{m}, \frac{1}{m}, \dots$	$\dots$	$\frac{1}{m}, \dots$

Suppose the true valuations are $\boldsymbol{v} = (\frac{1}{m} - \varepsilon, \frac{2}{m} - \varepsilon, \dots, 1 - \varepsilon, \frac{1}{m} + \varepsilon, \dots, \frac{1}{m} + \varepsilon)$ for some fixed $\varepsilon > 0$. Let $M = (Q, x)$ be the symmetric mechanism defined by $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$. Then $Q(\boldsymbol{v}) = \{m+1 \dots m\}$, $C(Q(\boldsymbol{v})) = 1$, and $\sum_{i=1}^{m} v_i = \frac{m+1}{2} - m\varepsilon$, hence $C(Q(\boldsymbol{v})) + \sum_{i \notin Q(\boldsymbol{v})} v_i = \Theta(n)$. However $C([n]) = 2$. This completes the proof. $\square$

It is turns out that there are "one-price" Moulin mechanisms that offer a better guarantee for economic efficiency. As a corollary of Lemma 3.3.9 and the subsequent Lemma 3.4.7 we get:

Lemma 3.4.4. *Let c be a symmetric, subadditive, and non-decreasing cost function. Then, the Moulin mechanism induced by the 2-uniform cost-sharing method is 2-BB and $2H_n$-EFF.*

For the proof, we need a result by Roughgarden and Sundararajan [64] that relates the economic efficiency of Moulin mechanisms to a property of their cost-sharing methods.

Definition 3.4.5 (Roughgarden and Sundararajan [64]). *A cost-sharing method ξ is α-*summable *with regard to costs C if for all sets $A \subseteq [n]$ and all orders $a_1, \dots, a_{|A|}$ of A it holds that $\sum_{p=1}^{|A|} \xi_{a_p}(\{a_1, \dots, a_p\}) \le \alpha \cdot C(A)$.*

Proposition 3.4.6 (Roughgarden and Sundararajan [64]). *Let ξ be a cross-monotonic cost-sharing method. Suppose ξ is β-BB with regard to approximate costs C' and optimal costs C, and moreover α-summable with regard to C. Then, the Moulin mechanism induced by ξ is α-EFF.*

We remark here that in the conference version by Roughgarden and Sundararajan [64], the definition of budget balance is slightly different to our Definition 2.1.10. This leads to a different economic-efficiency guarantee in Proposition 3.4.6. We therefore refer to the extended version in which the same definition as in this thesis is used.

Lemma 3.4.7. *Let c be a symmetric, subadditive, and non-decreasing cost function. Then the 2-uniform cost-sharing method is $O(\log n)$-summable.*

Proof. Let $A \subseteq [n]$ be a set of players and $a_1, \dots, a_{|A|}$ an arbitrary order of A. Then

$$\sum_{p=1}^{|A|} \xi_{a_p}(\{a_1, \dots, a_p\}) \le 2 \cdot \sum_{p=1}^{|A|} \frac{c(p)}{p} \le 2H_{|A|} \cdot c(|A|) \le 2H_n \cdot c(|A|). \qquad \square$$

3.4.2 Makespan Minimization with Non-Identical Jobs

Theorem 3.4.8. *For sharing the makespan cost when scheduling n (not necessarily identical) jobs on m related machines ($Q||C_{\max}$), there is always a $(d \cdot \frac{\sqrt{17}+1}{4})$-BB cost-sharing mechanism. Here, d is the number of different processing requirements. The outcome of the cost-sharing mechanism can be computed in time $O(n^2 + n \cdot \log m)$.*

Proof. Let $\boldsymbol{p} \in \mathbb{N}^n$ be the vector with the processing requirements and let $\boldsymbol{s} \in \mathbb{N}^m$ contain the machine speeds. Let $P := \bigcup_{i \in [n]} \{p_i\}$ be the set of *different* processing requirements and define $d := |P|$. Moreover, for $\phi \in P$, define $N_\phi := \{i \in [n] \mid p_i = \phi\}$ as the set of all players whose job has processing requirement ϕ.

i) We denote by $c : [n] \to \mathbb{R}_{\ge 0}$ the optimal-makespan cost function if all jobs were identical with processing requirement 1. Compute $c(1), \dots, c(n)$ with a single run of LPT. This takes time $O(n \cdot \log m)$.

ii) Use Algorithm 3.2 to compute the 2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ that corresponds to costs c. This can be done in time $O(n)$.

iii) For each processing requirement $\phi \in P$, do the following: Use $(|N_\phi|, \boldsymbol{d}, \phi \cdot \boldsymbol{h}, \phi \cdot \boldsymbol{l})$ and only the bids of the players in N_ϕ as input for Algorithm 3.1. This yields a set of served players $Q_\phi \subseteq N_\phi$ and a cost distribution $\boldsymbol{x}^\phi$. Moreover, use LPT to compute a schedule $\boldsymbol{a}^\phi$ for the players in Q_ϕ. The time needed for each processing requirement ϕ is $O(|N_\phi|^2)$ for Algorithm 3.1 and $O(|N_\phi| \cdot \log m)$ for LPT. Altogether, this step hence can be computed in time $O(n^2 + n \cdot \log m)$.

iv) Let the set of served player be $Q := \bigcup_{\phi \in P} Q_\phi$. Assign each player i the cost share $x_i := x_i^{p_i}$ that was computed for processing requirement p_i. Finally, merge all schedules $\boldsymbol{a}^\phi$. This takes time $O(n)$.

Hence, the total running time is $O(n^2 + n \cdot \log m)$. It remains to be shown that this mechanism is $(d \cdot \frac{\sqrt{17}+1}{4})$-BB with respect to the actual cost $C'(Q)$, which is the makespan resulting from merging all schedules $\boldsymbol{a}^\phi$, and the optimal cost $C(Q)$:

$$\begin{aligned}
C'(Q) &\le \sum_{\phi \in P} \phi \cdot c(|Q_\phi|) && \text{because merging is subadditive} \\
&\le \sum_{\phi \in P} \sum_{i \in Q_\phi} x_i^\phi && \text{by Theorem 3.4.1} \\
&\le \sum_{\phi \in P} \frac{\sqrt{17}+1}{4} \cdot C(Q_\phi) && \text{by Theorem 3.4.1} \\
&\le d \cdot \frac{\sqrt{17}+1}{4} \cdot C(Q). && \square
\end{aligned}$$

3.5 Characterizing Symmetry and 1-BB

3.5.1 An Impossibility Result

Theorem 3.5.1. *There is no GSP mechanism that is 1-BB with regard to the symmetric 4-player cost function C specified as follows:*

p	1	2	3	4
$c(p)$	1	3	6	7

For the proof of Theorem 3.5.1, we need the subsequent Propositions 3.5.2, 3.5.4, and 3.5.5, together with Lemma 3.5.6.

Proposition 3.5.2 (Moulin [51]). *Let $M = (Q, x)$ be a GSP cost-sharing mechanism and ξ its cost-sharing method. Then, for all $S \subseteq [n]$, $i, j \notin [n] \setminus S$, $i \neq j$, at least one of the following three conditions holds:*

i) $\xi_i(S \cup \{i,j\}) < \xi_i(S \cup \{i\})$ *and* $\xi_j(S \cup \{i,j\}) < \xi_j(S \cup \{j\})$,

ii) $\xi_i(S \cup \{i,j\}) = \xi_i(S \cup \{i\})$,

iii) $\xi_j(S \cup \{i,j\}) = \xi_j(S \cup \{j\})$.

Note here that Proposition 3.5.2 implies the following simple observation: Suppose ξ is the cost-sharing method of a GSP cost-sharing mechanism, $S \subsetneq [n]$, $j,k \in [n] \setminus S$, and $j \neq k$. Then the following implication holds:

$$\xi_j(S \cup \{j,k\}) > \xi_j(S \cup \{j\}) \Longrightarrow \xi_k(S \cup \{k\}) = \xi_k(S \cup \{j,k\})\,. \tag{3.5.3}$$

Proposition 3.5.4 (Immorlica et al. [36]). *Let $M = (Q,x)$ be a GSP cost-sharing mechanism and ξ its cost-sharing method. Then, if every player i bids $b_i > \xi_i([n])$ it holds that $Q(\boldsymbol{b}) = [n]$.*

Proposition 3.5.5 (Immorlica et al. [36]). *Let $M = (Q,x)$ be a GSP cost-sharing mechanism and ξ its cost-sharing method. Then, for all $S \subseteq [n]$ and $i \in [n]$, it holds that $\forall j \in S : \xi_j(S) \leq \xi_j(S \cup \{i\})$ or $\forall j \in S : \xi_j(S) \geq \xi_j(S \cup \{i\})$.*

Lemma 3.5.6. *Let the symmetric 3-player cost function C be specified as follows:*

p	1	2	3
$c(p)$	1	3	6

Then, every 1-BB and GSP cost-sharing mechanism has one of the following cost-sharing methods μ, ν (up to renumbering of the players).

U	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
$\mu(U)$	$(0,0,0)$	$(1,0,0)$	$(0,1,0)$	$(0,0,1)$	$(2,1,0)$	$(2,0,1)$	$(0,2,1)$	$(3,2,1)$
$\nu(U)$	$(0,0,0)$	$(1,0,0)$	$(0,1,0)$	$(0,0,1)$	$(2,1,0)$	$(2,0,1)$	$(0,2,1)$	$(2,3,1)$

Proof. Let M be a 1-BB GSP cost-sharing mechanism and ξ its cost-sharing method. We will show that $\xi \in \{\mu, \nu\}$. Note that in order to increase readability, we omit parentheses in this proof when it is unambiguous to do so.

Since $C\{j,k\} > C\{j\} + C\{k\}$ for any $j,k \in [3]$, $j \neq k$, it follows that $\xi_j\{j,k\} > \xi_j\{j\} = C\{j\}$ or $\xi_k\{j,k\} > \xi_k\{k\} = C\{k\}$. Then, by (3.5.3), we have $\xi_j\{j,k\} = C\{j\}$ or $\xi_k\{j,k\} = C\{k\}$.

Let $\triangleleft$ denote the binary relation $\triangleleft := \{(a,b) \in [3]^2 \mid \xi_a\{a,b\} \geq \xi_b\{a,b\}\}$. W.l.o.g., there are two cases.

i) Case $\xi_3\{2,3\} = 1$, $\xi_2\{1,2\} = 1$, and $\xi_1\{1,3\} = 1$ ($\triangleleft$ is cyclic order):

   Again, w.l.o.g., we may assume that $\xi_1\{1,2,3\} > 1 = \xi_1\{1,3\}$. It follows by (3.5.3) that $\xi_2\{1,2,3\} = \xi_2\{2,3\} = 3 - 1 = 2$. Then, $\xi_2\{1,2,3\} > 1 = \xi_2\{1,2\}$

and hence, by the same argument, $\xi_3\{1,2,3\} = \xi_3\{1,3\} = 3-1 = 2$. Now 1-BB implies $\xi_1\{1,2,3\} = 6-2-2 = 2$, i.e., $\xi\{1,2,3\} = (2,2,2)$.

Now assume that the true valuation vector is $\boldsymbol{v} = (2,2,2)$. Since $Q(\boldsymbol{v}) = \{1,2,3\}$ would make all players indifferent, a GSP mechanism would, w.l.o.g., select player 1 as the only player having strictly positive utility, i.e., $Q(\boldsymbol{v}) \in \{\{1\},\{1,2\}\}$. Then players 2 and 3 could cooperate to help player 3 because with the same argument as before it has to hold that $Q(2, b^\infty, b^\infty) = \{2,3\}$. A contradiction to GSP.

ii) Case $\xi_3\{2,3\} = 1$, $\xi_3\{1,3\} = 1$, and $\xi_2\{1,2\} = 1$ ($\lhd$ is linear order):

Assume first that $\xi_3\{1,2,3\} > 1$. By (3.5.3) we get that $\xi_2\{1,2,3\} = \xi_2\{1,2\} = 1$ and $\xi_1\{1,2,3\} = \xi_1\{1,2\} = 3-1 = 2$, hence $\xi_3\{1,2,3\} = 6-2-1 = 3$. Suppose now the true valuations are $\boldsymbol{v} = (2, b^\infty, b^\infty)$. Then player 1 is indifferent to getting the service and he could help player 3 by bidding -1 or help player 2 by bidding b^∞, a contradiction.

Assume now that $\xi_3\{1,2,3\} < 1$. Since $\xi_2\{1,2,3\} > 1 = \xi_2\{1,2\}$ would imply $\xi_3\{1,2,3\} = 1$ (again, by (3.5.3)), we have that $\xi_2\{1,2,3\} \leq 1$ in this case. Now consider the true valuation vector $\boldsymbol{v} = (b^\infty, \xi_2\{1,2,3\}, v_3)$ where $v_3 \in (\xi_3\{1,2,3\}, 1)$. Here, player 2 is obviously indifferent about getting the service and as a result he could either help player 3 by bidding b^∞ (using Proposition 3.5.4) or player 1 by bidding -1 (since $\xi_1\{1,2,3\} \geq 6-1-1 > 1 = \xi_1\{1\}$).

Hence, $\xi_3\{1,2,3\} = 1$. Assume next that $\xi_1\{1,2,3\} > 3$. Then, $\xi_2\{1,2,3\} < 2$. Moreover, when $\boldsymbol{v} = (b^\infty, v_2, b^\infty)$ with $v_2 \in (\xi_2\{1,2,3\}, 2)$, all players are served for $\boldsymbol{v}$ (Proposition 3.5.4). However, player 1 would be better off by bidding some $b_1 \in (2, \xi_1\{1,2,3\})$ instead of b^∞, in which case only players 1 and 3 would be served. A contradiction to SP.

Now if $\xi_1\{1,2,3\} < 3$, then $\xi_2\{1,2,3\} > 2 = \xi_2\{2,3\}$ and therefore, by (3.5.3), $\xi_1\{1,2,3\} = \xi_1\{1,3\} = 2$, i.e., $\xi\{1,2,3\} = (2,3,1)$.

Hence, we have shown that $\xi\{1,2,3\} \in \{(3,2,1),(2,3,1)\}$, which completes the proof.

Proof (of Theorem 3.5.1). The proof is by contradiction and consists of several steps. Assume that there is a 1-BB and GSP cost-sharing mechanism. Let ξ be its cost-sharing method. Note first that since players may opt to not participate (by submitting a negative bid), the results of Lemma 3.5.6 hold for all subset $U \subset [4]$ with $|U| = 3$.

i) ξ induces a unique order on the set of players.

Let $\lhd$ denote the binary relation $\lhd := \{(a,b) \mid \xi_a\{a,b\} \geq \xi_b\{a,b\}\} \subseteq [4]^2$. By Lemma 3.5.6, for any 3-element-subset $U \subset [4]$, the restriction of $\lhd$ to U is a linear order on U.

Now assume $\lhd$ is not a linear order on the whole of $[4]$. Since reflexivity and antisymmetry are obviously fulfilled, this means that there are $a,b,c \in [4]$ with $a \neq b$, $a \neq c$, $b \neq c$, $a \lhd b$, $b \lhd c$, and $c \lhd a$. Clearly, this is a contradiction for the 3-element-set $U = \{a,b,c\}$.

Hence, $\lhd$ is a linear order, and we may, w.l.o.g., assume $1 \lhd 2 \lhd 3 \lhd 4$ in the following.

ii) Finally, we will show that GSP and 1-BB lead to a contradiction.

By Lemma 3.5.6, for each 3-element subset $U \subset [4]$, there are only two possible vectors of cost shares, e.g., $\xi\{1,2,3\} \in \{(3,2,1,0),(2,3,1,0)\}$.

Assume first $\xi_4\{1,2,3,4\} > 1$. Then, since $1 = \xi_4(\{1,2,3,4\} \setminus i)$ for all $i \in \{1,2,3\}$ it follows by (3.5.3) that $\xi_i\{1,2,3,4\} = \xi_i\{1,2,3\}$, a contradiction to 1-BB. Similarly, if $\xi_4\{1,2,3,4\} < 1$, then (3.5.3) implies for all $i \in \{1,2,3\}$ that $\xi_i\{1,2,3,4\} \leq \xi_i\{1,2,3\}$. Again a contradiction to 1-BB. Consequently, $\xi_4\{1,2,3,4\} = 1$.

Because of Proposition 3.5.5 it holds that either $\xi_{[3]}\{1,2,3,4\} \geq \xi_{[3]}\{1,2,3\}$ or $\xi_{[3]}\{1,2,3,4\} \leq \xi_{[3]}\{1,2,3\}$. Consequently, 1-BB implies $\xi_{[3]}\{1,2,3,4\} = \xi_{[3]}\{1,2,3\}$.

Hence, we only need to consider the following two cases:

a) Case $\xi\{1,2,3\} = (3,2,1,0)$, i.e., $\xi\{1,2,3,4\} = (3,2,1,1)$:

If $\xi\{1,3,4\} = (3,0,2,1)$ then player 2 can help either player 3 or 1 in case that the true valuation vector is $\boldsymbol{v} = (b^\infty, 2, \frac{3}{2}, b^\infty)$: He could bid b^∞ or -1 because $Q(\boldsymbol{v}_{-2}, b^\infty) = \{1,2,3,4\}$ and $Q(\boldsymbol{v}_{-2}, -1) = \{1,4\}$.

Also, if $\xi\{1,3,4\} = (2,0,3,1)$, player 2 can again help either player 3 or 1 in case that the true valuation vector is $\boldsymbol{v} = (b^\infty, 2, b^\infty, b^\infty)$, by bidding b^∞ or -1.

b) Case $\xi\{1,2,3\} = (2,3,1,0)$, i.e., $\xi\{1,2,3,4\} = (2,3,1,1)$:

We can use essentially the same arguments as in the previous case: If $\xi\{2,3,4\} = (0,3,2,1)$ then player 1 can help either player 3 or 2 in case that the true valuation vector is $\boldsymbol{v} = (2, b^\infty, \frac{3}{2}, b^\infty)$: He could bid b^∞ or -1 because $Q(\boldsymbol{v}_{-1}, b^\infty) = \{1,2,3,4\}$ and $Q(\boldsymbol{v}_{-1}, -1) = \{2,4\}$.

Also, if $\xi\{2,3,4\} = (0,2,3,1)$, player 1 can again help either player 3 or 2 in case that the true valuation vector is $\boldsymbol{v} = (2, b^\infty, b^\infty, b^\infty)$, by bidding b^∞ or -1.

Hence, all cases are in contradiction to GSP. Consequently, there is no mechanism that satisfies GSP and 1-BB. □

3.5.2 GSP and 1-BB Cost-Sharing Mechanisms for Three Players

Theorem 3.5.7. *If the number of players is at most 3, then for every symmetric costs C there is a 1-BB and GSP mechanism.*

Proof. Define the symmetric cost-sharing method ξ as follows:

p	$\xi_1(p), \dots, \xi_p(p)$	condition
1	$c(1)$	
2	$\frac{c(2)}{2}, \frac{c(2)}{2}$	if $\frac{c(2)}{2} \leq \xi_1(1) = c(1)$
	$c(2) - c(1), \xi_1(1)$	otherwise
3	$\frac{c(3)}{3}, \frac{c(3)}{3}, \frac{c(3)}{3}$	if $\frac{c(3)}{3} \leq \xi_2(2)$
	$c(3) - 2 \cdot \xi_2(2), \xi_2(2), \xi_2(2)$	otherwise, if $c(3) - c(2) < \xi_2(2)$
	$\xi_1(2), c(3) - c(2), \xi_2(2)$	otherwise, if $c(3) - c(2) < \xi_1(2)$
	$c(3) - c(2), \xi_1(2), \xi_2(2)$	otherwise

If it does not holds that $c(3) - c(2) > c(2) - c(1) > c(1)$, it can easily be verified that ξ is a valid symmetric cost-sharing method. Hence, due to Theorem 3.2.8, the symmetric mechanism defined by ξ is GSP.

Otherwise, if $c(3) - c(2) > c(2) - c(1) > c(1)$, the cost function c is supermodular. The following mechanism that shares cost incrementally is GSP: Start with the empty player set Q and do the following iteratively, for every player $i = 3, 2, 1$: If i bids strictly more than his marginal cost $C(Q \cup \{i\}) - C(Q)$, then charge him his marginal cost and add him to Q. Otherwise, he will not be served.

Now a player can only improve if the number of players added before him decreases. However, this would require a coalition where some member loses utility. □

We close by remarking that the mechanism proposed for the case $c(3) - c(2) > c(2) - c(1) > c(1)$ is called a *sequential stand-alone mechanism* (see also Section 4.4.2). Somewhat counterintuitively, if costs are supermodular but not necessarily symmetric, then sequential stand-alone mechanisms are in general not GSP [51, p. 297, second remark after Proposition 1].

3.6 Beyond Symmetric Costs

In this section, we outline a possible direction for generalizing precedence mechanisms to arbitrary costs. In detail, we formulate conditions for non-symmetric cost-sharing methods so that the resulting precedence mechanisms are WGSP. Afterwards, we show what is the "missing link" to GSP. Note that in this section, we take the lexicographic order with the usual significance. For instance, $(1, 2) \prec (2, 1)$.

We start with a simple observation that may be of independent interest:

Lemma 3.6.1. *Consider an arbitrary precedence mechanism. Let $\boldsymbol{v}$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$. Suppose $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$ for some $i \in [n]$. Then, there is a player $j < i$ with $u_j(\boldsymbol{b}) < u_j(\boldsymbol{v})$. Consequently, $P(\boldsymbol{v}, Q(\boldsymbol{b})) \prec P(\boldsymbol{v}, Q(\boldsymbol{v}))$.*

Proof. W.l.o.g., i is the first improving player. By way of contradiction, assume $u_j(\boldsymbol{b}) = u_j(\boldsymbol{v})$ for all $j < i$. By definition of precedence mechanisms, it must hold that there is a player $k > i$ with $x_k(\boldsymbol{b}) > v_k$. Otherwise, $P(\boldsymbol{v}, Q(\boldsymbol{b}))$ would be non-negative and $P(\boldsymbol{v}, Q(\boldsymbol{b})) \succ P(\boldsymbol{v}, Q(\boldsymbol{v}))$. Due to VP, it now follows that also $b_k > v_k$ and hence $k \in K$. This contradicts the assumption that K is successful. □

3.6.1 Precedence-Monotonic Cost Shares

Definition 3.6.2. *A cost-sharing method ξ is* precedence monotonic *if for all $S, T \subseteq [n]$ and $i \in S$ with $\xi_i(T) > \xi_i(S)$ or $i \notin T$ there is a set R with $(S \cap T \cap [i]) \cup i \subseteq R \subseteq S \cup T$ so that $\forall k \in S : \xi_k(R) \leq \xi_k(S)$ and $\forall k \in T : \xi_k(R) \leq \xi_k(T)$.*

Theorem 3.6.3. *Precedence mechanisms with a precedence monotonic cost-sharing method satisfy NPT, VP, and CS.*

Proof. NPT is fulfilled because cost-sharing methods are non-negative. VP is fulfilled since the mechanism only chooses feasible sets. In order to show CS, consider arbitrary valuations $\boldsymbol{v}$ and a player i with $i \notin Q(\boldsymbol{v}) =: S$. Let $T := S \cup i$, $b_i > \max_{A \subseteq [n] | i \in A} \xi_i(A)$, and $\boldsymbol{b} := (\boldsymbol{v}_{-i}, b_i)$.

By Definition 3.6.2, there is a set R (for S, T, and i) with $(S \cap [i]) \cup i \subseteq R \subseteq S \cup i$ and $\forall k \in S : \xi_k(R) \leq \xi_k(S)$ and $\forall k \in T : \xi_k(R) \leq \xi_k(T)$. Hence, $P(\boldsymbol{b}, R) \succ P(\boldsymbol{b}, S)$ and R is $\boldsymbol{b}$-feasible. Now note that $S \in \operatorname{lex\,max}_{A \subseteq [n] | i \notin A} \{P(\boldsymbol{b}, A)\}$. Consequently, since $P(\boldsymbol{b}, Q(\boldsymbol{b})) \succeq P(\boldsymbol{b}, R) \succ P(\boldsymbol{b}, S)$, it must hold that $i \in Q(\boldsymbol{b})$. □

Theorem 3.6.4. *Let M be a precedence mechanism with precedence monotonic cost shares. Then M is WGSP.*

Proof. The proof is by way of contradiction: Let $\boldsymbol{v}$ contain the true valuations, $K \subseteq [n]$ be a non-empty coalition, and $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \gg u_K(\boldsymbol{v})$. Define $S := Q(\boldsymbol{v})$ and $T := Q(\boldsymbol{b})$. Clearly, this implies $T \supseteq K$. By Lemma 3.6.1, there is a player $i < \min K$ with $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$, so $i \in S$ and $v_i > x_i(\boldsymbol{v}) = \xi_i(S)$. W.l.o.g., assume that i is the first such player. Note that, again by Lemma 3.6.1, this implies $u_j(\boldsymbol{b}) = u_j(\boldsymbol{v})$ for all $j < i$. By Definition 3.6.2, there is a set R (for S, T, and i) with $(S \cap T \cap [i]) \cup i \subseteq R \subseteq S \cup T$ and $\forall k \in S : \xi_k(R) \leq \xi_k(S)$ and $\forall k \in T : \xi_k(R) \leq \xi_k(T)$.

For every $k \in (T \setminus S) \cap [i]$, we have that $u_k(\boldsymbol{b}) = u_k(\boldsymbol{v}) = 0$ and $b_k = v_k$, so $u_k(\boldsymbol{b} \mid b_k) = 0$. Moreover, for all $k \in (T \cap S) \cap [i]$, it holds that $k \in R$ and $\xi_k(R) \leq \xi_k(T)$. Since $\xi_i(R) \leq \xi_i(S)$ and either $\xi_i(S) < \xi_j(T)$ or $i \notin T$, it holds that $P(\boldsymbol{b}, R) \succ P(\boldsymbol{b}, T)$. Finally, R is $\boldsymbol{b}$-feasible because for all $k \in R \cap T$, it holds that $\xi_k(R) \leq \xi_k(T) \leq b_k$, and for all $k \in R \setminus T$, it holds that $k \in S$ and $k \notin K$, so $\xi_k(R) \leq \xi_k(S) \leq v_k = b_k$. This is a contradiction. □

Example 3.6.5. *Any cross-monotonic cost-sharing method is trivially precedence monotonic, because for all $S, T \subseteq [n]$, the set $R := S \cup T$ satisfies the desired properties of Definition 3.6.2.*

Example 3.6.6. *Suppose C is a supermodular cost function. Then, a cost-sharing method that charges every player his marginal cost is precedence monotonic because for all $S, T \subseteq [n]$, the set $R := (S \cap T \cap [i]) \cup i$ satisfies the desired properties of Definition 3.6.2.*

3.6.2 The Missing Link to GSP

We show that the missing link to GSP is only a very weak version of WUNB: It must be guaranteed that if an indifferent player i changes his bid to either -1 or b^∞, then no other player's utility must improve. We first need several simple properties of precedence mechanisms.

Lemma 3.6.7. *Consider an arbitrary precedence mechanism. Let $\boldsymbol{v}$ contain the true valuations, let i be an arbitrary player, and let $\boldsymbol{b}$ be an i-variant of $\boldsymbol{v}$.*

i) Suppose $i \notin Q(\boldsymbol{v})$ and $b_i < v_i$. Then $i \notin Q(\boldsymbol{b})$ and $u(\boldsymbol{v}) = u(\boldsymbol{b})$.

ii) Suppose $i \notin Q(\boldsymbol{v})$ and $i \in Q(\boldsymbol{b})$. Then $x_i(\boldsymbol{b}) \geq v_i$.

iii) Suppose $i \in Q(\boldsymbol{v})$ and $b_i > x_i(\boldsymbol{v})$. Then $i \in Q(\boldsymbol{b})$ and $u_i(\boldsymbol{b}) \leq u_i(\boldsymbol{v})$. If also $b_i \leq v_i$, then $u_i(\boldsymbol{b}) = u_i(\boldsymbol{v})$.

iv) Suppose $M_i(\boldsymbol{v}) = M_i(\boldsymbol{b})$. Then $u(\boldsymbol{b}) = u(\boldsymbol{v})$.

Proof. i) Since VP implies $x_i(\boldsymbol{b}) \leq b_i$, it holds that $u_i(\boldsymbol{b}) \geq u_i(\boldsymbol{v})$. Hence, if $u(\boldsymbol{b}) \neq u(\boldsymbol{v})$, then $P(\boldsymbol{b}, Q(\boldsymbol{v})) \succ P(\boldsymbol{b}, Q(\boldsymbol{b}))$ due to Lemma 3.6.1. A contradiction because $Q(\boldsymbol{v})$ is $\boldsymbol{b}$-feasible. Finally, $u_i(\boldsymbol{b}) = u_i(\boldsymbol{v})$ and $b_i < v_i$ imply $i \notin Q(\boldsymbol{b})$.

ii) By way of contradiction, assume $x_i(\boldsymbol{b}) < v_i$. Then $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$ and $P(\boldsymbol{b}, Q(\boldsymbol{v})) \succ P(\boldsymbol{b}, Q(\boldsymbol{b}))$ due to Lemma 3.6.1. A contradiction because $Q(\boldsymbol{v})$ is $\boldsymbol{b}$-feasible.

iii) By (ii) (and changing the roles of $\boldsymbol{v}$ and $\boldsymbol{b}$), we have $i \in Q(\boldsymbol{b})$. Now if $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$, then $P(\boldsymbol{b}, Q(\boldsymbol{b})) \prec P(\boldsymbol{b}, Q(\boldsymbol{v}))$ by Lemma 3.6.1. This is a contradiction, since $Q(\boldsymbol{v})$ is $\boldsymbol{b}$-feasible. Hence, if $b_i \leq v_i$, then $x_i(\boldsymbol{b}) \leq x_i(\boldsymbol{v}) \leq x_i(\boldsymbol{b})$.

iv) If $i \notin Q(\boldsymbol{v})$ and $i \notin Q(\boldsymbol{b})$, then the claim follows by (i). If $i \in Q(\boldsymbol{v})$ and $i \in Q(\boldsymbol{b})$ and $u(\boldsymbol{b}) \neq u(\boldsymbol{v})$, then $P(\boldsymbol{b}, Q(\boldsymbol{v})) \succ P(\boldsymbol{b}, Q(\boldsymbol{b}))$ due to Lemma 3.6.1. A contradiction because $Q(\boldsymbol{v})$ is $\boldsymbol{b}$-feasible. □

In the following, we will need Theorem 5.3.8, which is a result from Section 5.3.2.

Lemma 3.6.8. *Let M be a WGSP precedence mechanism. Suppose that for all true valuations $\boldsymbol{v}$, all players i, and all i-variants $\boldsymbol{b}$ of $\boldsymbol{v}$ with $b_i \in \{-1, b^\infty\}$ and $u_i(\boldsymbol{v}) = u_i(\boldsymbol{b}) = 0$ it holds that $\forall j \in [n] : u_j(\boldsymbol{b}) \leq u_j(\boldsymbol{v})$. Then M is GSP.*

Proof. By way of contradiction, assume that M is not GSP. Then, due to Theorem 5.3.8, it is not even 2-GSP. This means there are true valuations $\boldsymbol{v}$, a coalition $K = \{i, i'\}$, and a K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ so that, w.l.o.g., $u_{i'}(\boldsymbol{b}) > u_{i'}(\boldsymbol{v})$ and $u_i(\boldsymbol{b}) = u_i(\boldsymbol{v})$. Denote $S := Q(\boldsymbol{v})$

and $T := Q(\boldsymbol{b})$ and $\boldsymbol{b}^i := (\boldsymbol{v}_{-i}, b_i)$. By Lemma 3.6.1, there is a player $j < i'$ with $u_j(\boldsymbol{b}) < u_j(\boldsymbol{v})$, so $j \in S$ and $v_j > x_j(\boldsymbol{v}) = \xi_j(S)$.

By way of contradiction, assume that $u_i(\boldsymbol{v}) > 0$. Then $i \in S$, $i \in T$, and $b_i \geq x_i(\boldsymbol{b}) = x_i(\boldsymbol{v})$. Since M is SP, it must hold that $u_{i'}(\boldsymbol{b}^i) = u_{i'}(\boldsymbol{b}) > u_i(\boldsymbol{v})$. Otherwise, if $u_{i'}(\boldsymbol{b}^i) < u_{i'}(\boldsymbol{b})$, player i' could improve and manipulate at $\boldsymbol{b}^i$ by bidding $b_{i'}$. Similarly, if $u_{i'}(\boldsymbol{b}^i) > u_{i'}(\boldsymbol{b})$, then player i' could improve at $\boldsymbol{b}$ by bidding $v_{i'}$.

It holds that $i \notin Q(\boldsymbol{b}^i)$. Otherwise, $x_i(\boldsymbol{b}^i) = x_i(\boldsymbol{v})$ due to SP and then $u(\boldsymbol{b}^i) = u(\boldsymbol{v})$ due to Lemma 3.6.7 (iv). Hence, $b_i \leq x_i(\boldsymbol{v}) < v_i$ due to Lemma 3.6.7 (iii). Now, if $b_i < x_i(\boldsymbol{v})$, then $x_i(\boldsymbol{b}) \leq b_i < x_i(\boldsymbol{v})$, a contradiction. Therefore, $b_i = x_i(\boldsymbol{v})$.

For all $k < i'$, $k \neq i$, it must hold that $u_k(\boldsymbol{b}^i) \geq u_k(\boldsymbol{v})$. Otherwise, $P(\boldsymbol{b}^i, S) \succ P(\boldsymbol{b}^i, Q(\boldsymbol{b}^i))$, which is a contradiction because S is $\boldsymbol{b}^i$-feasible. In particular, we have now $u_j(\boldsymbol{b}) = u_j(\boldsymbol{b}^i) \geq u_j(\boldsymbol{v})$ where the equality is due to $M_{i'}(\boldsymbol{b}^i) = M_{i'}(\boldsymbol{b})$ and Lemma 3.6.7 (iv). A contradiction. Hence, it holds that $u_i(\boldsymbol{v}) = u_i(\boldsymbol{b}) = 0$.

Now, if $i \in Q(\boldsymbol{b})$, then $u(\boldsymbol{b}_{-i}, b^\infty) = u(\boldsymbol{b})$ because of the threshold property and due to Lemma 3.6.7 (iv). Similarly, if $i \notin Q(\boldsymbol{b})$, then $u(\boldsymbol{b}_{-i}, -1) = u(\boldsymbol{b})$ due to Lemma 3.6.7 (i). Hence, we may assume w.l.o.g., that $b_i \in \{-1, b^\infty\}$. This completes the proof. □

3.7 Conclusion

In this chapter, we made a systematic first step for finding GSP mechanisms that perform better than Moulin mechanisms. Furthermore, we continued the line of characterization efforts by specifically looking at symmetric costs. It came as a surprise that despite their simplicity, these costs do not necessarily allow for GSP and 1-BB mechanisms. While symmetric costs are arguably of limited practical interest, we yet transferred our techniques to the minimum makespan problem as an application and also to a setting with non-symmetric costs. Clearly, our work leaves open many issues:

- For symmetric and/or subadditive costs, we still need an exact characterization with respect to the *best* possible budget balance that GSP mechanisms can achieve.
- Section 3.6 gave directions in which to generalize precedence mechanisms. How can these ideas be used to develop polynomial-time mechanisms for combinatorial optimization problem that do not necessarily induce symmetric costs?
- Finally: Is it possible at all to design GSP mechanisms that improve both on the budget balance *and* on the economic efficiency of Moulin mechanisms?

Chapter 4

Cost Sharing Without Indifferences: To Be or Not to Be (Served)

4.1 Overview of Contribution

Models Without Indifferences In this chapter, we study cost sharing where indifferences do not occur. In detail, we discuss three alternative—yet in a precise sense related—models, together with explanations why we regard them as equally reasonable as the "standard model" where indifferences may exist:

i) First, we find it plausible that players' utilities might not be strictly quasi-linear; instead, when players have the choice between being served for their valuation and not being served at all, they would still prefer the service. For an illustrative example, one might think of auctions here: A collector would probably prefer receiving an item also if this requires spending the maximum amount of money he could afford. We call mechanisms that are GSP with respect to these modified utilities as *group-strategyproof against service-aware players*[1] (SGSP).

ii) Second, even when utilities are quasi-linear as in the standard model, we find it a credible assumption on human behavior that a player would not join a coalition that prevents further service to himself (let alone a coalition that decreases his utility). Under this behavioral assumption, the GSP requirement is unnecessarily strong and could be relaxed so to not imply indifference about *losing* the service. In detail, we say that a mechanism is *weakly group-strategyproof against service-aware players* (WSGSP) if any defection by a coalition that increases some member's utility inevitably either decreases the utility of one of its other members or prevents one of its other members from further service.

iii) Third and last, one could also assert that the case where a player's valuation equals one of the—only finitely many—prices used by the cost-sharing mechanism is a rare and negligible event (see also Juarez [39]). Hence, the argument is that a slightly changed model where indifferences cannot occur by definition is sufficient for practical applications. Specifically, we are (only) interested here whether a mechanism is *group-strategyproof on the restricted domain of valuations* that results from excluding all possible cost shares used by the mechanism.

[1] In our WINE'07 paper [8], we called this property "group-strategyproof against collectors" due to the connotations explained before. In this thesis, I chose a more general term because I think this behavior is in no way limited to collectors and auctions.

Immediately from the definition, it follows that SGSP implies WSGSP, which in turn implies GSP on the restricted domain of valuations that does not contain the cost shares used by the mechanism. Conversely, we show a canonic way to extend any mechanism that is GSP on such a restricted domain to be SGSP on the usual domain $\mathbb{R}^n$.

The main benefit of the three models without indifferences is that they allow for cost-sharing mechanisms with much improved budget balance and economic efficiency compared to mechanisms that are required to be GSP on the usual domain of valuations $\mathbb{R}^n$. Yet, only a small amount of collusion resistance is sacrificed: In particular, even WSGSP is a strictly more robust collusion-resistance property than WGSP.

Techniques for Designing SGSP Mechanisms We introduce a novel family of SGSP mechanisms by devising an iterative algorithm based on *set selection* and *price functions* σ and ρ: In each iteration, a set of players S not yet assigned a cost share is selected by σ and offered a price specified by ρ. If there are players in S who cannot afford this price, they are rejected. Otherwise, each player in S is assigned the price. We call mechanisms induced by our new algorithm *egalitarian* because the algorithmic idea is reminiscent of Dutta and Ray's algorithm for computing egalitarian solutions [22].

If σ always selects the most cost-efficient set and ρ the respective price, we prove that egalitarian mechanisms guarantee 1-BB for arbitrary costs and additionally $2H_n$-EFF for the large class of subadditive costs. In particular, our result implies for many natural optimization problems that there are SGSP (and thus WGSP) cost-sharing mechanisms that provide exact budget balance and an economic efficiency that is asymptotically optimal for truthful and (approximately) budget-balanced cost-sharing mechanisms (i.e., $O(\log n)$-EFF, recall Section 1.5.2). This great advantage over Moulin mechanism is, e.g., illustrated by the rooted Steiner tree problem: Here, (optimal) costs are subadditive, which ensures existence of a 1-BB and $O(\log n)$-EFF egalitarian mechanism. In contrast, no Moulin mechanism can be better than 2-BB [42, Theorem 7.1] and $\Omega(\log^2 n)$-EFF [65, Theorem 4.2].

Afterwards, we show that the family of acyclic mechanisms by Mehta et al. [50] is in fact also SGSP and thus notably more collusion-resistant than just WGSP. Moreover, we prove that our egalitarian mechanisms constitute a subclass of acyclic mechanisms.

Finally, we observe that SGSP and 1-BB alone is not hard to achieve. Even trivial *sequential stand-alone mechanisms* that charge all players marginal costs are SGSP and 1-BB. However, for many natural cost-sharing problems with subadditive costs, the economic efficiency of sequential stand-alone mechanisms is poor. Hence, more advanced mechanisms like our egalitarian mechanisms should be used here. On the other hand, when costs are *supermodular*, sequential stand-alone mechanisms perform very well: Using a result by Brenner and Schäfer [11] (recall Section 1.5.2), we show that they even achieve $O(1)$-EFF.

Efficient Computation We develop a framework and techniques for coping with the computational complexity of egalitarian mechanisms, especially if the underlying optimization problems are hard. Besides the use of approximation algorithms, the key idea here are "monotonic" costs $C(S)$ that must not increase when replacing a player $i \in S$ by some other player $j \notin S$ with index $j < i$. In this case, finding the most cost-efficient set only requires iterating through all possible cardinalities (and not all possible subsets any more). The main issue here is how to pair good (but possibly non-monotonic) approximation algorithms with a monotonic cost function.

Finally, we give applications that underline the power of our new approach. For sharing the makespan cost of scheduling n jobs on m parallel machines, we achieve better budget balance than all known GSP mechanisms, while maintaining $O(\log n)$-EFF (see Table 1.1). Moreover, we achieve 1-BB and $O(\log n)$-EFF for essentially all makespan scheduling problems that are optimally solvable in polynomial time. Lastly, we also obtain results for the bin packing problem and for problems with supermodular optimal cost functions. The latter includes, e.g., several scheduling problems when the objective is to minimize the sum of completion times.

4.2 Cost Sharing Without Indifferences

In this section, we formally define the three new collusion-resistance properties without indifferences. Afterwards, we will show how they are related and that, in a precise sense, there are some natural constraints under which all three models are equivalent. Recall that relaxing the collusion-resistance requirements implies *weaker* coordination capabilities (or simply less willingness to cooperate), and is done by *strengthening* the assumptions on successful coalitions.

4.2.1 Definitions

Service-Aware Players It seems plausible that utilities are not strictly quasi-linear: When players have the choice between being served for their valuation and not being served at all, they might still prefer the service. Formally, we say that players are *service-aware* if a player i prefers outcome $(Q^*, \boldsymbol{x}^*)$ over $(Q', \boldsymbol{x}')$ if and only if $q_i^* \cdot v_i - x_i^* > q_i' \cdot v_i - x_i'$ or $(q_i^* \cdot v_i - x_i = q_i' \cdot v_i - x_i'$ and $q_i^* > q_i')$. In order to more easily compare our results, we do not define a new utility function but instead incorporate service-awareness into a new variant of collusion resistance.

Definition 4.2.1. *A mechanism M is* group-strategyproof against service-aware players *(SGSP) if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all non-empty coalitions $K \subseteq [n]$ there is no K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ with $(u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$ and $q_K(\boldsymbol{b}) \geq q_K(\boldsymbol{v}))$ or $(u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$ and $q_K(\boldsymbol{b}) > q_K(\boldsymbol{v}))$.*

Limited Coalition Formation Alternatively, we propose a new notion of coalition-resistance *in between* GSP and WGSP. The main motivation is the behavioral assumption that players are not willing to sacrifice being served for no personal reward, i.e., players are not indifferent to *losing* the service. We believe that this behavior is very plausible for human beings.

Definition 4.2.2. *A mechanism M is* weakly group-strategyproof against service-aware players (WSGSP) *if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all non-empty coalitions $K \subseteq [n]$ there is no K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$ and $q_K(\boldsymbol{b}) \geq q_K(\boldsymbol{v})$.*

Restricted Valuation Domain A third model without indifferences was proposed by Juarez [39]. Here, the domain of valuations is restricted so that players' valuations are always different to the cost shares used by the mechanism. Formally, this can be done as follows: Let M be a cost-sharing mechanism with cost-sharing method ξ. For each player i, let $D_i := \mathbb{R} \setminus \bigcup_{S \subseteq [n]} \xi_i(S)$. That is, D_i contains all reals except player i's possible payments. Note that D_i is still dense in $\mathbb{R}$. Define $D := D_1 \times \cdots \times D_n$. When valuations equal to some payment are a rare and in practice a negligible event, the argument is that one could just as plausibly focus only on the *restriction* $M|_D = (Q|_D, x|_D)$, where $Q|_D : D \to 2^{[n]}$ and $x|_D : D \to \mathbb{R}^n_{\geq 0}$.

Definition 4.2.3. *A mechanism M is* group-strategyproof on the restricted domain of valuations *$D \subseteq \mathbb{R}^n$ if for all true valuations $\boldsymbol{v} \in D$ and all non-empty coalitions $K \subseteq [n]$ there is no K-variant $\boldsymbol{b} \in D$ of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$.*

4.2.2 Equivalence of Models Without Indifferences

We show that for any mechanism that is GSP on the restricted domain D there is a canonical SGSP mechanism (on $\mathbb{R}^n$). Conversely, even WSGSP (on $\mathbb{R}^n$) implies GSP on the restricted domain D. Specifically, we show that all three models are equivalent once we require upper continuity, non-bossiness, and the threshold property.

Lemma 4.2.4. *Let M be a SGSP cost-sharing mechanism. Then M is upper-continuous.*

Proof. By way of contradiction, assume that $M = (Q, x)$ is not upper-continuous. Hence, there is a true valuation vector $\boldsymbol{v}$ and a player i so that $i \notin Q(\boldsymbol{v})$ whereas for all $z > v_i$ it holds that $i \in Q(\boldsymbol{v}_{-i}, z)$. Due to the threshold property, it holds that $x(\boldsymbol{v}_{-i}, z) = v_i$ for all $z > v_i$. Hence, $\{i\}$ is a successful SGSP-coalition; a contradiction. □

Lemma 4.2.5. *Let M be a WSGSP and upper-continuous cost-sharing mechanism. Then M is (outcome) non-bossy.*

Proof. Denote $M = (Q, x)$. Let $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, let $i \in [n]$, and let $\boldsymbol{b}$ be an i-variant. As intermediate steps, we prove the following implications:

i) $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) \Longrightarrow u(\boldsymbol{b}) = u(\boldsymbol{v})$

ii) $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 1 \Longrightarrow Q(\boldsymbol{b}) = Q(\boldsymbol{v})$

iii) $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 0$ and $b_i > v_i \Longrightarrow Q(\boldsymbol{b}) = Q(\boldsymbol{v})$

iv) $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 0$ and $b_i < v_i \Longrightarrow Q(\boldsymbol{b}) = Q(\boldsymbol{v})$

Note that the threshold property implies that $M_i(\boldsymbol{b}) = M_i(\boldsymbol{v})$ is equivalent to $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v})$. Implication (i) is clearly fulfilled because of WSGSP.

To see the other implications, suppose that $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v})$ indeed holds. By way of contradiction, assume there is a player $j \neq i$ such that, w.l.o.g., $j \notin Q(\boldsymbol{v})$ and $j \in Q(\boldsymbol{b})$, i.e., $x_j(\boldsymbol{b}) = v_j$. Let $\boldsymbol{v}^*$ be a j-variant of $\boldsymbol{v}$ with $v_j < v_j^* < \theta_j(\boldsymbol{v}_{-j})$. Such a v_j^* exists due to upper continuity. The threshold property implies $M_j(\boldsymbol{v}^*) = M_j(\boldsymbol{v})$. We also define $\boldsymbol{b}^* := (\boldsymbol{b}_{-j}, v_j^*)$. Hence, $b_j^* = v_j^* > v_j = b_j$ and again the threshold property implies $M_j(\boldsymbol{b}^*) = M_j(\boldsymbol{b})$. We can now verify the remaining three implications:

ii) Case $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 1$:

   Then $u_i(\boldsymbol{b}) = u_i(\boldsymbol{v}) = u_i(\boldsymbol{v}^*)$ where the last equality is due to (i). Hence, the coalition $\{i, j\}$ can help player j at $\boldsymbol{v}^*$ by bidding as in $\boldsymbol{b}$. This is a contradiction.

iii) Case $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 0$ and $b_i > v_i$:

   Since $M_j(\boldsymbol{b}^*) = M_j(\boldsymbol{b})$ and $j \in Q(\boldsymbol{b})$, we have $M(\boldsymbol{b}^*) = M(\boldsymbol{b})$ due to implication (ii); so, in particular, $i \notin Q(\boldsymbol{b}^*)$. Since $b_i^* = b_i > v_i = v_i^*$, the threshold property implies also $i \notin Q(\boldsymbol{v}^*)$. Consequently, player i can help player j at $\boldsymbol{v}^*$ by bidding b_i; a contradiction.

iv) Case $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v}) = 0$ and $b_i < v_i$

   As in the previous case, we have $i \notin Q(\boldsymbol{b}^*)$ due to implication (ii). Moreover, $M_j(\boldsymbol{v}^*) = M_j(\boldsymbol{v})$ and $v_j^* > v_j$, so $q_i(\boldsymbol{v}^*) = q_i(\boldsymbol{v}) = 0$ due to (iii). Hence, player i can again help player j at $\boldsymbol{v}^*$ by bidding b_i; a contradiction.

Consequently, $q_i(\boldsymbol{b}) = q_i(\boldsymbol{v})$ implies $q(\boldsymbol{b}) = q(\boldsymbol{v})$. Together with implication (i), this completes the proof. □

Lemma 4.2.6. *Let M be a cost-sharing mechanism with cost-sharing method ξ, and let D be the restricted domain of valuations as in Section 4.2.1. Then, M is SGSP if and only if $M|_D$ is GSP, M is upper-continuous, M is (outcome) non-bossy, and M fulfills the threshold property.*

Proof. Sufficiency ("⇒") is straightforward: Upper continuity and outcome non-bossiness follow from Lemmata 4.2.4 and 4.2.5. Moreover, SGSP clearly implies SP. It also implies GSP on the restricted domain of valuations D: By way of contradiction, suppose there are true valuations $\boldsymbol{v} \in D$, a non-empty coalition K, and a K-variant $\boldsymbol{b} \in D$ of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$. Then, it must hold that also $q(\boldsymbol{b}) \geq q(\boldsymbol{v})$. Consequently, K is SGSP-successful, a contradiction.

In the rest of the proof, we verify necessity ("⇐"). Let $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b} \in \mathbb{R}^n$ be a K-variant of $\boldsymbol{v}$. By way of contradiction, assume that $(u_K(\boldsymbol{b}) > u_K(\boldsymbol{b})$ and $q_K(\boldsymbol{b}) \geq q_K(\boldsymbol{v}))$ or $(u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{b})$ and $q_K(\boldsymbol{b}) > q_K(\boldsymbol{v})$, i.e., K is a SGSP-successful coalition. We show that both $\boldsymbol{v}$ and $\boldsymbol{b}$ can be transformed into K-variants $\boldsymbol{v}', \boldsymbol{b}' \in D$ so that $u_i(\boldsymbol{b} \mid v_i') \geq u_i(\boldsymbol{v} \mid v_i')$ for all $i \in K$, with at least one strict inequality. We give an algorithmic argument: First, initialize $\boldsymbol{v}' := \boldsymbol{v}$

and $\boldsymbol{b}' := \boldsymbol{b}$. Then, for every player $i = 1, \dots n$, increase both v_i' and b_i' by the same amount $\varepsilon > 0$ so that still $v_i', b_i' \in D$ and neither $M_i(\boldsymbol{v}')$ nor $M_i(\boldsymbol{b}')$ changes. Such an ε exists because D_i is dense in $\mathbb{R}$ and because M is upper-continuous. Due to (outcome) non-bossiness, $M_{-i}(\boldsymbol{v}')$ and $M_{-i}(\boldsymbol{b}')$ do no change, either.

In the end, both $\boldsymbol{v}' \in D$ and $\boldsymbol{b}' \in D$. Moreover, $M(\boldsymbol{v}) = M(\boldsymbol{v}')$, $M(\boldsymbol{b}') = M(\boldsymbol{b})$, and $\boldsymbol{v}'$ and $\boldsymbol{b}'$ are K-variants. It holds for all players $i \in K$ that $u_i(\boldsymbol{b}' \mid v_i') \geq u_i(\boldsymbol{v}' \mid v_i')$. As a last step, we show that there is at least one player $i \in K$ with $u_i(\boldsymbol{b}' \mid v_i') > u_i(\boldsymbol{v}' \mid v_i')$. This is obvious if there is a player $i \in K$ with $u_i(\boldsymbol{b} \mid v_i) > u_i(\boldsymbol{v} \mid v_i)$. If this is not the case, then there is a player $i \in Q(\boldsymbol{b}) \setminus Q(\boldsymbol{v})$ with $x_i(\boldsymbol{b}) = v_i$. Now, since $v_i' > v_i$, it follows that $u_i(\boldsymbol{b}' \mid v_i') = v_i' - x_i(\boldsymbol{b}') = v_i' - x_i(\boldsymbol{b}) > 0 = u_i(\boldsymbol{v} \mid v_i')$. This completes the contradiction. □

Corollary 4.2.7. *Let M be a cost-sharing mechanism. Then M is SGSP if and only if M is WSGSP and upper-continuous.*

Proof. This follows from Lemma 4.2.5 and Lemma 4.2.6. □

Figure 4.1 gives an overview of the previous implications. In the following, we give examples showing that the converse directions do not hold true.

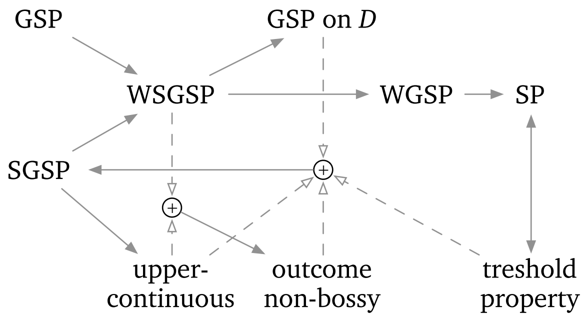


Figure 4.1: Hierarchy of collusion-resistance properties

Lemma 4.2.8. *There are cost-sharing mechanisms that are*

i) GSP but not SGSP,

ii) SGSP but not GSP, or

iii) WGSP and upper-continuous but not GSP on D (where $D \subset \mathbb{R}^n$ is the restricted domain of valuations as in Section 4.2.1).

Proof. i) Any lower-continuous GSP cost-sharing mechanism is not SGSP.

ii) Let there be $n = 2$ players and define $M = (Q, x)$ as follows: The cost-sharing method is $\xi_1(\{1\}) = \xi_2(\{2\}) = 1$ and $\xi(\{1,2\}) = (2,1)$. Moreover,

$$Q(\boldsymbol{b}) := \begin{cases} \{1,2\} & \text{if } b_1 \geq 2 \text{ and } b_2 \geq 1 \\ \{1\} & \text{if } b_1 \geq 1 \text{ and } b_2 < 1 \\ \{2\} & \text{if } b_1 < 2 \text{ and } b_2 \geq 1 \\ \emptyset & \text{otherwise.} \end{cases}$$

For an illustration, see Figure 4.2a. Note first that M is upper-continuous and satisfies the threshold property (Proposition 2.1.8). Moreover, for player 2, this threshold value is constantly $\theta_2(b_1) = 1$. The threshold value for player 1 is $\theta_1(b_2) = 2$ if player 2 gets the service (i.e., $b_2 \geq 1$) and 1 otherwise; i.e., the threshold value only changes if player 2 loses the service. This proves SGSP.

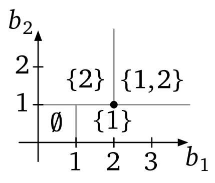
M is not GSP: Assume $\boldsymbol{v} = (2,1)$. Then player 2 could help player 1 by bidding strictly less than 1.

iii) Let there be $n = 2$ players and define $M' = (Q', x')$ as follows: The cost-sharing method is $\xi_1(\{1\}) = \xi_2(\{2\}) = 2$ and $\xi(\{1,2\}) = (3,1)$. Moreover,

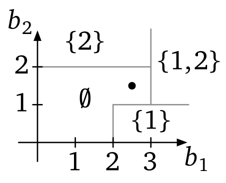
$$Q'(\boldsymbol{b}) := \begin{cases} \{1,2\} & \text{if } b_1 \geq 3 \text{ and } b_2 \geq 1 \\ \{1\} & \text{if } b_1 \geq 2 \text{ and } b_2 < 1 \\ \{2\} & \text{if } b_1 < 3 \text{ and } b_2 \geq 2 \\ \emptyset & \text{otherwise.} \end{cases}$$

For an illustration, see Figure 4.2b. Note first that M' is SP as the threshold property is fulfilled. Moreover, M' is upper-continuous. Now, M' always chooses a $\boldsymbol{b}$-feasible set that maximizes player 2's utility. Hence, both players can never form a successful WGSP-coalition together. This proves WGSP.

Suppose now $\boldsymbol{v} = (2.5, 1.5)$. Then, player 2 could help player 1 by bidding strictly less than 1, e.g., let $\boldsymbol{b} = (2.5, 0.5)$. It holds that $\boldsymbol{v}, \boldsymbol{b} \in D$, so M' is not GSP on the restricted domain D. □



(a) Graphical illustration of Q



(b) Graphical illustration of Q'

Figure 4.2: Examples showing that collusion-resistance variants are not equivalent

4.2.3 WSGSP Implies Separability

We show in the following that already WSGSP is sufficient for separability; hence, collusion resistance according to any of the three models without indifference implies existence of a cost-sharing method. A corresponding result for GSP mechanisms is due to Moulin [51]. However, our Theorem 4.2.9 is stronger, since GSP is relaxed to WSGSP, and *strong* CS is relaxed to CS. In particular, our result continues to hold if the domain of valuations is restricted to non-negative bids.

Theorem 4.2.9. *Let M be a WSGSP mechanism. Then, M is separable. This result holds even if the domain of valuations is restricted to $\mathbb{R}^n_{\geq 0}$ (i.e., even if M does not satisfy strong CS).*

Proof. Let D denote the domain of valuations. We establish the result both for $D = \mathbb{R}^n$ and $D = \mathbb{R}^n_{\geq 0}$. Define a cost-sharing method $\xi : 2^{[n]} \to \mathbb{R}^n_{\geq 0}$ as follows: Denote $\mathfrak{b}^- := -1$ if $D = \mathbb{R}^n$ and $\mathfrak{b}^- := 0$ if $D = \mathbb{R}^n_{\geq 0}$. Now, define $\widehat{b} : 2^{[n]} \to D$ by $\widehat{b}_i(S) := b^\infty$ if $i \in S$ and $\widehat{b}_i(S) := \mathfrak{b}^-$ otherwise. Then, let $\xi_i(S) := x(\widehat{b}(S))$. In order to prove the theorem, it is sufficient to show that for any true valuations $\boldsymbol{v}$ it holds that $x(\boldsymbol{v}) = \xi(Q(\boldsymbol{v}))$. We do this by induction over $m \in [n]$:

Claim (Induction Hypothesis). Suppose $\emptyset \neq S \subseteq [n]$ with $|S| \leq m$, and $\boldsymbol{b} \in \mathbb{R}^n$ is an S-variant of $\boldsymbol{v}$ such that $b_i = b^\infty$ if $i \in S \cap Q(\boldsymbol{v})$, $b_i = \mathfrak{b}^-$ if $i \in S \setminus Q(\boldsymbol{v})$, and $b_i = v_i$ otherwise. Then $u(\boldsymbol{v}) = u(\boldsymbol{b})$, and for all $i \in S$ with $v_i > 0$ or $i \in Q(\boldsymbol{v})$ it holds that $M_i(\boldsymbol{v}) = M_i(\boldsymbol{b})$.

Proof (of claim). The *base case* is $m = 1$: Suppose $S = \{i\}$ and $\boldsymbol{b}$ is as in the induction hypothesis. Due to the threshold property, $M_i(\boldsymbol{b}) = M_i(\boldsymbol{v})$. Hence, also $u(\boldsymbol{b}) = u(\boldsymbol{v})$ due to WSGSP. Otherwise, player i could help some other player either by bidding v_i if the true valuation vector was $\boldsymbol{b}$ or or b_i if it was $\boldsymbol{v}$.

For the *induction step* "$(m-1) \to m$", assume the induction hypothesis holds up to $m - 1$. Suppose $S \subseteq [n]$ with $|S| = m$ and $\boldsymbol{b}$ is as in the induction hypothesis. Define $S' := \{j \in S \mid v_j > 0 \text{ or } j \in Q(\boldsymbol{v})\}$. If $D = \mathbb{R}^n$ then consider the coalition $K := S$, otherwise if $D = \mathbb{R}^n_{\geq 0}$, let $K := S'$. In the latter case it holds for all $j \in S \setminus K$ that $v_j = b_j = 0$. The proof of the claim proceeds in several steps:

i) $u_S(\boldsymbol{b}) \leq u_S(\boldsymbol{v})$

   This holds because otherwise there is a player $i \in S$ with $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v}) = u_i(\boldsymbol{b}_{-i}, v_i)$. Here the equality is due to the induction hypothesis. This contradicts SP.

ii) For all $i \in K : (i \in Q(\boldsymbol{b}) \iff i \in Q(\boldsymbol{v}))$

   Let $i \in K$. Obviously, if $i \in Q(\boldsymbol{v})$, then $i \in Q(\boldsymbol{b})$ because $b_i = b^\infty$. If $D = \mathbb{R}^n$ then $i \notin Q(\boldsymbol{v})$ similarly implies $i \notin Q(\boldsymbol{b})$ because $b_i = \mathfrak{b}^- < 0$. On the other hand, if $D = \mathbb{R}^n_{\geq 0}$ then $i \notin Q(\boldsymbol{v})$ implies $v_i > 0$ by definition of S' and $b_i = \mathfrak{b}^- = 0$. Consequently, $i \notin Q(\boldsymbol{b})$ because otherwise $u_i(\boldsymbol{b}) = v_i > 0 = u_i(\boldsymbol{v})$. Here, the first equality is due to NPT and VP. A contradiction to (i).

iii) $u_S(\boldsymbol{b}) \geq u_S(\boldsymbol{v})$

This holds because otherwise there is a player $i \in S$ with $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$. Note that this implies $v_i > 0$, so $i \in K$. Then, due to (i) and (ii), the coalition K can help i at $\boldsymbol{b}$ by bidding as in $\boldsymbol{v}$. This contradicts WSGSP.

iv) $M_K(\boldsymbol{b}) = M_K(\boldsymbol{b})$ and $u(\boldsymbol{b}) = u(\boldsymbol{v})$

Since $K \subseteq S$, steps (i), (ii), and (iii) clearly imply $M_K(\boldsymbol{b}) = M_K(\boldsymbol{v})$. Now if there is a player $j \in [n] \setminus S$ with $u_j(\boldsymbol{b}) \neq u_j(\boldsymbol{v})$, then the coalition K can help j either at $\boldsymbol{v}$ or at $\boldsymbol{b}$ by bidding $\boldsymbol{b}_K$ or $\boldsymbol{v}_K$, respectively. ■

We remark that if $D = \mathbb{R}^n_{\geq 0}$ then it may happen that $Q(\widehat{b}(Q(\boldsymbol{v}))) \supsetneq Q(\boldsymbol{v})$. This is not a problem because all players $j \in Q(\widehat{b}(Q(\boldsymbol{v}))) \setminus Q(\boldsymbol{v})$ satisfy $b_j = v_j = 0$, so $x_j(\boldsymbol{b}) = 0 = x_j(\boldsymbol{v})$. □

4.2.4 WGSP Does Not Imply Separability

We close this section by observing that WGSP, in contrast to WSGSP, does not imply separability, i.e., existence of a cost-sharing method. This observation gives some intuition why WSGSP is notably stronger than only WGSP.

As a simple corollary of the threshold property, the outcome of any SP mechanism could be computed as follows: For every player i, compute the threshold value θ_i, together with a rule what to do in case of indifference, $\phi_i \in \{serve, reject\}$. Then, serve all players i with $b_i > \theta_i$ or ($b_i = \theta_i$ and $\phi_i = serve$) for price θ_i; reject all others. Next, we give a simple idea to transform this observation into a WGSP mechanism.

Definition 4.2.10. *Suppose the outcome of a mechanism M can be computed as follows: For each $j = 1, \ldots, n$, compute $\sigma_j \in [n]$, $\theta_{\sigma_j} \in \mathbb{R}_{\geq 0}$, and $\phi_{\sigma_j} \in \{serve, reject\}$ as functions of $b_{\sigma_1}, \ldots, b_{\sigma_{j-1}}$ so that $\sigma_1, \ldots, \sigma_n$ becomes a permutation of $[n]$. Then M is called a* sequential mechanism.

Note that, for every sequential mechanism, σ_1 is a constant.

Lemma 4.2.11. *Every sequential mechanism is WGSP.*

Proof. Let $\boldsymbol{v}$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$. Consider the first iteration i in which a player from K is considered for input $\boldsymbol{b}$, i.e., $i := \min\{j \in [n] \mid \sigma_j(\boldsymbol{b}) \in K\}$. Obviously we have for all $j \in [i]$ that $\sigma_j(\boldsymbol{b}) = \sigma_j(\boldsymbol{v})$ and $\theta_{\sigma_j(\boldsymbol{b})} = \theta_{\sigma_j(\boldsymbol{v})}$. Hence, $u_{\sigma_i(\boldsymbol{b})}(\boldsymbol{b}) \leq u_{\sigma_i(\boldsymbol{b})}(\boldsymbol{v})$ because the threshold value for player $\sigma_i(\boldsymbol{b}) \in K$ has not changed. Therefore, K cannot be a WGSP-successful coalition. □

Corollary 4.2.12. *There is a WGSP and 1-BB mechanism that is not separable.*

Proof. Let there be $n = 3$ players, and let the cost function be defined by $C(S) := 1$ if $S \neq \emptyset$ and $C(\emptyset) = 0$. Define a sequential mechanism M as follows: If $b_1 \geq \frac{1}{2}$, let

$\boldsymbol{\sigma} := (1,2,3)$ and otherwise $\boldsymbol{\sigma} = (1,3,2)$. Now find the first player (according to the order $\boldsymbol{\sigma}$) who can pay for himself and also for all remaining players with a non-negative bid. Let M serve this set.

Formally: The threshold value is $\theta_{\sigma_i} := 1$ if $b_{\sigma_j} < 1$ for all $j < i$, and $\theta_{\sigma_i} := 0$ otherwise. Moreover, the mechanism is upper-continuous, i.e., $\phi_i :=$ *serve* for all i. Now let $\boldsymbol{b} := (0,1,1)$ and $\boldsymbol{b}' := (\frac{1}{2},1,1)$. Then $Q(\boldsymbol{b}) = Q(\boldsymbol{b}') = \{2,3\}$ but $x(\boldsymbol{b}) = (0,1,0)$ and $x(\boldsymbol{b}') = (0,0,1)$. □

4.3 Egalitarian Mechanisms

Egalitarian mechanisms borrow an algorithmic idea proposed by Dutta and Ray [22] for computing egalitarian solutions. Given a set of players $Q \subseteq [n]$, cost shares are computed by doing the following 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.

Before discussing *most cost-efficient* subsets in Section 4.3.2, we generalize Dutta and Ray's idea by making use of a more general *set selection function* σ and *price function* ρ. 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, $\sigma(Q,N)$ selects the players $S \subseteq Q \setminus N$ who are assigned the cost share $\rho(Q,N)$. We require σ and ρ to be *valid* (see discussion below):

Definition 4.3.1. *Set selection and price functions σ and ρ are* valid *if the following holds for all $N \subsetneq Q' \subseteq Q \subseteq [n]$:*

W1) $\emptyset \neq \sigma(Q,N) \subseteq Q \setminus N$,

W2) $\sigma(Q,N) \subseteq Q' \Longrightarrow \sigma(Q,N) = \sigma(Q',N)$ *and* $\rho(Q,N) = \rho(Q',N)$,

W3) $\rho(Q,N) \leq \rho(Q',N)$,

W4) $0 \leq \rho(Q,N) \leq \rho(Q, N \cup \sigma(Q,N))$.

Now egalitarian mechanisms are defined by Algorithm 4.1.

We shortly comment on validity: Property (W1) implies that any player is assigned a cost share only once and that the algorithm terminates. Property (W2) is a consistency property. It will ensure that the outcome does not change if a rejected player unilaterally modifies his bid such that he is rejected in a different iteration (see the proof of Theorem 4.4.7). Finally, properties (W3) and (W4) imply that the assigned prices are non-decreasing throughout the iterations of the algorithm.

Theorem 4.3.2. *Egalitarian mechanism are SGSP.*

We defer the proof of Theorem 4.3.2 to Section 4.4.1 where it will be an immediate corollary of Theorems 4.4.6 and 4.4.7.

```
Input:    valid set selection and price functions $\sigma, \rho$; bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output:   set of served players $Q \in 2^{[n]}$; vector of cost shares $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$
1: $Q := [n]$; $N := \emptyset$; $\boldsymbol{x} := 0$
2: while $N \neq Q$ do
3:     $S := \sigma(Q,N)$, $a := \rho(Q,N)$
4:     $Q := Q \setminus \{i \in S \mid b_i < a\}$
5:     if $S \subseteq Q$ then $x_i := a$ for all $i \in S$; $N := N \cup S$
```

Algorithm 4.1: Egalitarian mechanisms

4.3.1 Efficiency of Egalitarian Mechanisms

In order to show economic-efficiency bounds, a further property of price functions is needed:

Definition 4.3.3. *Let ρ be a price function, and $\beta > 0$. Then, ρ is called β-*average for costs *C if for all $N \subsetneq Q \subseteq [n]$ and all $\emptyset \neq A \subseteq Q \setminus N$, it holds that $\rho(Q,N) \leq \beta \cdot \frac{C(A)}{|A|}$.*

Lemma 4.3.4. *Let σ and ρ be valid set selection and price functions such that ρ is β-average for costs C. Moreover, let $\emptyset \neq A \subseteq [n]$ and $\boldsymbol{b} \in \mathbb{R}^n$ be a bid vector with $b_i \geq \beta \cdot \frac{C(A)}{|A|}$ for all $i \in A$. Then, the egalitarian mechanism $M = (Q, x)$ serves at least one player $i \in A$, i.e., $A \cap Q(\boldsymbol{b}) \neq \emptyset$.*

Proof. By way of contradiction, assume that $A \cap Q(\boldsymbol{b}) = \emptyset$. Consider the first iteration k in which Algorithm 4.1 rejects a player $i \in A$: This happens in line 4. We indicate all variables in this iteration immediately before line 4 with a subscript k. Since player i is dropped,

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

where the last inequality holds because $A \subseteq Q_k \setminus N_k$ and ρ is β-average. A contradiction. □

Theorem 4.3.5. *Let σ and ρ be valid set selection and price functions such that ρ is β-average for non-decreasing costs C. Suppose the egalitarian mechanism $M = (Q, x)$ always recovers at least the actual cost C'. Then, M is $(2\beta \cdot H_n)$-EFF.*

Proof. Let $\boldsymbol{v}$ contains the true valuations. Denote $Q := Q(\boldsymbol{v})$, $\boldsymbol{x} := x(\boldsymbol{v})$, and let $\overline{v_i} := \max\{v_i, 0\}$. Moreover, let $P \subseteq [n]$ be a set that minimizes $C(P) + \sum_{i \notin P} \overline{v_i}$. We have

$$\begin{aligned} SC(Q) &= C'(Q) + \sum_{i \in [n] \setminus Q} \overline{v_i} \\ &\leq \sum_{i \in Q \cap P} x_i + \sum_{i \in Q \setminus P} x_i + \sum_{i \in [n] \setminus Q} \overline{v_i} && \text{due to cost recovery} \\ &\leq \sum_{i \in Q \cap P} x_i + \sum_{i \in P \setminus Q} \overline{v_i} + \sum_{i \in [n] \setminus P} \overline{v_i} && \text{due to } x_i \leq v_i \text{ for } i \in Q\,. \end{aligned}$$

Hence,

$$\frac{SC(Q)}{C(P)+\sum_{i\notin P}\overline{v_i}} \le \frac{\sum_{i\in Q\cap P} x_i + \sum_{i\in P\setminus Q}\overline{v_i} + \sum_{i\in[n]\setminus P}\overline{v_i}}{C(P)+\sum_{i\in[n]\setminus P}\overline{v_i}} \le \frac{\sum_{i\in Q\cap P} x_i + \sum_{i\in P\setminus Q}\overline{v_i}}{C(P)}\,.$$

The last inequality holds because the fraction on the left-hand side is at least 1 and the same non-negative value is subtracted in both numerator and denominator. Now, consider an arbitrary iteration k when Algorithm 4.1 decides to accept a player $i \in Q \cap P$ in line 5. Fix all variables just after line 3 in that iteration k and indicate them with a subscript k. We have

$$x_i = a_k = \rho(Q_k, N_k) \le \beta \cdot \frac{C((Q\cap P)\setminus N_k)}{|(Q\cap P)\setminus N_k|} \le \beta \cdot \frac{C(Q\cap P)}{|(Q\cap P)\setminus N_k|}\,,$$

where the inequalities hold because $(Q\cap P)\setminus N_k \subseteq Q_k \setminus N_k$, ρ is β-average for costs C, and C is non-decreasing. Now let $i_1,\dots,i_{|Q\cap P|}$ be the players in $Q\cap P$ ordered according to the iteration in which they are accepted. Note that if a player i_j is accepted in iteration k, then $|(Q\cap P)\setminus N_k| \ge |Q\cap P| - j + 1$ because at most $j-1$ players from $Q\cap P$ can be contained in N_k. Consequently, we get

$$x_{i_j} \le \beta \cdot \frac{C(Q\cap P)}{|Q\cap P| - j + 1}$$

and thus $\sum_{i\in Q\cap P} x_i \le \beta \cdot H_{|Q\cap P|} \cdot C(Q\cap P)$.

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 4.3.4, we would have $(P\setminus Q)\cap Q \ne \emptyset$, a contradiction. Inductively and by the same lemma, for every $j = 1,\dots,|P\setminus Q|-1$, there has to be a player $i\in P\setminus Q$ with $v_i < \beta \cdot \frac{C(P\setminus Q)}{|P\setminus Q|-j}$. Therefore, $\sum_{i\in P\setminus Q} v_i \le \beta \cdot H_{|P\setminus Q|} \cdot C(P\setminus Q)$.

Combining the previous bounds and exploiting that C is non-decreasing, we get

$$\frac{SC(Q)}{C(P)+\sum_{i\notin P}\overline{v_i}} \le \frac{\beta \cdot H_{\max\{|Q\cap P|,|P\setminus Q|\}} \cdot (C(Q\cap P) + C(P\setminus Q))}{C(P)} \le 2\beta \cdot H_n\,.$$

This completes the proof. □

4.3.2 Most Cost-Efficient Set Selection

How can concrete set selection and price functions be defined so that they are valid and the previous findings apply? This is what we answer next.

Definition 4.3.6. *A set selection function σ and its corresponding price function ρ are called* most cost-efficient with regard to optimal costs C *if they satisfy (W2) and*

$$\sigma(Q,N) \in \arg\min_{\emptyset\ne T\subseteq Q\setminus N}\left\{\frac{C(N\cup T)-C(N)}{|T|}\right\},$$
$$\rho(Q,N) = \min_{\emptyset\ne T\subseteq Q\setminus N}\left\{\frac{C(N\cup T)-C(N)}{|T|}\right\}.$$

We remark that (W2) is not hard to achieve: A canonical way is, e.g., to always choose the lexicographic maximum of all sets contained in $\arg\min_T\{(C(N \cup T) - C(N))/|T|\}$.

Lemma 4.3.7. *Most cost-efficient set selection and price functions σ and ρ are valid. If the costs C are subadditive then ρ is also 1-average for C.*

Proof. It is a simple observation that σ and ρ fulfill properties (W1)–(W3) of Definition 4.3.1. To see property (W4), 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

$$a \leq \frac{C(N \cup S \cup S') - C(N \cup S)}{|S'|} = a'.$$

Now assume that C is subadditive. Again, let $N \subsetneq Q \subseteq [n]$ and $\emptyset \neq A \subseteq Q \setminus N$. Then,

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

Hence, ρ is 1-average for C if C is subadditive. □

As a corollary of Theorem 4.3.5 and Lemma 4.3.7 we get:

Theorem 4.3.8. *For arbitrary costs C, any egalitarian mechanism M induced by most cost-efficient set selection and prices is 1-BB. If C is non-decreasing and subadditive, then M is also $2H_n$-EFF .*

Unfortunately, evaluating a most cost-efficient set selection function σ can take exponentially many steps (in n). Furthermore, computing optimal costs C is often NP-hard. In Section 4.5, we thus study how to pick "suitable" cost-efficient subsets in polynomial time. We conclude this subsection by showing that our bound on the social cost is tight up to a factor of 2.

Lemma 4.3.9. *For costs C defined by $C(S) := 1$ for all $\emptyset \neq S \subseteq [n]$, any egalitarian mechanism induced by most cost-efficient set selection and prices is no better than H_n-EFF.*

Proof. Let $\boldsymbol{v} := (\frac{1}{i} - \epsilon)_{i=1\ldots n}$ be the true valuation vector, where $\epsilon \in (0, \frac{1}{n})$. Then, $Q(\boldsymbol{v}) = \emptyset$ because in Algorithm 4.1, line 4, one player after the other would be dropped. Now, $C([n]) = 1$ while $SC(\emptyset) = H_n - n \cdot \epsilon$. □

Lemma 4.3.10. *For any $\alpha > 1$, there is a non-decreasing cost function $C : 2^{[4]} \to \mathbb{R}_{\geq 0}$ so that no egalitarian mechanism induced by most cost-efficient set selection and prices is better than α-EFF.*

Proof. Let $z := 6\alpha + 1$. Define C as follows: $C(\{i\}) = 1$ for all $i \in [4]$. Let $C(\{1,2\}) := 2$ and $C(T) := 3$ for any other $T \subset [4]$ with $|T| = 2$. Let $C(\{1,2,3\}) := 4$ and $C(T) := 5$ for any other $T \subset [4]$ with $|T| = 3$. Furthermore, $C([4]) := z$.

Let $M = (Q, x)$ be an egalitarian mechanism induced by most cost-efficient set selection and prices, and suppose the true valuation vector is $\boldsymbol{v} = (1, 1, 2, z - 5)$. Algorithm 4.1 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, $Q(\boldsymbol{v}) = \{1,2,3\}$ and $SC(Q(\boldsymbol{v})) = 4 + (z - 5) = 6\alpha$. However, $C(\{2,3,4\}) + v_1 = 5 + 1 = 6$. □

4.3.3 Submodular and Supermodular Costs

We remark that if a cost function C is submodular, then the egalitarian mechanism induced by most cost-efficient set selection and prices is unique. Moreover, it is also a Moulin mechanism. This holds because by Definition 4.3.6, its cost-sharing method is exactly the egalitarian solution by Dutta and Ray [22]—and for submodular costs, the egalitarian solution is known to produce cross-monotonic cost shares [21].

On the other hand, when costs are supermodular, there is always a singleton set among the most cost-efficient subsets: Suppose the set of remaining players is Q, and the set of already accepted players is N. Consider now an arbitrary set $T = \{t_1, \dots, t_{|T|}\} \subseteq Q \setminus N$. Denote $N_i := N \cup \{t_i\}$. Due to supermodularity, we get

$$\begin{aligned} C(N \cup T) = C\left(\bigcup_{i=1}^{|T|} N_i\right) &\geq C(N_1) + C\left(\bigcup_{i=2}^{|T|} N_i\right) - C(N) \\ &\geq \cdots \geq \sum_{i=1}^{|T|} \big(C(N_i) - C(N)\big) + C(N). \end{aligned}$$

By an averaging argument, there is at least one $i \in [|T|]$ so that

$$\frac{C(N \cup T) - C(N)}{|T|} \geq \frac{\sum_{j=1}^{|T|} \big(C(N_j) - C(N)\big)}{|T|} \geq C(N_i) - C(N),$$

which proves the claim. Hence, when costs are supermodular, egalitarian mechanisms based on most cost-efficient set selection are sequential mechanisms (cf. Definition 4.2.10). They also coincide with Brenner and Schäfer's *singleton mechanisms* [11]. In general, singleton mechanisms can be seen as egalitarian mechanism with set and price selection functions that satisfy only conditions (W1) and (W2) of Definition 4.3.1, but that additionally fulfill that $\sigma(Q, N)$ is always a singleton set.

4.4 Acyclic Mechanisms and SGSP

Acyclic mechanisms have been introduced by Mehta et al. [50] as a generalization (from an algorithmic point of view) of Moulin mechanism. Their outcome is likewise computed by simulating iterative ascending auctions. However, for any set of remaining players S, there is a specific order in which prices are offered to the players. This order is specified by an *offer function* $\tau : 2^{[n]} \to \mathbb{R}_{\geq 0}$. Now, whenever a player cannot afford an offer, a new iteration is started prematurely. Roughly speaking, acyclic mechanisms "conceal" the lack of cross-monotonicity from the players and thus preserve truthfulness. Mehta et al. [50] proved that acyclic mechanisms are WGSP if they are driven by *valid* cost-sharing methods and offer functions.

Definition 4.4.1 (Mehta et al. [50]). *Let ξ be a cost-sharing method and τ be an offer function. For all $i \in S \subseteq [n]$ let*

$$\begin{aligned} E_i(S) &:= \{j \in S \mid \tau_j(S) = \tau_i(S)\}, \\ L_i(S) &:= \{j \in S \mid \tau_j(S) < \tau_i(S)\}, \\ G_i(S) &:= \{j \in S \mid \tau_j(S) > \tau_i(S)\} \end{aligned}$$

*be the subsets of players in S with **e**qual, **l**esser, and **g**reater offer time compared to i. Then τ is called* valid for ξ *if for all $i \in S \subseteq [n]$:*

i) $\xi_i(S \setminus T) = \xi_i(S)$ for all $T \subseteq G_i(S)$ and

ii) $\xi_i(S \setminus T) \geq \xi_i(S)$ for all $T \subseteq G_i(S) \cup (E_i(S) \setminus \{i\})$.

Now, acyclic mechanisms are defined by Algorithm 4.2.

```
Input:   cost-sharing method $\xi$; valid offer function $\tau$; bid vector $\boldsymbol{b}$
Output:  set of players $Q$, vector of cost shares $\boldsymbol{x}$
1: $Q := [n]$
2: while $\exists i \in Q : b_i < \xi_i(Q)$ do
3:     Choose an arbitrary non-empty set $T \subseteq \arg\min_{i \in Q \mid b_i < \xi_i(Q)} \{\tau_i(Q)\}$
4:     $Q := Q \setminus T$
5: $\boldsymbol{x} := \xi(Q)$
```

Algorithm 4.2: Acyclic mechanisms

We remark that Algorithm 4.2 is more general than the original algorithm given by Mehta et al. [50]. They proposed a special case of Algorithm 4.2 where T in line 3 is always a singleton set, chosen deterministically according to some arbitrary tie breaking scheme. For instance, such a deterministic tie breaking scheme could be to always pick the singleton set T consisting only of the player with the smallest number; i.e., formally $T := \min(\arg\min_{i \in Q \mid b_i < \xi_i(Q)} \{\tau_i(Q)\})$.

4.4.1 Egalitarian Mechanisms Are Acyclic

In the following, we show that acyclic mechanisms are well-defined also by Algorithm 4.2. As a welcome by-product, we immediately get that acyclic mechanisms are in fact SGSP and thus notably stronger than only WGSP. Moreover, it is then a straightforward result that egalitarian mechanisms are indeed acyclic.

We start with a simple proposition, which was already shown by Mehta et al. [50]. Even though their proof was only for the special case of Algorithm 4.2 as described above, it can be reused word by word. Like Mehta et al. [50], we say that a player j is *offered the price p in iteration i* if the following conditions hold immediately before line 3 of iteration i: First, $j \in Q$. Second, if k is a player who will be dropped in line 4, then $\tau_j(Q) \leq \tau_k(Q)$. Third, $p = \xi_j(Q)$.

Proposition 4.4.2 (Mehta et al. [50]). *Suppose Algorithm 4.2 offers the price p to player j in iteration i and the price p' in a subsequent iteration. Then $p \leq p'$. Moreover, suppose Q^* is the set of players returned by Algorithm 4.2. Then, at the beginning of iteration i (i.e., immediately after line 2), it holds that $L_j(Q) \subseteq Q^*$ and for all $k \in L_j(Q)$ it holds that $\xi_k(Q^*) = \xi_k(Q)$.*

The next two technical lemmata contain the main technique for showing that the order in which players are dropped is irrelevant.

Lemma 4.4.3. *Let S be an output set of Algorithm 4.2 for the bid vector $\boldsymbol{b}$. Suppose A is a strict superset of S, i.e., $S \subsetneq A \subseteq [n]$. Then the following holds:*

i) There is a player $k \in A \setminus S$ with $b_k < \xi_k(A)$.

ii) Suppose there is a player $j \in S$ with $\xi_j(S) < \xi_j(A)$. Then there is a player $k \in L_j(A) \setminus S$ with $b_k < \xi_k(A)$.

Proof. i) Consider the first iteration in which some $k \in A \setminus S$ is dropped. Immediately before line 4 it holds that $b_k < \xi_k(Q) \leq \xi_k(A)$. Here, the second inequality holds because Proposition 4.4.2 implies $A = Q \setminus B$ for some $B \subseteq G_k(Q) \cup (E_k(Q) \setminus \{k\})$.

ii) Due to Definition 4.4.1, $L_j(A) \setminus S$ is non-empty, and for all $\ell \in L_j(A) \setminus S$ it holds that $\xi_\ell(A) = \xi_\ell(S \cup L_j(A))$ because $S \cup L_j(A) = A \setminus B$ for some set $B \subseteq G_\ell(S)$. Define now $A' := S \cup L_j(A)$. Since $S \subsetneq A' \subseteq A$, it follows by (i) that there is a player $k \in A' \setminus S = L_j(A) \setminus S$ with $b_k < \xi_k(A') = \xi_k(S \cup L_j(A)) = \xi_k(A)$. □

Lemma 4.4.4. *Let $K \subseteq [n]$ be a set of players, and let $\boldsymbol{v}, \boldsymbol{b}$ be K-variants. Consider two distinct executions of Algorithm 4.2, the first for input $\boldsymbol{v}$ and the second for $\boldsymbol{b}$. Let the output sets be R and S, respectively. Suppose that $K \cap S \supseteq K \cap R$ and $\xi_K(S) \leq \boldsymbol{v}_K$ (if $K = \emptyset$ then this is a vacuous truth). Then, $S \subseteq R$.*

Proof. Let r denote the number of iterations (i.e., repetitions *of the body* of the while-loop) needed for the first execution (with input $\boldsymbol{v}$). Define $Q_0 := [n]$, and for $i \in [r]$, define Q_i and T_i as the values of Q and T at the end of iteration i (i.e., immediately after line 4). Clearly, it holds for all $i \in [r]$ that $Q_i = [n] \setminus (T_1 \cup \dots \cup T_i)$. Moreover, $Q_r = R$.

Now let $q \in \mathbb{N}_0$ be maximal so that the first q iterations are identical for both executions; i.e., in each iteration $i = 1, \dots, q$ of the second execution for input $\boldsymbol{b}$ the set

T_i is chosen, too. Consequently, $S \subseteq Q_q$. In the following, we show by induction over $i \in \{q+1 \dots r\}$ that $S \subseteq Q_i$.

We first verify the *base case* $i = q+1$. Consider an arbitrary player $j \in T_{q+1}$. Also during the second execution, he is offered price $\xi_j(Q_q) > v_j$ in iteration $q+1$. Due to Proposition 4.4.2, any price offered to him in a subsequent iteration cannot be smaller. Consequently, $j \notin S$.

Finally, we verify the *induction step* $i \to (i+1)$. Due to the *induction hypothesis*, $S \subseteq Q_i$. Consider again an arbitrary player $j \in T_{i+1}$. By way of contradiction, assume $j \in S$. Then, $\xi_j(Q_i) > v_j \geq \xi_j(S)$ where the first inequality is due to $j \in T_{i+1}$. Consequently, due to Lemma 4.4.3 (ii), it holds that there is a player $k \in L_j(Q_i) \setminus S$ with $b_k < \xi_k(Q_i)$. However, by Proposition 4.4.2, it holds for all players $k \in L_j(Q_i)$ that $k \in R$ and $k \notin T_{i+1}$, so $v_k \geq \xi_k(Q_i) > b_k$. Then, $k \in K$ but $k \in R \setminus S$. This is a contradiction. □

Theorem 4.4.5. *The output of Algorithm 4.2 is independent of the way T in chosen in line 3.*

Proof. This follows immediately from Lemma 4.4.4 for the special case $\boldsymbol{v} = \boldsymbol{b}$, i.e., $K = \emptyset$. □

Theorem 4.4.6. *Acyclic mechanisms are SGSP.*

Proof. Let $M = (Q, x)$ be an acyclic mechanism, let $\boldsymbol{v}$ contain the true valuations, let K be a non-empty coalition, and let $\boldsymbol{b}$ be a K-variant. Define $R := Q(\boldsymbol{v})$ and $S := Q(\boldsymbol{b})$. By way of contradiction, assume that K is SGSP-successful, i.e., ($u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$ and $K \cap S \supseteq K \cap R$) or ($u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$ and $K \cap S \supsetneq K \cap R$).

By Lemma 4.4.4, it follows that $S \subseteq R$. Hence, there is a player $j \in K \cap S$ with $\xi_j(S) < \xi_j(R)$, i.e., even $S \subsetneq R$. Due to Lemma 4.4.3 (ii), there is then a player $k \in L_j(R) \setminus S$ with $b_k < \xi_k(R) \leq v_k$, so $k \in K$ but $k \in R \setminus S$. This is a contradiction. □

We now show that for valid set selection and price functions σ and ρ, Algorithm 4.1 gives the same result as running Algorithm 4.2 with cost-sharing method ξ and offer function τ as defined by Algorithm 4.3. Hence, every egalitarian mechanism is acyclic.

```
Input:  valid set selection and price functions $\sigma, \rho$; set of players $Q \subseteq [n]$
Output: vector of cost shares $\boldsymbol{\xi} \in \mathbb{R}^n_{\geq 0}$; offer times $\boldsymbol{\tau} \in \mathbb{R}^n_{\geq 0}$
1: $N := \emptyset$; $\boldsymbol{\xi} := 0$; $\boldsymbol{\tau} := 0$
2: while $N \neq Q$ do
3:     $S := \sigma(Q, N)$, $a := \rho(Q, N)$
4:     $\xi_i := a$ and $\tau_i := 1 + \max_{j \in Q}\{\tau_j\}$ for all $i \in S$; $N := N \cup S$
```

Algorithm 4.3: Cost-sharing method and offer function of egalitarian mechanisms

Theorem 4.4.7. *Egalitarian mechanisms are acyclic mechanisms.*

Proof. Let σ, ρ be valid set and price selection functions, and let ξ and τ be the cost-sharing method and offer functions defined by Algorithm 4.3. We first show that τ is valid for ξ. Denote the values of all variables immediately after line 3 of iteration k with a subscript k and with the input player set in parentheses. Let $Q \subseteq [n]$ and $i \in Q$ be arbitrary. Fix now k as the (unique) iteration with $i \in S_k(Q)$.

i) Let $T \subseteq G_i(Q)$ be arbitrary. We show by induction over $m \in [k]$ that $N_m(Q) = N_m(Q \setminus T)$, $S_m(Q) = S_m(Q \setminus T)$, and $a_m(Q) = a_m(Q \setminus T)$. Then, $\xi_i(Q) = a_k(Q) = a_k(Q \setminus T) = \xi_i(Q \setminus T)$. For the *base case* $m = 1$, we have $N_1(Q) = N_1(Q \setminus T) = \emptyset$. For the *induction step* $m-1 \to m$, the induction hypothesis implies $N_m(Q) = N_m(Q \setminus T)$.

   Now, both for the base case and for the induction step, we have $S_m(Q) \subseteq Q \setminus T$ because for all $j \in S_m(Q) : \tau_i(Q) = m$ whereas for all $j \in T : \tau_i(T) > k$. Hence, (W2) implies $S_m(Q) = S_m(Q \setminus T)$ and $a_m(Q) = a_m(Q \setminus T)$.

ii) Let $T \subseteq G_i(Q) \cup (E_i(Q) \setminus \{i\})$ be arbitrary. With exactly the same inductive argument as for (i), we get for all $m \in [k-1]$ that $N_m(Q) = N_m(Q \setminus T)$, $S_m(Q) = S_m(Q \setminus T)$, and $a_m(Q) = a_m(Q \setminus T)$. Moreover, also $N_k(Q) = N_k(Q \setminus T)$. Now, due to property (W3) and $Q \setminus T \subseteq Q$, we have $a_k(Q) \leq a_k(Q \setminus T)$. Furthermore, $a_k(Q \setminus T) \leq \xi_i(Q \setminus T)$ since a is non-decreasing in Algorithm 4.3 due to property (W4). Thus, $\xi_i(Q) = a_k(Q) \leq a_k(Q \setminus T) \leq \xi_i(Q \setminus T)$.

Finally, we show that the egalitarian mechanism induced by σ, ρ yields the same outcome as the acyclic mechanism induced by ξ, τ. Whenever Algorithm 4.1 accepts a set $S := \sigma(Q, N)$ this means that the players in S have the minimum offering time of those in $Q \setminus N$ and that $b_i \geq a := \rho(Q, N)$ for all $i \in S$. Consequently, also the acyclic mechanism serves these players for the same price. On the other hand, when Algorithm 4.1 rejects players from S, the same players are also rejected by the acyclic mechanism, due to Theorem 4.4.5. □

4.4.2 Sequential Stand-Alone Mechanisms

We close this section by noting that SGSP and 1-BB alone is in fact not hard to achieve. The following (sequential) mechanisms are called *sequential stand-alone mechanisms* by Moulin [51] and work as follows: Start with the empty player set Q and do the following iteratively, for every player $i = 1, \dots, n$: If i can afford his marginal cost $C(Q \cup \{i\}) - C(Q)$, then charge him this price and add him to Q. Otherwise, he will not be served. A formal definition is given in Algorithm 4.4.

Lemma 4.4.8. *Sequential stand-alone mechanisms are 1-BB. Moreover, they are acyclic mechanisms and thus SGSP.*

Proof. Define the cost-sharing method ξ and the offer function τ as follows. For $S \subseteq [n]$ and $i \in [n]$, let $\xi_i(S) := C(S \cap [i]) - C(S \cap [i-1])$. Furthermore, $\tau_i(S) := i$. Note that τ is indeed valid for ξ. It is now a simple observation that Algorithm 4.2 with input ξ, τ, and $\boldsymbol{b}$ yields the same output as Algorithm 4.4 with input C and $\boldsymbol{b}$. □

```
Input:   non-decreasing cost function $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$; bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output:  set of served players $Q \in 2^{[n]}$; vector of cost shares $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$
1: $Q := \emptyset$; $\boldsymbol{x} := 0$
2: for $i := 1, \dots, n$ do
3:     if $b_i \geq C(Q \cup \{i\}) - C(Q)$ then
4:         $x_i := C(Q \cup \{i\}) - C(Q)$; $Q := Q \cup \{i\}$
```

Algorithm 4.4: Sequential stand-alone mechanisms

However, for many natural cost-sharing problems with subadditive costs, the economic efficiency of sequential stand-alone mechanisms is poor. There is hence good reason to use more advanced mechanisms like our egalitarian ones:

Lemma 4.4.9. *For the cost function $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$ with $C(S) = 1$ for all $\emptyset \neq S \subseteq [n]$, sequential stand-alone mechanisms are no better than n-EFF.*

Proof. Let $M = (Q, x)$ be the sequential stand-alone mechanism for C. Let $\boldsymbol{v} := (1 - \varepsilon)_{i=1\dots n}$. Then, $Q(\boldsymbol{v}) = \emptyset$ and thus $SC(Q(\boldsymbol{v})) = (1 - \varepsilon) \cdot n$. However, $SC([n]) = 1$. □

Interestingly, sequential stand-alone mechanisms are useful for cost-sharing problems with supermodular costs. Recall that supermodular costs imply that serving two disjoint sets of players separately is never more costly than serving both groups at once. They can best be seen as a result of congestion effects that occur in the underlying optimization problem. This includes, for instance, traffic networks where the objective is the total (or average) latency or min-sum scheduling problems (see, e.g., Schulz and Uhan [68]).

In order to show bounds on the economic efficiency, we slightly extend the notion of subadditivity: We say a cost function C is *α-subadditive* ($\alpha \geq 1$) if for all $A, B \subseteq [n]$ it holds that $C(A \cup B) \leq \alpha \cdot (C(A) + C(B))$.

Theorem 4.4.10. *For any costs C the sequential stand-alone mechanism M is 1-BB. If C is supermodular and α-subadditive, i.e., it always holds that $C(A) + C(B) - C(A \cap B) \leq C(A \cup B) \leq \alpha \cdot (C(A) + C(B))$, then M is also α-EFF.*

In order to prove Theorem 4.4.10, we use a result by Brenner and Schäfer [11]. They gave a bound on the economic efficiency of singleton mechanisms, which are clearly a superclass of sequential stand-alone mechanisms. The proof bears some resemblance to Theorem 4.3.5 and is not repeated here. Note that we state the result only in terms of sequential stand-alone mechanisms.

Definition 4.4.11 (Brenner and Schäfer [11]). *A cost-sharing method ξ is said to be* weakly monotone *with respect to costs C if for all sets of players $A \subseteq B \subseteq [n]$ it holds that $\sum_{i \in A} \xi_i(B) \geq C(A)$.*

Proposition 4.4.12 (Brenner and Schäfer [11]). *Let M be a sequential stand-alone mechanism so that its induced cost-sharing method ξ is weakly monotone with respect to costs C. Suppose that for all $A, B \subseteq [n]$ it holds that $C(A \cup B) \leq \alpha \cdot (C(A) + C(B))$. Then, M is α-EFF.*

Proof (of Theorem 4.4.10). Let ξ denote the cost-sharing method induced by M. It is straightforward to see that ξ is "cross-monotonic with reversed signs" (formally, $-\xi$ is cross-monotonic): Let $A \subseteq B \subseteq [n]$. Then it holds for all $i \in [n]$ that $\xi_i(B) = C(B \cap [i]) - C(B \cap [i-1]) \geq C(A \cap [i]) - C(A \cap [i-1]) = \xi_i(A)$. Consequently, ξ is weakly monotone with respect to C because

$$\sum_{i \in A} \xi_i(B) \geq \sum_{i \in A} \xi_i(A) = C(A).$$

Now the proof follows by Proposition 4.4.12. □

4.5 A Framework for Polynomial-Time Computation

In this section, we show how to solve cost-sharing problems in polynomial time by using egalitarian mechanisms with a set selection function that picks the most cost-efficient set *with regard to costs of approximate solutions*.

Formally, an *optimization problem* with the objective to minimize cost is a triple $\Pi = (D, \boldsymbol{S} = (S_I)_{I \in D}, \boldsymbol{f} = (f_I)_{I \in D})$, where D is the set of problem instances (**d**omain) such that for any **i**nstance $I \in D$, S_I is the set of feasible **s**olutions, and $f_I : S_I \to \mathbb{R}_{\geq 0}$ is a function mapping any solution to its cost.

We henceforth write a cost-sharing problem as a pair $\Phi = (\Pi, \textsc{Inst})$, where Π is the underlying optimization problem and $\textsc{Inst} : 2^{[n]} \to D$ denotes the function mapping any subset of the n players to the respective instance of Π. In particular, Φ implicitly defines the optimal cost $C : 2^{[n]} \to \mathbb{R}_{\geq 0}$ by $C(T) := \min_{Z \in S_{\textsc{Inst}(T)}} \{f(Z)\}$. Moreover, for any algorithm $\textsc{alg}$ that computes feasible solutions for Π, we define $C_{\textsc{alg}} : 2^{[n]} \to \mathbb{R}_{\geq 0}$ by $C_{\textsc{alg}}(T) := f(\textsc{alg}(\textsc{Inst}(T)))$.

Resorting to approximate solutions does, of course, 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 $\textsc{alg}$ that is *monotonic* (see, e.g., Murgolo [53]): 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 requirement in the case of scheduling), and reducing a player's size must not increase the cost of the algorithm's solution. Provided that this property holds we can then simply number the players in the order of their size such that $C_{\textsc{alg}}(MIN_{|U|}\, T) \leq C_{\textsc{alg}}(U)$ 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 $C_{\textsc{alg}}$ is needed whereas $\textsc{alg}$ itself does not need to be monotonic any more.

Definition 4.5.1. *Let $\Phi = (\Pi, \textsc{Inst})$ be a cost-sharing problem. Suppose* ALG *is an approximation algorithm for Π, and $C_{\text{mono}} : 2^{[n]} \to \mathbb{R}_{\geq 0}$ is a cost function that satisfies the following:*

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

Then, the pair $(\textsc{alg}, C_{\text{mono}})$ is called a β-relaxation for Φ.

Since we do not require C_{mono} to be subadditive (as necessary to directly apply Theorem 4.3.5), some additional care is needed as described in the following.

Given a β-relaxation $R := (\textsc{alg}, C_{\text{mono}})$, we define set selection and price functions σ_R and ρ_R recursively as follows. Suppose the set of remaining players is Q and the set of already accepted players is N. Let ξ be the vector of cost shares computed by Algorithm 4.3 for input σ_R, ρ_R, and N. Moreover, let

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

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

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

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

Lemma 4.5.2. *Let $R = (\textsc{alg},\ C_{\text{mono}})$ be a β-relaxation for some cost-sharing problem Φ. Then the following holds:*

i) σ_R and ρ_R are valid.

ii) ρ_R is β-average for C.

Proof. i) Let $\sigma := \sigma_R$, $\rho := \rho_R$. Let ξ be the cost-sharing method induced by σ and ρ. We show that Definition 4.3.1 holds. Clearly, properties (W1) and (W2) are fulfilled. To see (W3), let $N \subsetneq Q' \subseteq Q \subseteq [n]$. Define $\Sigma(N) := \sum_{i \in N} \xi_i(N)$ and $S := \sigma(Q,N), k := |S|$ and $S' := \sigma(Q',N), k' := |S'|$. Since $1 \leq k' \leq |Q' \setminus N| \leq |Q \setminus N|$,

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

Furthermore,

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

Since $\rho(Q', N)$ is equal to one of these upper bounds, we have $\rho(Q, N) \leq \rho(Q', N)$.

Finally, to see property (W4), 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,

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

Moreover, we have $MIN_{k+k'}(Q \setminus N) = S \cup S'$. Also, it is easy to see that $\Sigma(N') = \Sigma(N) + k \cdot \rho(Q, N)$ by making use of property (W2), similarly as in first part of the proof of Theorem 4.4.7. Consequently,

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

implying that

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

Again, $\rho(Q, N')$ is the minimum of the upper bounds, and therefore $\rho(Q, N) \leq \rho(Q, N \cup \sigma(Q, N))$.

ii) Let $N \subsetneq Q \subseteq [n]$ and $A \subseteq Q \setminus N$. Then,

$$\rho_R(Q,N) \leq \frac{C_{\text{mono}}(MIN_{|A|}(Q \setminus N))}{|A|} \leq \frac{C_{\text{mono}}(A)}{|A|} \leq \frac{\beta \cdot C(A)}{|A|} . \qquad \square$$

To also compute a feasible solution for the instance of the optimization problem that corresponds to the players served by an egalitarian mechanism, we need:

Definition 4.5.3. *Let $\Phi = (\Pi, \textsc{Inst})$ be a cost-sharing problem where $\Pi = (D, \boldsymbol{S}, \boldsymbol{f})$. Then, Φ is called* mergable *if for all disjoint $T, U \subseteq [n]$ and for all $X \in S_{\textsc{Inst}(T)}$ and $Y \in S_{\textsc{Inst}(U)}$, there is a $Z \in S_{\textsc{Inst}(T \cup U)}$ with $f(Z) \leq f(X) + f(Y)$. We denote this operation by $Z = X \oplus Y$.*

Based on σ_R and ρ_R, Algorithm 4.5 completely solves the cost-sharing problem, including computing a feasible solution for the underlying optimization problem. In the following, we verify correctness.

Lemma 4.5.4. *Let $R = (\textsc{alg}, C_{\text{mono}})$ be a β-relaxation for a mergable cost-sharing problem Φ. The following holds:*

i) At the end of each iteration of Algorithm 4.5, it holds that $\boldsymbol{x} = \xi(N)$ where ξ is the cost-sharing method defined by Algorithm 4.3 for input σ_R and ρ_R.

ii) In every iteration, line 3 of Algorithm 4.5 needs at most $2n$ evaluations of C_{mono}.

iii) The mechanism defined by Algorithm 4.5 is β-BB.

```
Input:   $\beta$-relaxation $R = (\textsc{alg}, C_{\text{mono}})$; bid vector $\boldsymbol{b} \in \mathbb{R}^n$
Output:  set of served players $Q \in 2^{[n]}$, vector of cost shares $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$,
         solution $Z \in S_{\textsc{Inst}(Q)}$
 1: $\boldsymbol{x} := 0, Q := [n], N := \emptyset, Z :=$ “empty solution”
 2: while $N \neq Q$ do
 3:     $S := \sigma_R(Q,N)$; $a := \rho_R(Q,N)$
 4:     $Q := Q \setminus \{i \in S \mid b_i < a\}$
 5:     if $S \subseteq N$ then
 6:         if $C_{\text{mono}}(N \cup S) - \sum_{i \in N} x_i \leq C_{\text{mono}}(S)$ then
 7:             $Z := \textsc{alg}(\textsc{Inst}(N \cup S))$
 8:         else
 9:             $Z := Z \oplus \textsc{alg}(\textsc{Inst}(S))$
10:         $N := N \cup S$; $x_i := a$ for all $i \in S$
```

Algorithm 4.5: Egalitarian mechanisms with β-relaxations

Proof. Consider the execution of Algorithm 4.5 for input R and $\boldsymbol{b}$. Let $m \in \mathbb{N}$ be the number of iterations needed. For all $k \in [m]_0$, indicate the value of all variables at the end of the k-th iteration (i.e., immediately after line 10 if $k > 0$, and immediately before line 2 if $k = 0$) with a superscript k. Moreover, let $p(k)$ be the number of times line 5 has evaluated to true before the end of iteration k.

i) We show that $\xi(N^k)$ is computed in exactly the same way as $\boldsymbol{x}^k$ was. Consider therefore the execution of Algorithm 4.3 for input σ_R, ρ_R, and N^k. Indicate the value of all variables at the end of the j-th iteration with a tilde and a superscript j. Again, superscript 0 refers to the variable values immediately before the while-loop.

   We prove by induction over $k \in [m]_0$ that $\widetilde{N}^{p(k)} = N^k$ and $\widetilde{\xi}^{p(k)} = \boldsymbol{x}^k$. Clearly, the *base case* $k = 0$ is fulfilled because $\widetilde{N}^0 = N^0 = \emptyset$ and $\widetilde{\xi}^0 = \boldsymbol{x}^0 = 0$. In the following, we verify the *induction step* $(k-1) \to k$. If line 5 of Algorithm 4.5 evaluated to false then $p(k) = p(k-1)$, $N^k = N^{k-1}$, and $\boldsymbol{x}^k = \boldsymbol{x}^{k-1}$, which proves the induction step for this case.

   Consider therefore the case that line 5 evaluated to true. Then, $p(k)-1 = p(k-1)$. We have that $\widetilde{S}^{p(k)} = \sigma(\widetilde{Q}^{p(k)}, \widetilde{N}^{p(k)-1}) = \sigma(N^k, N^{k-1})$ where the last inequality is due to the induction hypothesis. Moreover, we have $S^k = \sigma(Q^{k-1}, N^{k-1}) = \sigma(N^k, N^{k-1})$ where the last equality is due to (W2) and $S^k \subseteq N^k$. Hence, $\widetilde{S}^{p(k)} = S^k$ and likewise $\widetilde{a}^{p(k)} = a^k$. The induction step follows.

ii) This follows directly from the definition of σ_R and ρ_R.

iii) Define $\Sigma(N^k) := \sum_{i \in N_k} x_i^k$. We show by induction over $k \in [m]_0$ that $f(Z^k) \leq \Sigma(N^k) \leq C_{\text{mono}}(N^k)$.

The base case $k = 0$ holds trivially. For the induction step $(k-1) \to k$, we only need to consider the case that line 5 evaluated to true in iteration k. Otherwise, $Z^k = Z^{k-1}$ and $N^k = N^{k-1}$, so we would be done. Now, if $C_{\text{mono}}(N^{k-1} \cup S^k) - \Sigma(N^{k-1}) \leq C_{\text{mono}}(S^k)$ then $f(Z^k) = C_{\text{ALG}}(N^k) \leq C_{\text{mono}}(N^k) = \Sigma(N^k)$. Otherwise,

$$\begin{aligned} f(Z^k) &= f(Z^{k-1} \oplus \textsc{alg}(\textsc{Inst}(S^k)) \\ &\leq f(Z^{k-1}) + C_{\text{ALG}}(S^k) \leq \Sigma(N^{k-1}) + C_{\text{mono}}(S^k), \end{aligned}$$

where the last inequality is due to the induction hypothesis and because C_{ALG} is a lower bound for C_{mono}. Now, $\Sigma(N^k) = \Sigma(N^{k-1}) + C_{\text{mono}}(S^k)$. Since line 6 evaluated to false, we moreover have $\Sigma(N^{k-1}) + C_{\text{mono}}(S^k) < C_{\text{mono}}(N^{k-1} \cup S^k) = C_{\text{mono}}(N^k)$. Hence, the induction step follows.

Clearly, the output of Algorithm 4.5 is $Q^m = N^m$, $\boldsymbol{x}^m$, and Z^m. We have shown that $C'(Q^m) = f(Z^m) \leq \Sigma(Q^m) \leq C_{\text{mono}}(Q^m) \leq \beta \cdot C(Q^m)$. This completes the proof. □

As a corollary of Lemmata 4.5.2 and 4.5.4, we obtain:

Theorem 4.5.5. *Let Φ be a mergable cost-sharing problem, and let $(\textsc{alg}, C_{\text{mono}})$ be a β-relaxation for Φ. Then the mechanism defined by Algorithm 4.5 is SGSP, β-BB, and $(2\beta \cdot H_n)$-EFF. Moreover, Algorithm 4.5 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.*

4.6 Applications

We use three approaches for obtaining β-relaxations that are polynomial-time computable in the size of the *succinct representation* of the cost-sharing problem plus the bid vector: Monotonic approximation algorithms (Section 4.6.2), a non-monotonic approximation algorithm with a polynomial-time computable monotonic bound C_{mono} (Section 4.6.3), and optimal costs that are monotonic and polynomial-time computable (Section 4.6.4). Subsequently, we also give some remarks about applying sequential stand-alone mechanisms to cost-sharing problems with supermodular optimal costs (Section 4.6.5).

4.6.1 Makespan Minimization and Bin Packing Cost-Sharing Problems

A makespan cost-sharing problem $Q||C_{\max}$ is *succinctly represented* by a pair $(\boldsymbol{p}, \boldsymbol{s})$ where $\boldsymbol{p} \in \mathbb{N}^n$ contains the processing requirements of the n jobs, and $\boldsymbol{s} \in \mathbb{N}^m$ contains the speeds of the m machines. Recall that each player owns exactly one job and that for any set of players $S \subseteq [n]$, $C(S)$ is the value of a minimum-makespan schedule for the jobs from S.

A bin packing cost-sharing problem is succinctly represented by a vector of item sizes $\varsigma \in (0,1]^n$. Each player owns exactly one item, and for any set of players $S \subseteq [n]$, $C(S)$ is the minimum number of bins with capacity 1 that are needed for S.

In order to keep our notation clean when designing β-relaxations that fulfill Definition 4.5.1, we assume in this section that players' indices are always sorted in ascending order of their processing requirements (in the case of scheduling) or item sizes (in the case of bin packing). This is without loss of generality: Otherwise, players could be sorted (in a deterministic way) before Algorithm 4.5 is called, which adds only $O(\log n)$ to the running time and is thus always negligible.

Lemma 4.6.1. *Any bin packing or makespan cost-sharing problem $\Phi = (\Pi, \textsc{Inst})$ is mergable in time $O(n)$. Moreover, $\textsc{Inst}$ is computable in linear time (in the size of the succinct representation of Φ).*

Proof. For bin packing with disjoint item/player sets T and U, we obtain a bin packing for $T \cup U$ by taking both the bins with items from T and the bins with items from U. The costs (number of bins) simply add up. For scheduling 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. □

4.6.2 Monotonic Approximation Algorithms

Makespan Costs on Identical Machines We start by considering identical-machine makespan cost-sharing problems ($P||C_{\max}$). Their succinct representation is $(\boldsymbol{p}, m)$ where $\boldsymbol{p} \in \mathbb{N}^n$ and $m \in \mathbb{N}$. The LPT (longest processing time first) algorithm [27] is known to be a $\frac{4m-1}{3m}$-approximation algorithm for this problem. Recall that it processes the jobs in decreasing order and assigns each job to the machine on which its completion time will be smallest. Its running time is $O(n \cdot \log n)$ for the sorting phase and $O(n \cdot \log m)$ for the job assignment phase. For identical machines, we show that LPT is monotonic with regard to processing requirements. In order to do so, we first need a technical lemma. Let SORT denote a function that sorts the components of a vector in ascending order.

Lemma 4.6.2. *Let $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ be vectors whose components are sorted in ascending order. Moreover, let $c, d \in \mathbb{R}$ and define $\boldsymbol{a}' := \textsc{sort}(\boldsymbol{a}_{-1}, a_1 + c)$ and $\boldsymbol{b}' := \textsc{sort}(\boldsymbol{b}_{-1}, b_1 + d)$. Suppose that $\boldsymbol{a} \le \boldsymbol{b}$ and $c \le d$; then it holds that $\boldsymbol{a}' \le \boldsymbol{b}'$.*

Proof. Let $j, k \in [n]$ be arbitrary with $a'_j = a_1 + c$ and $b'_k = b_1 + d$. Note that a value may occur several times in the vector. By definition,

$$\begin{aligned}\boldsymbol{a}' &= (a_2, a_3, \dots, a_j,\ a_1 + c,\ a_{j+1}, \dots, a_n) \text{ and}\\ \boldsymbol{b}' &= (b_2, b_3, \dots, b_k, b_1 + d, b_{k+1}, \dots, b_n).\end{aligned}$$

Now let $i \in [n]$ be arbitrary. We verify that $a'_i \le b'_i$. Note that

$$a'_i = \begin{cases} a_{i+1} & \text{if } i < j \\ a_1 + c \in [a_i, a_{i+1}] & \text{if } i = j \\ a_i & \text{if } i > j \end{cases} \quad \text{and} \quad b'_i = \begin{cases} b_{i+1} & \text{if } i < k \\ b_1 + d \in [b_i, b_{i+1}] & \text{if } i = k \\ b_i & \text{if } i > k. \end{cases}$$

Consequently, $b'_i \ge b_i$ and, if $i < n$, then $a_{i+1} \ge a'_i$. By case analysis, we get:

- Case $i \geq k$ and $i \leq j$: Then $b'_i \geq b'_k = b_1 + d \geq a_1 + c = a'_j \geq a'_i$.
- Case $i \geq k$ and $i > j$: Then $b'_i \geq b_i \geq a_i = a'_i$.
- Case $i < k$: Then $b'_i = b_{i+1} \geq a_{i+1} \geq a'_i$. □

Lemma 4.6.3. *Let $n, m \in \mathbb{N}$, $i \in [n]$, and $\boldsymbol{p}, \boldsymbol{p}' \in \mathbb{N}^n$ be i-variants with $p_i < p'_i$. Then $f(\textsc{lpt}(\boldsymbol{p}, m)) \leq f(\textsc{lpt}(\boldsymbol{p}', m))$.*

Proof. We will compare the executions of LPT for input $\boldsymbol{p}$ and $\boldsymbol{p}'$. For input $\boldsymbol{p}$, denote by $\boldsymbol{l}(i) \in \mathbb{R}^m$ the vector that, at the end of iteration $i \in [n]_0$, contains the completion times of all m machines, sorted in ascending order. For input $\boldsymbol{p}'$, define $\boldsymbol{l}'(i)$ correspondingly. By iteration 0 we denote the precondition $\boldsymbol{l}(0) = \boldsymbol{l}'(0) = (0, \dots, 0)$ of LPT's inner loop.

Define $\boldsymbol{q} := \textsc{sort}(\boldsymbol{p})$ and $\boldsymbol{q}' := \textsc{sort}(\boldsymbol{p}')$. By Lemma 4.6.2, it holds that $\boldsymbol{q} \leq \boldsymbol{q}'$. We prove by induction that for all iterations $i \in [n]_0$ it holds that $\boldsymbol{l}(i) \leq \boldsymbol{l}'(i)$. Obviously, the *base case* $i = 0$ is fulfilled by definition. For the *induction step* $i \to (i+1)$, note that by definition of LPT, it holds that $\boldsymbol{l}(i+1) = \textsc{sort}(\boldsymbol{l}_{-1}(i), l_1(i) + q_{n-i})$ and $\boldsymbol{l}'(i+1) = \textsc{sort}(\boldsymbol{l}'_{-1}(i), l_1(i) + q'_{n-i})$. Hence, $\boldsymbol{l}(i+1) \leq \boldsymbol{l}'(i+1)$ due to Lemma 4.6.2.

Now, the makespan of the solution for input $\boldsymbol{p}$ is $f(\textsc{lpt}(\boldsymbol{p}, m)) = \boldsymbol{l}_m(n) \leq \boldsymbol{l}'_m(n) = f(\textsc{lpt}(\boldsymbol{p}', m))$, i.e., no greater than the makespan of the solution for input $\boldsymbol{p}'$. □

Remark. For non-identical machines, LPT is not monotonic with regard to processing requirements. Let $\boldsymbol{s} := (200, 99)$, $\boldsymbol{p} := (205, 200, 150, 150)$, and $\boldsymbol{p}' := (200, 200, 150, 150)$. Then, the corresponding costs are $f(\textsc{lpt}(\boldsymbol{p}, \boldsymbol{s})) = \frac{101}{40}$ and $f(\textsc{lpt}(\boldsymbol{p}', \boldsymbol{s})) = \frac{110}{40}$.

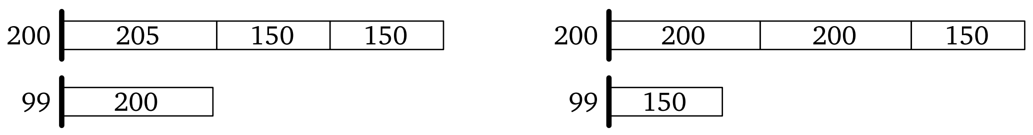


Figure 4.3: For non-identical machines, LPT is not monotonic with regard to processing requirements

As a corollary of Lemma 4.6.3 we get:

Theorem 4.6.4. *For any identical-machine makespan cost-sharing problem $(P||C_{\max})$ with succinct representation $(\boldsymbol{p}, m)$, where $p_1 \leq \cdots \leq p_n$, it holds that $(\textsc{lpt}, C_{\textsc{lpt}})$ is a $\frac{4m-1}{3m}$-relaxation and Algorithm 4.5 runs in time $O(n^3 \cdot \log m)$.*

Bin Packing and Makespan Costs on Related Machines We also obtain 2-relaxations for bin packing and for makespan cost-sharing problems on related machines $(Q||C_{\max})$. The key ingredient here is the following monotonic approximation algorithm RFFD ("rounded first fit decreasing") for bin backing:

Given the vector of item sizes $\varsigma \in (0,1]^n$, round each size up to the next power of 2, i.e., let item i's rounded size be $\varsigma'_i := 2^{\lceil \log_2 \varsigma_i \rceil}$ for all $i \in [n]$. Then, run the FFD (first fit decreasing) algorithm, which is known to produce an optimal packing for this modified instance ς' (see Coffman, Jr. et al. [15]). 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.

Lemma 4.6.5. *For any bin packing cost-sharing problem with succinct representation ς, where $\varsigma_1 \leq \cdots \leq \varsigma_n$, there is a 2-relaxation for C and Algorithm 4.5 runs in time $O(n^3 \cdot \log n)$.*

Proof. Since RFFD is a monotonic 2-approximation algorithm, it holds that $(\text{RFFD}, C_{\text{RFFD}})$ is a 2-relaxation. The overall running-time when given as input to Algorithm 4.5 is $O(n^3 \cdot \log n)$. □

Remark. The tight bound of FFD is $\frac{11}{9} \cdot opt + \frac{6}{9}$, [20]. However, FFD is not monotonic: Let $\varsigma := (\frac{9}{17}, \frac{9}{17}, \frac{5}{17}, \frac{5}{17}, \frac{5}{17}, \frac{4}{17}, \frac{4}{17}, \frac{4}{17}, \frac{3}{17}, \frac{3}{17})$ and $\varsigma' := (\varsigma_{-2}, \frac{8}{17})$. Then, the corresponding costs are $f(\text{FFD}(\varsigma)) = 3$ and $f(\text{FFD}(\varsigma')) = 4$.

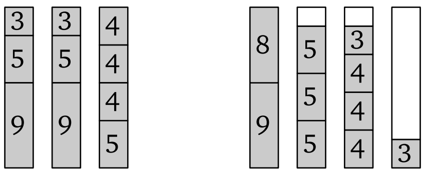


Figure 4.4: FFD is not monotonic

Remark. It is known that the NFD (next fit decreasing) algorithm is monotonic [53] and a 2-approximation algorithm for the bin packing problem. Hence, also $(\text{NFD}, C_{\text{NFD}})$ is a 2-relaxation.

Theorem 4.6.6. *For any related-machine makespan cost-sharing problem ($Q||C_{\max}$) with succinct representation $(\boldsymbol{p}, \boldsymbol{s})$, where $p_1 \leq \cdots \leq p_n$, there is a 2-relaxation. Algorithm 4.5 runs in time $O(n^3 \cdot \log m \cdot \log \sum_{i \in [n]} p_i)$.*

Proof. Consider the decision variant of the following modified bin packing problem: Given $n \in \mathbb{N}$ items with sizes $\varsigma \in \mathbb{Q}^n_{>0}$ and $m \in \mathbb{N}$ bins with capacities $\boldsymbol{c} \in \mathbb{Q}^m_{>0}$, decide whether all n items fit into the m bins. Let FFD* be the following algorithm: Run FFD. That is, in descending order of item sizes, put every item into the first bin where it fits. Note here that no changes are necessary to account for the variable bin capacities. If, at some point an item does not fit any more, return “false”. Otherwise, return “true”.

We show that FFD* is optimal when item sizes are *divisible*, meaning that every item size is exactly divided by any smaller item size (cf. also Coffman, Jr. et al. [15]). Let $n, m \in \mathbb{N}$, $\varsigma \in \mathbb{Q}^n_{>0}$ be a vector of divisible item sizes, and $\boldsymbol{c} \in \mathbb{Q}^m_{>0}$ be the capacity vector.

W.l.o.g., assume $\varsigma_1 \geq \cdots \geq \varsigma_n$ here. Suppose item j is the first item that does not fit any more into one of the m bins, i.e., $\textsc{ffd}^*(\boldsymbol{\varsigma}, \boldsymbol{c}) = \text{false}$. This means that the remaining space in any bin is less than ς_j.

Now note that, for *every* packing of the first $(j-1)$ items, the filling level in each bin is always a multiple of ς_j. Consequently, if in some other bin packing there was still space for item j, there would also be a bin for which the filling level exceeds the bin capacity. This is a contradiction and proves optimality of $\textsc{ffd}^*$ for divisible item sizes.

Now let $\textsc{rffd}^*$ denote the algorithm that first rounds each item size up to the next power of 2 and then calls $\textsc{ffd}^*$. Due to the observation that $\textsc{ffd}^*$ is optimal for divisible item sizes, we know that $\textsc{rffd}^*$ is monotonic in the item sizes. In order to obtain a 2-relaxation for makespan minimization, we can employ $\textsc{rffd}^*$ together with binary search (compare also Algorithm 4.7): Within trivial upper and lower bounds, search for the minimum makespan d so that $\textsc{rffd}^*(\frac{\boldsymbol{p}}{d}, \boldsymbol{s}) = \text{true}$. □

4.6.3 Non-Monotonic Approximation Algorithms with a Polynomial-Time Computable Monotonic Bound

Besides the previous result, we also show how to adapt the PTAS for identical machines ($P||C_{\max}$) by Hochbaum and Shmoys [34]. Although the running time of the PTAS is prohibitive for any small ϵ, the result is theoretically interesting: First, any fixed budget balance greater than 1 can be achieved in polynomial time. Second, the approach here is different to before: Not the PTAS itself is monotonic but only a bound computed inside the algorithm.

The basic idea of the PTAS is a reduction to bin packing (see Algorithm 4.7): Given processing requirements $\boldsymbol{p} \in \mathbb{N}^n$, binary search between trivial upper and lower bounds is employed in order to find a makespan d such that the bin packing instance $\frac{\boldsymbol{p}}{d}$ does *not* need more than m bins of capacity $(1+\varepsilon)$, whereas the bin packing instance $\frac{\boldsymbol{p}}{d-1}$ *does* need more than m bins. Specifically, the PTAS makes use of $\textsc{bpDual}_\varepsilon$, which is an ϵ-dual approximation algorithm for the bin packing problem [34, pp.149–151]. For completeness, it is shown in Algorithm 4.6. $\textsc{bpDual}_\varepsilon$ outputs solutions that are *dual feasible*; this means that $\textsc{bpDual}_\varepsilon$ uses bins of capacity $(1+\epsilon)$ but never needs more bins than the feasible optimal solution (with capacity 1).

Now, for any bin packing instance $\boldsymbol{\varsigma} \in (0,1]^n$, let $S^*_\varsigma \supseteq S_\varsigma$ be the set of all dual-feasible solutions and $f^*_\varsigma : S^*_\varsigma \to \mathbb{N}$ be a function mapping each dual-feasible solution to its cost, i.e., to the number of used bins. We define $g^*_\varsigma : S^*_\varsigma \to \mathbb{N}$ by $g^*_\varsigma(Z) := \max\{f^*_\varsigma(Z), \lceil \sum_{i\in[n]} \varsigma_i \rceil\}$. Hence, the crucial property of g^*_ς is to guarantee that g^*_ς is never less than the total size of all items. We show that g^* is monotonic.

Lemma 4.6.7. *Let $\boldsymbol{\varsigma}, \boldsymbol{\varsigma}' \in \mathbb{Q}^n_{\geq 0}$ be two vectors of item sizes, $i \in [n]$, $\varsigma_i > \varsigma'_i$, and $\boldsymbol{\varsigma}_{-i} = \boldsymbol{\varsigma}'_{-i}$. Then $b := g^*_\varsigma(\textsc{bpDual}_\epsilon(\boldsymbol{\varsigma})) \geq g^*_{\varsigma'}(\textsc{bpDual}_\epsilon(\boldsymbol{\varsigma}')) =: b'$.*

Proof. By way of contradiction, assume $b < b'$. Now consider an execution of $\textsc{bpDual}_\epsilon$ for input $\boldsymbol{\varsigma}'$ (see Algorithm 4.6). All items of size $< \epsilon$ are called "small". Other items of size $\geq \epsilon$ are called "large". In the first phase, round each of the sizes of the large items

```
Input:  approximation $\epsilon \in (0,1)$; item size vector $\varsigma \in (0,1]^n$
Output: allocation $\boldsymbol{a} \in \mathbb{N}^n$
1: Partition the interval $(\epsilon, 1]$ of large sizes into $s := \lceil \frac{1}{\epsilon^2} \rceil$ equal-length subintervals
   $(l_i, l_{i+1}]$. Use $l_i$ as rounded size for all original sizes in this interval.
2: Determine all feasible configurations $(x_1, \dots, x_s) \in \mathbb{N}_0^s$ defined by $\sum_{i=1}^{s} x_i \cdot l_i \leq 1$
   (where $x_i$ is number of items with size in the interval $(l_i, l_{i+1}]$)
3: Use dynamic programming to find an allocation of the large items (using rounded
   sizes; excluding original sizes $\leq \epsilon$), based on following the recurrence:
   $$Bins(y_1, \dots, y_s) := 1 + \min_{\substack{(x_1,\dots,x_s) \\ \text{is feasible} \\ \text{configuration}}} \{Bins(y_1 - x_1, \dots, y_s - x_s)\}$$
   $Bins(y_1, \dots, y_s)$ is the minimum number of bins needed when there are $y_i$ pieces of
   size $l_i$.
4: Enlarge bins to $1 + \epsilon$ and go back to original sizes
5: Pack small items with original size $\leq \epsilon$ into an arbitrary bin containing $\leq 1$. If no
   such bin exists, open a new bin. Let $\boldsymbol{a}$ denote the final allocation of items to bins.
```

Algorithm 4.6: ϵ-dual approximation algorithm for bin packing

to one of constantly many sizes and solve this rounded instance optimally without the small items. Afterwards in the second phase, go back to original sizes and pack small items one after the other into an arbitrary bin containing ≤ 1. If no such bin exists, open a new bin.

Since the first phase computes *optimal* solutions of rounded instances, it is monotonic. Hence, $b < b'$ implies that only in the last phase where the small items are packed, the $(b+1)$-th bin is opened. More precisely, according to line 5 of Algorithm 4.6, there must be a point where b bins are used but all bins contain more than 1. However, this means that $\sum_{i \in [n]} \varsigma'_i > b$, i.e., the total size of all items of instance ς' is more than b. Specifically, $b = g^*_{\varsigma}(\text{BPDUAL}_{\epsilon}(\varsigma)) \geq \lceil \sum_{i \in [n]} \varsigma_i \rceil \geq \sum_{i \in [n]} \varsigma_i > \sum_{i \in [n]} \varsigma'_i > b$. A contradiction. □

Algorithm 4.7 contains the PTAS, together with a crucial extension in line 6. Note that this line is not necessary for the approximation guarantee, but only needed for monotonicity. For $n, m \in \mathbb{N}$ and $\boldsymbol{p} \in \mathbb{N}^n$, define $SIZE(\boldsymbol{p}, m) := \max\{\frac{1}{m} \cdot \sum_{i \in [n]} p_i, p_1, p_2, \dots, p_n\}$.

Note that *lower* in Algorithm 4.7 is always a lower bound on the optimal makespan: Since BPDUAL_{ϵ} is an ϵ-dual approximation algorithm, this holds at the beginning and also whenever *lower* is updated in line 8. On the other hand, $upper \cdot (1 + \epsilon)$ is always an upper bound both on the optimal makespan as well as on the makespan of the schedule $\boldsymbol{a}$. Our crucial extension of the PTAS is as follows: Letting $\varsigma := \frac{\boldsymbol{p}}{d}$, we use the check $g^*_{\varsigma}(\text{BPDUAL}_{\varepsilon}(\varsigma)) \leq m$ in the binary search (instead of testing f^*_{ς} as in the original PTAS).

Let $lower_{\varepsilon}(\boldsymbol{p}, m)$ denote the final value of *lower* returned by Algorithm 4.7 for input ε, $\boldsymbol{p}$, and m. That is, $lower_{\varepsilon}(\boldsymbol{p}, m)$ is the minimum d for which the check $g^*_{\varsigma}(\text{BPDUAL}_{\varepsilon}(\varsigma)) \leq$

```
Input:   approximation $\epsilon \in (0,1)$;
         vector $\boldsymbol{p} \in \mathbb{N}^n$ of processing requirements; number of machines $m \in \mathbb{N}$
Output:  allocation $\boldsymbol{a} \in [m]^n$, lower bound on optimum makespan $lower$
 1: $upper := 2 \cdot SIZE(\boldsymbol{p}, m)$
 2: $lower := SIZE(\boldsymbol{p}, m)$
 3: while $upper \neq lower$ do
 4:     $d := \lfloor (upper + lower)/2 \rfloor$
 5:     $\boldsymbol{a} :=$ BPDUAL$_\epsilon(\frac{\boldsymbol{p}}{d})$; set $b$ to number of bins used in $\boldsymbol{a}$
 6:     $b := \max\{b, \lceil \sum_{i \in [n]} \frac{p_i}{d} \rceil\}$            ▷ Crucial extension for monotonicity
 7:     if $b > m$ then
 8:         $lower := d + 1$                          ▷ Afterwards, still $lower \leq d \leq upper$
 9:     else
10:         $upper := d$                              ▷ Afterwards, still $lower \leq d \leq upper$
11: $\boldsymbol{a} :=$ BPDUAL$_\epsilon(\frac{\boldsymbol{p}}{lower})$                        ▷ Not necessary if $b \leq m$
```

Algorithm 4.7: Modified PTAS for the minimum makespan problem

m evaluates to true. Moreover, let HS$_\varepsilon$ denote the adapted PTAS. Now, $lower_\varepsilon(\boldsymbol{p}, m)$ is a lower bound on the optimal makespan and $(1+\varepsilon) \cdot lower_\varepsilon(\boldsymbol{p}, m)$ is an upper bound on the makespan of the schedule found by HS$_\varepsilon$. Moreover, $lower_\varepsilon(\boldsymbol{p})$ is computed within HS$_\varepsilon$ in polynomial time because monotonicity of g^* ensures that indeed the minimum d is found by the binary search. As a corollary of Lemma 4.6.7, we get:

Theorem 4.6.8. *Let Φ be an identical-machine makespan cost-sharing problem ($P||C_{\max}$) with succinct representation $(\boldsymbol{p}, m)$, where $p_1 \leq \cdots \leq p_n$. Define the monotonic cost function $C_{\text{mono}}(A) := (1+\varepsilon) \cdot lower_\varepsilon(\text{INST}(A))$. Then, (HS$_\epsilon$, C_{mono}) is a $(1+\varepsilon)$-relaxation for Φ, and Algorithm 4.5 runs in time $O(n^{2+\frac{1}{\varepsilon^2}} \cdot \log \sum_{i \in [n]} p_i)$.*

4.6.4 Makespan Problems with Monotonic Optimal Costs

There are several mergable makespan problems for which optimal costs are monotonic and computable in polynomial time. For instance, for the problem of scheduling identical jobs on identical parallel machines ($P|p_i = p|C_{\max}$), it holds that (LPT, C_{LPT}) is a 1-relaxation and Algorithm 4.5 runs in time $O(n^3 \cdot \log m)$. In the following, we give a selection of further such problems (see, e.g., Brucker [12]):

- Symmetric costs:
  - $Q|p_i = p|C_{\max}$
- Variable release dates:
  - $Q|p_i = p, r_i|C_{\max}$
  - $Q|\text{pmtn}, p_i = p, r_i|C_{\max}$

- Variable processing requirements:
  - $Q|\text{pmtn}|C_{\max}$

It is straightforward to see that all of the induced (optimal) cost functions are subadditive and the problems are mergable. This holds as well for the preemptive case. Moreover, the optimal costs are always monotonic in the variable property (release dates or processing requirements) so that determining the most cost-efficient set can always be done in polynomial time by only checking a single set for each cardinality (see Section 4.5): If jobs are ordered by increasing value of the variable property, the first k jobs minimize the cost over all sets of cardinality k. Consequently, we get that 1-relaxations exist for all of the above problems.

Theorem 4.6.9. *For sharing the (optimal) cost induced by any of the above makespan problems, there is a 1-BB and $2H_n$-EFF egalitarian mechanism. Its outcome can be computed in polynomial time.*

4.6.5 Scheduling Problems with Supermodular Costs

We find it interesting to note that Brenner and Schäfer's singleton mechanisms [11] for $P||\sum C_i$ and $1||\sum w_i C_i$ are in fact egalitarian mechanisms based on most cost-efficient set selection: For these problems, the induced (optimal) cost functions are supermodular (see, e.g., Schulz and Uhan [68]). Moreover, for any set of remaining players Q and any set of already accepted players N, a player $i \in Q \setminus N$ with minimal w_i/p_i constitutes a most cost-efficient set—and indeed, singleton mechanisms always choose one of these singleton sets. In particular, the assigned cost shares of each job are equal to the completion time under Smith's rule [73], an algorithm which assigns the jobs in the order of increasing ratios w_i/p_i and which is known to deliver optimal schedules for the above problems in polynomial time.

For $P||\sum w_i C_i$, computing optimal costs is NP-hard, but Smith's rule guarantees an approximation ratio of $(1+\sqrt{2})/2 \approx 1.21$, [40]. It is easy to verify that the costs induced by Smith's rule are supermodular. Somewhat unsurprisingly now, the egalitarian mechanisms induced by always choosing the most cost-efficient set with respect to this *approximation* cost are again equivalent to the singleton mechanisms by Brenner and Schäfer [11] for this problem.

Since the order in which players are offered prices is constant, the above mechanisms are in fact even sequential stand-alone mechanisms. Thus, all of the previously mentioned subclasses of the acyclic-mechanism framework coincide here in a natural way.

For the objective to minimize the completion time on identical parallel machines, Brenner and Schäfer [11] showed that optimal costs are 2-subadditive. Yet, it is a simple observation that the very same proof holds also for related parallel machines:

Proposition 4.6.10 (Brenner and Schäfer [11]). *Let C be the optimal cost function induced by $Q||\sum w_i C_i$. Then, C is 2-subadditive.*

Now recall Theorem 4.4.10 stating that when β-approximate costs are supermodular and α-subadditive, simple sequential stand-alone mechanisms guarantee β-BB and $(\alpha \cdot \beta)$-EFF (regardless of the order of the players). This implies, of course, that the above mechanisms for $P||\sum C_i$ and $1||\sum w_i C_i$ are 1-BB and 2-EFF and the mechanisms for $P||\sum w_i C_i$ are 1.21-BB and 2.42-EFF, as previously shown by Brenner and Schäfer [11]. Another polynomial-time solvable problem with supermodular optimal costs is $Q|p_i = 1|\sum w_i C_i$, [68]. We therefore obtain the following new result:

Theorem 4.6.11. *For sharing the (optimal) cost induced by $Q|p_i = 1|\sum w_i C_i$, every sequential stand-alone cost-sharing mechanism is 1-BB, 2-EFF, and computable in polynomial time.*

As a last remark, the recovered costs of a cross-monotonic cost-sharing methods are always subadditive. Consequently, it is not surprising that Moulin mechanisms suffer bad budget balance if the underlying optimization problem severely violates subadditivity. E.g., for the problem $1||\sum_i C_i$, no cross-monotonic cost-sharing method can be better than $\frac{n+1}{2}$-BB [10]. Obviously, the SGSP mechanisms discussed above tremendously improve on this.

4.7 Conclusion

The pivotal point of this chapter was to study cost-sharing scenarios where the case that a player feels indifferent about being served is negligible. We believe that SGSP (or one of the other collusion-resistance properties without indifferences) is often a viable replacement for the often too limiting GSP requirement. We consider the main asset of our work to be fourfold: (a) Characterizing the relationship between the new collusion-resistance properties. (b) Egalitarian mechanisms; showing existence of SGSP, 1-BB, and $2H_n$-EFF mechanisms for any non-decreasing subadditive costs. (c) Our framework for polynomial-time computability that reduces constructing SGSP, $O(1)$-BB, and $O(\log n)$-EFF mechanisms to finding monotonic approximation algorithms. (d) Showing that acyclic mechanisms are robust against the scheme underbidders are removed; as a consequence, they comprise egalitarian mechanisms and are SGSP—i.e., in a precise sense, they are remarkably stronger than was known before.

Of course, several immediate issues are left often by our work:

- Which other combinatorial optimizations problems can our polynomial-time framework be applied to?
- It is easy to see that rooted Steiner tree cost-sharing problems are mergable and their costs non-decreasing and subadditive; but do they allow for a β-relaxation?
- What are lower bounds on the performance guarantees by *polynomial-time* acyclic mechanism?

Chapter 5

Does Coalition Size Matter?

5.1 Overview of Contribution

In this chapter, we concentrate on the question whether reducing the maximum coalition size that a mechanism should withstand allows for a richer set of possible mechanisms. We say a mechanism is k-GSP (or k-WGSP, respectively) if it ensures collusion resistance up to coalition size k. In detail, our results are:

- While we give (arguably artificial) cost-sharing mechanisms that are k-GSP but not $(k+1)$-GSP, we obtain as our main result that already 2-GSP is equivalent to GSP once we require mechanisms to be *separable*, i.e., cost shares must only depend on the set of served players (and not directly on the bids). We remark that no general technique for the design of truthful cost-sharing mechanisms is known that violates separability. Our result can be seen as a generalization of the main theorem in an article by Mutuswami [55].
- In contrast to the previous result, WGSP is *not* equivalent to 2-WGSP plus separability.
- Even without separability, 2-GSP implies WGSP.

We regard the chief asset of our work to be threefold: First, our results indicate that the substantial "jump" in collusion resistance seems to occur from 1-GSP = SP to 2-GSP and not from $\Theta(1)$-GSP to $\omega(1)$-GSP. Second, GSP is often felt to be too strong an axiom with unrealistic implications on players' capabilities and behavior; now, the fact that GSP is equivalent to merely 2-GSP plus separability gives some a posteriori justification for GSP. Third and last, we firmly believe that our characterizations will facilitate devising and understanding new GSP cost-sharing mechanisms.

5.2 Notions of Non-Manipulability by Small Coalitions

Demand for a certain collusion resistance implies assumptions on players' behavior and their coalition-forming capabilities: For instance, if (a) side-payments are unlikely but (b) players yet have virtually unlimited means to communicate and (c) one expects them to help others even for no personal reward (e.g., by voluntary non-participation in case of indifference), then GSP is an appropriate axiom. Similarly, when players have no

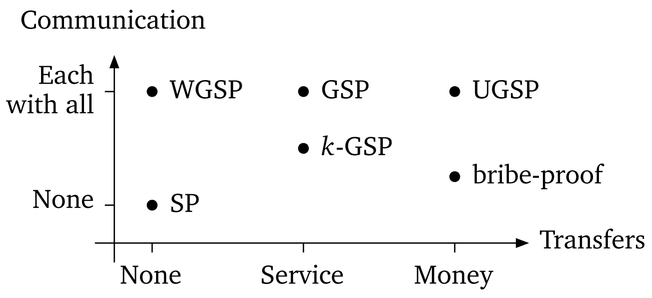


Figure 5.1: Two dimensions of coalition-forming capabilities

means to communicate at all, then simple SP is probably sufficient. One can also think of collusion resistance at the other end of the spectrum: We use the term "ultimate group-strategyproofness" (UGSP) here if a mechanism even prevents that coalitions can improve their *total utility* by manipulation. Essentially, WGSP, GSP, and UGSP imply different levels of transfers that coalitions might accomplish in order to be successful. Figure 5.1 provides a schematic illustration.

Since it seems unlikely that all players can communicate with each other and make binding agreements on collective manipulation, this gives rise to the following natural question: Can we increase the degree of freedom for designing cost-sharing mechanisms by relaxing the GSP requirement with respect to coalition sizes? Surprisingly, we prove in the rest of this chapter that the answer is essentially "no".

Definition 5.2.1. *A cost-sharing mechanism M is k-GSP (or k-WGSP) if for all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and all non-empty coalitions $K \subseteq [n]$ with $|K| \leq k$ there there is no K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$ (or $u_K(\boldsymbol{b}) \gg u_K(\boldsymbol{v})$, respectively).*

We remark that 2-GSP is equal to what Serizawa [70] called "pairwise SP". Moreover, note that 2-GSP immediately implies 2-WGSP, SP, and WUNB.

5.2.1 Resistance Against Coalitions with Side-Payments

We show in the following that resistance against coalitions is essentially infeasible if we assume that players are capable of organizing side-payments. This is an immediate corollary of a result by Schummer [69].

Definition 5.2.2 (Schummer [69]). *A mechanisms M is* bribe-proof *if for all all players $i, j \in [n]$, all side-payments $p \in \mathbb{R}$, and all true valuations $\boldsymbol{v} \in \mathbb{R}^n$ there is no i-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ so that $u_i(\boldsymbol{b}) + p > u_i(\boldsymbol{v})$ and $u_j(\boldsymbol{b}) - p > u_j(\boldsymbol{v})$.*

Note that neither $i = j$ nor $p = 0$ are excluded in the definition, hence bribe-proof implies SP. Intuitively, in a bribe-proof mechanism, player j cannot bribe i with p units of money into misreporting his valuation.

Schummer [69] calls bribe-proof the "weakest intuitive condition" that rules out misreports of coalitions of size two, when assuming that players are capable of side-payments. Indeed, the following is an easy observation:

Lemma 5.2.3. *Let M be a cost-sharing mechanism. Then M is bribe-proof if and only if for all true valuations $\boldsymbol{v}$ and all players $i, j \in [n]$ there is no i-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ so that $u_i(\boldsymbol{b}) + u_j(\boldsymbol{b}) > u_i(\boldsymbol{v}) + u_j(\boldsymbol{v})$.*

Proof. Verifying necessity ("⇐") is trivial, so we only consider sufficiency ("⇒"). Suppose $u_i(\boldsymbol{b}) + u_j(\boldsymbol{b}) = u_i(\boldsymbol{v}) + u_j(\boldsymbol{v}) + \varepsilon$ for some $\varepsilon > 0$. Define $p := u_j(\boldsymbol{b}) - u_j(\boldsymbol{v}) - \frac{\varepsilon}{2}$. Then $u_i(\boldsymbol{b}) + p = u_i(\boldsymbol{b}) + u_j(\boldsymbol{b}) - u_j(\boldsymbol{v}) - \frac{\varepsilon}{2} = u_i(\boldsymbol{v}) + u_j(\boldsymbol{v}) - u_j(\boldsymbol{v}) + \frac{\varepsilon}{2} > u_i(\boldsymbol{v})$. Moreover, $u_j(\boldsymbol{b}) - p = u_j(\boldsymbol{b}) - u_j(\boldsymbol{b}) + u_j(\boldsymbol{v}) + \frac{\varepsilon}{2} > u_j(\boldsymbol{v})$. □

One of Schummer's results is that if the domain of players' types is path-connected[1] and the set of alternatives is finite, then each player's utility does not depend on the other players' bids. For our purposes, since the domain of valuations (i.e., types) is the Euclidean space and thus path-connected, and since there are only finitely many subsets of players, it is sufficient to note:

Proposition 5.2.4 (Schummer [69]). *Let M be a bribe-proof cost-sharing mechanism and $i \in [n]$ be an arbitrary player with true valuation v_i. Suppose $\boldsymbol{b}, \boldsymbol{b}'$ are $([n] \setminus i)$-variants. Then $u_i(\boldsymbol{b}) = u_i(\boldsymbol{b}')$.*

As an immediate consequence we get that bribe-proofness rules out all but trivial cost-sharing mechanisms:

Corollary 5.2.5. *Suppose M is a bribe-proof mechanism. Then, for all players $i \in [n]$, the threshold value $\theta_i(\boldsymbol{b}_{-i})$ is a constant.*

5.2.2 Some Preliminary Implications by SP and WUNB

We start with some immediate consequences of SP and WUNB that will be needed throughout this chapter. Since service-allocation vectors will be more convenient than sets of served players, a mechanism will henceforth be denoted $M = (q, x)$ where $q : \mathbb{R}^n \to \{0,1\}^n$.

Lemma 5.2.6. *Let $M = (q, x)$ be a SP cost-sharing mechanism, $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, $i \in [n]$ be an arbitrary player, and $\boldsymbol{b}$ be an i-variant of $\boldsymbol{v}$. Then:*

i) $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$ and $q_i(\boldsymbol{v}) = 1 \Longrightarrow q_i(\boldsymbol{b}) = 0, u_i(\boldsymbol{b}) = 0 < u_i(\boldsymbol{v})$, and $b_i \leq \theta_i(\boldsymbol{v}_{-i}) = x_i(\boldsymbol{v}) < v_i$

ii) $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$ and $q_i(\boldsymbol{v}) = 0 \Longrightarrow q_i(\boldsymbol{b}) = 1, u_i(\boldsymbol{b}) < 0 = u_i(\boldsymbol{v})$, and $b_i \geq \theta_i(\boldsymbol{v}_{-i}) = x_i(\boldsymbol{b}) > v_i$

Proof. i) If $b_i > x_i(\boldsymbol{v})$, then i could manipulate and improve at $\boldsymbol{b}$ by bidding v_i; hence $b_i \leq x_i(\boldsymbol{v})$ due to SP. If $q_i(\boldsymbol{b}) = 1$, then $b_i \geq x_i(\boldsymbol{b})$ due to VP and $x_i(\boldsymbol{b}) > x_i(\boldsymbol{v})$ because $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$ by assumption; a contradiction. Hence, $q_i(\boldsymbol{b}) = 0$. Then, $u_i(\boldsymbol{b}) = 0$ and $v_i > x_i(\boldsymbol{v})$ because $u_i(\boldsymbol{v}) > 0$.

[1] A topological space X is *path-connected* if every two points $x, y \in X$ can be be connected by a path, i.e., there is a continuous function $f : [0,1] \to X$ with $f(0) = x$ and $f(1) = y$.

ii) This follows immediately because $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v}) = 0$ due to VP. □

Lemma 5.2.7. *Let M be a WUNB cost-sharing mechanism, $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, $i \in [n]$ be an arbitrary player, and $\boldsymbol{b}$ be an i-variant of $\boldsymbol{v}$. Then: $M_i(\boldsymbol{b}) = M_i(\boldsymbol{v}) \Longrightarrow u(\boldsymbol{b}) = u(\boldsymbol{v})$.*

Proof. If there was as player j so that, w.l.o.g., $u_j(\boldsymbol{b}) > u_j(\boldsymbol{v})$, then player i could help j at $\boldsymbol{v}$ by bidding b_i. A contradiction to WUNB. □

Lemma 5.2.8. *Let $M = (q, x)$ be a SP and WUNB cost-sharing mechanism, $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, $i \in [n]$ be an arbitrary player, and $\boldsymbol{b}$ be an i-variant of $\boldsymbol{v}$. Moreover, let $j \in [n] \setminus i$. Then:*

i) $u_j(\boldsymbol{b}) > u_j(\boldsymbol{v}) \Longrightarrow u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$

ii) $u_j(\boldsymbol{b}) < u_j(\boldsymbol{v})$ *and* $q_i(\boldsymbol{v}) = 1 \Longrightarrow q_i(\boldsymbol{b}) = 0$ *and* $b_i < \theta_i(\boldsymbol{v}_{-i}) = x_i(\boldsymbol{v}) \leq v_i$

iii) $u_j(\boldsymbol{b}) < u_j(\boldsymbol{v})$ *and* $q_i(\boldsymbol{v}) = 0 \Longrightarrow q_i(\boldsymbol{b}) = 1$ *and* $b_i > \theta_i(\boldsymbol{v}_{-i}) = x_i(\boldsymbol{b}) \geq v_i$

iv) $v_i = \theta_i(\boldsymbol{v}_{-i}) \Longrightarrow u_j(\boldsymbol{b}) \leq u_j(\boldsymbol{v})$

v) $u_{-i}(\boldsymbol{b}) \leq u_{-i}(\boldsymbol{v})$ *or* $u_{-i}(\boldsymbol{b}) \geq u_{-i}(\boldsymbol{v})$

vi) $u_j(\boldsymbol{b}) > u_j(\boldsymbol{v}) \Longrightarrow \theta_j(\boldsymbol{b}_{-j}) < \theta_j(\boldsymbol{v}_{-j})$

Proof. i) This is a trivial consequence of WUNB and SP.

ii) By WUNB and the threshold property, it holds that $b_i < x_i(\boldsymbol{v})$ because otherwise player i could help j at $\boldsymbol{b}$ by bidding v_i. Hence, $q_i(\boldsymbol{b}) = 0$.

iii) By WUNB, it holds that $q_i(\boldsymbol{b}) = 1$ and $b_i > x_i(\boldsymbol{b})$ because otherwise $u_i(\boldsymbol{b} \mid b_i) = 0$ and player i could help j at $\boldsymbol{b}$ by bidding v_i.

iv) By the threshold property, it holds that $u_i(\boldsymbol{v}) = u_i(\boldsymbol{b}) = 0$. Hence, the proof follows by (i).

v) By way of contradiction, assume there are $j, k \in [n] \setminus i$ with $u_j(\boldsymbol{b}) < u_j(\boldsymbol{v})$ and $u_k(\boldsymbol{b}) > u_k(\boldsymbol{v})$. Due to (ii) and (iii), player i gets the service for either $\boldsymbol{v}$ or $\boldsymbol{b}$, but not for both. We may assume w.l.o.g. that $q_i(\boldsymbol{v}) = 1$ and $q_i(\boldsymbol{b}) = 0$. Let now $\boldsymbol{b}'$ be another i-variant of $\boldsymbol{v}$ and $\boldsymbol{b}$ with $b'_i := x_i(\boldsymbol{v})$. Then the threshold property and (iv) ensure $u_j(\boldsymbol{v}) \leq u_j(\boldsymbol{b}')$ and $u_k(\boldsymbol{b}) \leq u_k(\boldsymbol{b}')$. It follows that if $q_i(\boldsymbol{b}') = 1$, then i can help k at $\boldsymbol{v}$ by bidding b'_i. Correspondingly, if $q_i(\boldsymbol{b}') = 0$, then i could help j at $\boldsymbol{b}$ by bidding b'_i. A contradiction to WUNB.

vi) Note that $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$ by (i). Since $u_j(\boldsymbol{b}) > u_j(\boldsymbol{v}) \geq 0$, it holds that $q_j(\boldsymbol{b}) = 1$ and $x_j(\boldsymbol{b}) < v_j$. Now, if $q_j(\boldsymbol{v}) = 1$, then $\theta_j(\boldsymbol{b}_{-j}) = x_j(\boldsymbol{b}) < x_j(\boldsymbol{v}) = \theta_j(\boldsymbol{v}_{-j})$. On the other hand, if $q_j(\boldsymbol{v}) = 0$, then $\theta_j(\boldsymbol{b}_{-j}) = x_j(\boldsymbol{b}) < b_j = v_j \leq \theta_j(\boldsymbol{v}_{-j})$, where the last inequality is due to the threshold property. □

5.2.3 k-GSP Is Strictly Weaker Than GSP

Before establishing the link between 2-GSP and GSP in the next sections, we give an example showing that k-GSP is not equivalent to GSP, for arbitrary $k < n$. Consider the mechanism defined by Algorithm 5.1.

```
Input:   bid vector $\boldsymbol{b} \in \mathbb{R}^3$
Output:  service allocation $\boldsymbol{q} \in \{0,1\}^3$; cost shares $\boldsymbol{x} \in \mathbb{R}^3_{\geq 0}$
1: if $\boldsymbol{b} = (1,1,1)$ then $\boldsymbol{q} := (1,1,1)$; $\boldsymbol{x} := (1,1,1)$
2: else
3:     $\boldsymbol{q} := (0,0,0)$; $\boldsymbol{x} := (0,0,0)$; $\boldsymbol{\theta} := (1,1,1)$
4:     if $b_1 > 1$ and $b_2 > 1$ then $\theta_3 := 2$
5:     for all $i \in \{1,2,3\}$ with $b_i > \theta_i$ do $q_i := 1$; $x_i := \theta_i$
```

Algorithm 5.1: 3-Player mechanism that is 2-GSP but not GSP

It is easy to see that this mechanism is 2-GSP because for any true valuations $\boldsymbol{v}$ the only player that could ever improve is player 3. In this case, however, $v_1 > 1$ and $v_2 > 1$, so in order to help player 3 both players 1 and 2 have to deviate. We remark that Algorithm 5.1 can be generalized for n players so that it is $(n-1)$-GSP but not n-GSP.

5.3 Group-Strategyproofness Against Only Two Players

5.3.1 Upper Continuity and 2-GSP Together Imply GSP

As a simple starting point, we first consider upper-continuous mechanisms. Recall that upper continuity is a straightforward way for dealing with indifferent players, in that they are always included in the set of served players. Upper-continuity is, e.g., fulfilled by Moulin mechanism and by acyclic mechanisms, but not by our symmetric mechanisms from Chapter 3. We first need the following simple observation that is essentially[2] a corollary of Lemma 4.2.5.

Lemma 5.3.1. *Let $M = (q, x)$ be an upper-continuous 2-GSP cost-sharing mechanism. Then M is also ONB.*

Thus, combined with a result by Mutuswami [55], upper continuity and 2-GSP together imply GSP:

Proposition 5.3.2 (Mutuswami [55]). *Let M be a SP, ONB, and WUNB cost-sharing mechanism. Then, M is also GSP.*

[2] In detail, Lemma 4.2.5 states that WSGSP and upper continuity together imply ONB. It is a simple observation that WSGSP could be replaced by "2-WSGSP" without invalidating the proof. Clearly, 2-GSP implies "2-WSGSP".

Corollary 5.3.3. *Let M be an upper-continuous 2-GSP cost-sharing mechanism. Then, M is also GSP.*

We remark here that Mutuswami [55] assumes non-negative bids in his work, i.e., that the domain of valuations is restricted to $\mathbb{R}^n_{\geq 0}$. Yet, his proof can be used without changes also for the more general setting where negative bids are allowed. In other words, the previous results of this subsection only require CS but not strong CS.

5.3.2 Separability and 2-GSP Together Imply GSP

We now generalize Corollary 5.3.3 to arbitrary separable mechanisms. Specifically, we will obtain as our main result that a 2-GSP cost-sharing mechanism is GSP if and only if it is separable. We start with an auxiliary lemma, stating that every 2-GSP cost-sharing mechanism is at least resistant against coalitions where deviators either do not participate (submit a negative bid) or bid very much.

Lemma 5.3.4. *Let $M = (q, x)$ be a 2-GSP cost-sharing mechanism, $\boldsymbol{v} \in \mathbb{R}^n$ contain the true valuations, $K \subseteq [n]$ be a non-empty coalition, and $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ so that for all $i \in K : b_i \in \{-1, b^\infty\}$. Then, either $u_i(\boldsymbol{b}) = u_i(\boldsymbol{v})$ for all $i \in K$ or $u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$ for at least one $i \in K$.*

In particular, Lemma 5.3.4 implies that a 2-GSP mechanism always computes a *Pareto-optimal* outcome when given truthful bids, meaning that all other outcomes either provide all players with the same utility or make at least one player worse off.

Proof. By way of contradiction, assume that $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$. Roughly speaking, we will look at what happens when players adopt the bids $\boldsymbol{b}$ in a sequential fashion.

W.l.o.g., we may assume that players are numbered so that $K = [k]$, $u_k(\boldsymbol{b}) > u_k(\boldsymbol{v})$, and there is an $m \in \{0 \dots k-1\}$ so that $b_i = -1$ for $i \in [m]$ and $b_i = b^\infty$ for $i \in \{m+1 \dots k\}$. Note that $m = 0$ is possible, but $m = k$ is not. For $i \in \{0 \dots k\}$, define $\boldsymbol{b}^i := (\boldsymbol{v}_{-[i]}, \boldsymbol{b}_{[i]})$. Clearly, $\boldsymbol{b}^0 = \boldsymbol{v}$ and $\boldsymbol{b}^k = \boldsymbol{b}$.

Our assumptions, together with VP, imply for all $i \in [m]$ that $0 = u_i(\boldsymbol{b}^i) = u_i(\boldsymbol{b}) = u_i(\boldsymbol{v})$. Now an inductive argument yields for all $i \in [m]$ that

$$\forall j \in \{i+1 \dots n\} : u_j(\boldsymbol{b}^i) \leq u_j(\boldsymbol{v}). \tag{5.3.5}$$

Clearly, this holds for the *base case* $i = 1$ due to WUNB. Now for the *induction step* $i \to (i+1)$, suppose the induction hypothesis (5.3.5) holds for i. Then $0 \overset{\text{VP}}{\leq} u_{i+1}(\boldsymbol{b}^i) \overset{\text{IH}}{\leq} u_{i+1}(\boldsymbol{v}) = 0$. Since also $u_{i+1}(\boldsymbol{b}^{i+1}) = 0$, we have for all $j \in \{i+2 \dots n\}$ that $u_j(\boldsymbol{b}^{i+1}) \leq u_j(\boldsymbol{b}^i) \overset{\text{IH}}{\leq} u_j(\boldsymbol{v})$, where the first inequality is again due to WUNB. Consequently, (5.3.5) holds also for $i+1$.

Let now

$$p := \max\{i \in \{m+1 \dots k\} \mid u_i(\boldsymbol{b}^i) < u_i(\boldsymbol{b}^{i-1})\} \tag{5.3.6}$$

be the "last" player who loses utility when adopting bid b_p. Since $u_k(\boldsymbol{b}) > u_k(\boldsymbol{v}) \overset{(5.3.5)}{\geq} u_k(\boldsymbol{b}^m)$ and due to Lemma 5.2.8 (i), we have that (5.3.6) is well-defined. Lemma 5.2.6 together with $q_p(\boldsymbol{b}^p) = 1$ implies

$$q_p(\boldsymbol{b}^{p-1}) = 0 \text{ and } u_p(\boldsymbol{b}^p) < u_p(\boldsymbol{b}^{p-1}) = 0 \overset{\text{VP}}{\leq} u_p(\boldsymbol{v}). \tag{5.3.7}$$

By definition of p and again by Lemma 5.2.8 (i), it must hold that $u_p(\boldsymbol{b} \mid b_p) \leq u_p(\boldsymbol{b}^p \mid b_p)$ because otherwise p would not have been maximal. Now recall that $q_p(\boldsymbol{b}) = 1$ due to $b_p = b^\infty$. Hence, $x_p(\boldsymbol{b}) \geq x_p(\boldsymbol{b}^p) \overset{(5.3.7)}{>} v_p$ and so $u_p(\boldsymbol{b}) < 0$. This is a contradiction to $p \in K$. □

Theorem 5.3.8. *Let $M = (q, x)$ be a separable 2-GSP cost-sharing mechanism. Then M is also GSP.*

Proof. We show for all $k \in \{3 \ldots n\}$ that if M is $(k-1)$-GSP, then M is also k-GSP. The proof of this statement is by contradiction. Let $\boldsymbol{v}$ be the true valuation vector, suppose $k \in \{3 \ldots n\}$, and assume there are a coalition $K \subseteq [n]$ with $|K| = k$ and a K-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ so that $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$.

Outline of contradiction Roughly speaking, we proceed as follows: Starting from the true valuations $\boldsymbol{v}$, we let the players in K adopt the bid vector $\boldsymbol{b}$ in a sequential fashion. This process is divided into two phases: First, all those players deviate who gain the service for $\boldsymbol{b}$ but not increased utility (compared to $\boldsymbol{v}$). In the second phase, all other players switch to the bids as in $\boldsymbol{b}$. It will turn out that the utilities in the second phase are essentially stagnant, so the crucial changes in utility have to occur during the first phase. This yields a contradiction, both when the first phase is short (at most one player) and when it is long.

Remaining Details We first note that our assumption implies $k < n$ and $\exists i \in [n] \setminus K : u_i(\boldsymbol{b}) < u_i(\boldsymbol{v})$. Otherwise, due to having a cost-sharing method, the grand coalition $[n]$ would also be successful by bidding $\boldsymbol{b}' \in \mathbb{R}^n$ defined by $b'_i = b^\infty$ if $q_i(\boldsymbol{b}) = 1$ and $b'_i = -1$ otherwise. This is a contradiction to Lemma 5.3.4. Moreover, we have for all $i \in K$ that

$$\exists j \in K \setminus i : u_j(\boldsymbol{v}_{-i}, b_i) > u_j(\boldsymbol{v}), \tag{5.3.9}$$

so $\forall i \in K : u_i(\boldsymbol{v}_{-i}, b_i) < u_i(\boldsymbol{v})$ due to Lemma 5.2.8 (i). Otherwise, if for some $i \in K$ there was no $j \in K \setminus i$ with $u_j(\boldsymbol{v}_{-i}, b_i) > u_j(\boldsymbol{v})$, then the coalition $K \setminus i$ could improve at $(\boldsymbol{v}_{-i}, b_i)$ by bidding as in $\boldsymbol{b}$, which contradicts $(k-1)$-GSP. Now Lemma 5.2.6 implies:

Claim 1. For each player $i \in K$, exactly one of the following two conditions holds:

i) $b_i \geq \theta_i(\boldsymbol{v}_{-i}) > v_i$, $q_i(\boldsymbol{v}) = 0$, $u_i(\boldsymbol{v}) = 0$

ii) $b_i \leq \theta_i(\boldsymbol{v}_{-i}) < v_i$, $q_i(\boldsymbol{v}) = 1$, $u_i(\boldsymbol{v}) > 0$, *and* $q_i(\boldsymbol{b}) = 1$

For notational convenience and w.l.o.g., we assume that players are numbered such that $K = [k]$, $u_n(\boldsymbol{b}) < u_n(\boldsymbol{v})$, and there is a $\lambda \in [k]$ with

- for all $i \in \{1 \dots \lambda - 1\} : q_i(\boldsymbol{v}) = 0$, $q_i(\boldsymbol{b}) = 1$, and $x_i(\boldsymbol{b}) = v_i$,
- for all $i \in \{\lambda \dots k\} : u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$ or $M_i(\boldsymbol{b}) = M_i(\boldsymbol{v})$.

This is not a restrictive assumption, because the case $q_i(\boldsymbol{v}) = 1$ but $q_i(\boldsymbol{b}) = 0$ cannot occur by Claim 1.

We now look at what happens if players adopt the bid vector $\boldsymbol{b}$ in a sequential fashion. As an abbreviating notation, we define for all $S \subseteq [n]$ the vector $\boldsymbol{b}^S := (\boldsymbol{v}_{-S}, \boldsymbol{b}_S)$. Roughly speaking, the following technical claim says that utilities stay fixed in the second phase, i.e., once the players in $\{1 \dots \lambda - 1\}$ (those who gain the service for $\boldsymbol{b}$ but not increased utility) have deviated to the bids as in $\boldsymbol{b}$.

Claim 2. Suppose S is a set of players with $\{1 \dots \lambda - 1\} \subseteq S \subseteq [k]$. If $|S| \geq 2$, then

$$\forall i \in [n] : u_i(\boldsymbol{b} \mid b_i^S) = u_i(\boldsymbol{b}^S \mid b_i^S). \tag{5.3.10}$$

Moreover, if $|S| = 1$ then

$$\forall i \in [n] : u_i(\boldsymbol{b} \mid b_i^S) \geq u_i(\boldsymbol{b}^S \mid b_i^S). \tag{5.3.11}$$

Proof (of Claim 2). We prove by induction on the size $s \in \{\max\{1, \lambda - 1\} \dots k\}$ of S that (5.3.10) and (5.3.11) hold (provided that the respective constraints on S are fulfilled). The *base case* $s = k$ holds trivially because it implies $S = [k]$ and hence $\boldsymbol{b}^S = \boldsymbol{b}$. We therefore only need to consider the induction step.

Induction Step ($s \to s - 1$) Assume that (5.3.10) is fulfilled for all sets S with $|S| \geq s$. Fix now some set S with $|S| = s - 1$. We show, by a sequence of substeps, that (5.3.10) and (5.3.11) hold. Let $\ell \in [k] \backslash S$ be a player and define $T := S \cup \ell$. Note that $\ell \in \{\lambda \dots k\}$.

i) If $|S| \geq 2$, then it holds that

$$\forall i \in [n] \setminus \ell : u_i(\boldsymbol{b}^T \mid b_i^S) \leq u_i(\boldsymbol{b}^S \mid b_i^S).$$

By way of contradiction, assume that $u_i(\boldsymbol{b}^T \mid b_i^S) > u_i(\boldsymbol{b}^S \mid b_i^S)$ for some $i \in [n] \setminus \ell$. Then $u_\ell(\boldsymbol{b}^T) < u_\ell(\boldsymbol{b}^S)$ by Lemma 5.2.8 (i). If $b_\ell > v_\ell$ then $q_\ell(\boldsymbol{b}^S) = 0$ by Lemma 5.2.6. On the other hand, if $b_\ell < v_\ell$, then $q_\ell(\boldsymbol{b}^S) \overset{\text{L5.2.6}}{=} 1 \overset{\text{C1}}{=} q(\boldsymbol{b})$ and $x_\ell(\boldsymbol{b}) \overset{\text{VP}}{\leq} b_\ell \overset{\text{P2.1.8}}{\leq} x_\ell(\boldsymbol{b}^S)$. In both cases, $u_\ell(\boldsymbol{b}) \geq u_\ell(\boldsymbol{b}^S)$.

Now, due to the induction hypothesis, we have for all $j \in [n] \setminus \ell$ that $u_j(\boldsymbol{b} \mid b_j^S) = u_j(\boldsymbol{b} \mid b_j^T) \overset{\text{IH}}{=} u_j(\boldsymbol{b}^T \mid b_j^T) = u_j(\boldsymbol{b}^T \mid b_j^S) \overset{\text{L5.2.8(v)}}{\geq} u_j(\boldsymbol{b}^S \mid b_j^S)$. For player i, the last inequality is strict. Hence, the coalition $K' := ([k] \setminus S) \cup i$ can manipulate and help i at $\boldsymbol{b}^S$, by bidding as in $\boldsymbol{b}$. This is a contradiction to $(k-1)$-GSP because $|K'| \leq k - |S| + 1 \leq k - 1$.

ii) If $|S| \geq 1$, then it also holds that

$$\forall i \in [n] \setminus \ell : u_i(\boldsymbol{b}^T \mid b_i^S) \geq u_i(\boldsymbol{b}^S \mid b_i^S).$$

Again, assume by way of contradiction that $u_i(\boldsymbol{b}^T \mid b_i^S) < u_i(\boldsymbol{b}^S \mid b_i^S)$ for some $i \in [n] \setminus \ell$. By Lemma 5.2.8 (ii) and (iii), we have $q_\ell(\boldsymbol{b}^T) \neq q_\ell(\boldsymbol{b}^S)$. Consider the two cases:

- Case $b_\ell < v_\ell$:

  Then, $q_\ell(\boldsymbol{b}^S) \overset{\text{P2.1.8}}{=} 1 \overset{\text{C1}}{=} q_\ell(\boldsymbol{b})$, $b_\ell \overset{\text{L5.2.8(ii)}}{<} x_\ell(\boldsymbol{b}^S)$, and $q_\ell(\boldsymbol{b}^T) = 0$. Since we also have $u_\ell(\boldsymbol{b} \mid b_\ell) \overset{\text{IH}}{=} u_\ell(\boldsymbol{b}^T \mid b_\ell) = 0$ by the induction hypothesis, it follows that $x_\ell(\boldsymbol{b}) = b_\ell$. Altogether, $u_\ell(\boldsymbol{b}) > u_\ell(\boldsymbol{b}^S)$.

  Now, there must be a player $j \in [k] \setminus T$ with

  $$0 \leq u_j(\boldsymbol{b}) < u_j(\boldsymbol{b}^S), \tag{5.3.12}$$

  so $q_j(\boldsymbol{b}^S) = 1$. Otherwise, the coalition $K' := [k] \setminus S$ could manipulate and help ℓ at $\boldsymbol{b}^S$, by bidding as in $\boldsymbol{b}$. This is a contradiction to $(k-1)$-GSP because $|K'| = k - |S| \leq k - 1$.

  Define $W := S \cup j$. Since $b_\ell < v_\ell$, we have that $u_\ell(\boldsymbol{b}^W) \overset{\text{IH}}{=} u_\ell(\boldsymbol{b}) \overset{\text{C1}}{>} 0$, so $M_\ell(\boldsymbol{b}^W) = M_\ell(\boldsymbol{b})$.

  Since $u_\ell(\boldsymbol{b}^W) = u_\ell(\boldsymbol{b}) > u_\ell(\boldsymbol{b}^S)$, it follows by Lemma 5.2.8 (i) that $u_j(\boldsymbol{b}^W) < u_j(\boldsymbol{b}^S)$. Together with $q_j(\boldsymbol{b}^S) \overset{(5.3.12)}{=} 1$, Lemma 5.2.6 (i) implies now $q_j(\boldsymbol{b}^W) = 0$ and $b_j < v_j$, so $q_j(\boldsymbol{b}) \overset{\text{C1}}{=} 1$. Then, since $u_j(\boldsymbol{b} \mid b_j) \overset{\text{IH}}{=} u_j(\boldsymbol{b}^W \mid b_j) = 0$, it must hold that $b_j = x_j(\boldsymbol{b}) \overset{(5.3.12)}{>} x_j(\boldsymbol{b}^S)$. This is a contradiction to SP because player j could improve at $\boldsymbol{b}^W$ by bidding v_j.

- Case $b_\ell > v_\ell$:

  Then, $q_\ell(\boldsymbol{b}^T) = 1$ and $v_\ell \leq x_\ell(\boldsymbol{b}^T) < b_\ell$ due to Lemma 5.2.8 (iii). Since $u_\ell(\boldsymbol{b} \mid b_\ell) \overset{\text{IH}}{=} u_\ell(\boldsymbol{b}^T \mid b_\ell) > 0$, we have $q_\ell(\boldsymbol{v}) \overset{\text{C1}}{=} 0$, $q_\ell(\boldsymbol{b}) = 1$ and $x_\ell(\boldsymbol{b}) = x_\ell(\boldsymbol{b}^T) \geq v_\ell$. However, this contradicts $\ell \in \{\lambda \ldots k\}$ because neither $u_\ell(\boldsymbol{b}) > u_\ell(\boldsymbol{v})$ nor $M_\ell(\boldsymbol{v}) = M_\ell(\boldsymbol{b})$.

iii) We can now complete the induction step and show that (5.3.10) and (5.3.11) hold. Consider first a player $i \in [n] \setminus \ell$. Then $u_i(\boldsymbol{b} \mid b_i^S) = u_i(\boldsymbol{b} \mid b_i^T) \overset{\text{IH}}{=} u_i(\boldsymbol{b}^T \mid b_i^T) = u_i(\boldsymbol{b}^T \mid b_i^S)$. Hence, if $|S| \geq 2$ then the previous substeps (i) and (ii) imply

$$u_i(\boldsymbol{b} \mid b_i^S) = u_i(\boldsymbol{b}^S \mid b_i^S).$$

If $|S| = 1$, then substep (ii) implies

$$u_i(\boldsymbol{b} \mid b_i^S) \geq u_i(\boldsymbol{b}^S \mid b_i^S).$$

Now consider player ℓ. It holds that $u_\ell(\boldsymbol{b}) \le u_\ell(\boldsymbol{b}^S)$ because otherwise $K' := [k] \setminus S$ could manipulate and help ℓ at $\boldsymbol{b}^S$ by bidding as in $\boldsymbol{b}$. Since $|K'| = k - |S| \le k - 1$, this would be a contradiction to $(k-1)$-GSP. Now, if the inequality was strict, i.e., if $0 \le u_\ell(\boldsymbol{b}) < u_\ell(\boldsymbol{b}^S)$, then $q_\ell(\boldsymbol{b}^S) = 1$ and $x_\ell(\boldsymbol{b}^S) < v_\ell$. Consider the two cases:

- Case $q_\ell(\boldsymbol{b}) = 0$:

  Then $b_\ell \overset{\text{C1}}{>} v_\ell$, so $M_\ell(\boldsymbol{b}^T) \overset{\text{P2.1.8}}{=} M_\ell(\boldsymbol{b}^S)$ and, in particular, $x_\ell(\boldsymbol{b}^T) = x_\ell(\boldsymbol{b}^S) < v_\ell < b_\ell$. Consequently, $u_\ell(\boldsymbol{b} \mid b_\ell) \overset{\text{IH}}{=} u_\ell(\boldsymbol{b}^T \mid b_\ell) > 0$. A contradiction to $q_\ell(\boldsymbol{b}) = 0$.

- Case $q_\ell(\boldsymbol{b}) = 1$:

  Then $x_\ell(\boldsymbol{b}^S) < x_\ell(\boldsymbol{b}) \overset{\text{VP}}{\le} b_\ell$, so $M_\ell(\boldsymbol{b}^T) \overset{\text{P2.1.8}}{=} M_\ell(\boldsymbol{b}^S)$. However, due to $u_\ell(\boldsymbol{b} \mid b_\ell) \overset{\text{IH}}{=} u_\ell(\boldsymbol{b}^T \mid b_\ell) > 0$, we have then $x_\ell(\boldsymbol{b}) = x_\ell(\boldsymbol{b}^T) = x_\ell(\boldsymbol{b}^S)$. A contradiction.

Hence, it must hold that $u_\ell(\boldsymbol{b}) = u_\ell(\boldsymbol{b}^S)$. This completes the proof of the claim. ■

We now consider the first phase and show that, as long as only players in $\{1 \dots \lambda - 1\}$ have deviated to the bids as in $\boldsymbol{b}$, there is always a player who strictly benefits when also the remaining players switch to $\boldsymbol{b}$.

Claim 3. Suppose $\ell \in \{1 \dots \lambda - 1\}$. *Then*

$$\exists i \in [\ell] : u_i(\boldsymbol{b} \mid b_i) > u_i(\boldsymbol{b}^{[\ell]} \mid b_i).$$

Proof (of Claim 3). For the *base case* $\ell = 1$, note that Claim 1 and the fact that $1 \in \{1 \dots \lambda - 1\}$ imply $q_1(\boldsymbol{v}) = 0$, $q_1(\boldsymbol{b}^1) = q_1(\boldsymbol{b}) = 1$, and $b_1 \ge x_1(\boldsymbol{b}^1) > v_1 = x_1(\boldsymbol{b})$; so $u_1(\boldsymbol{b} \mid b_1) > u_1(\boldsymbol{b}^1 \mid b_1)$.

For the *induction step* ($\ell \to \ell + 1$), assume the induction hypothesis holds for ℓ, i.e., there is some $i \in [\ell]$ with $u_i(\boldsymbol{b} \mid b_i) > u_i(\boldsymbol{b}^{[\ell]} \mid b_i)$. Now either

- $u_i(\boldsymbol{b} \mid b_i) > u_i(\boldsymbol{b}^{[\ell]} \mid b_i) \ge u_i(\boldsymbol{b}^{[\ell+1]} \mid b_i)$ or
- $u_i(\boldsymbol{b}^{[\ell+1]} \mid b_i) > u_i(\boldsymbol{b}^{[\ell]} \mid b_i)$, in which case we have $u_{\ell+1}(\boldsymbol{b}^{[\ell+1]}) < u_{\ell+1}(\boldsymbol{b}^{[\ell]})$ due to Lemma 5.2.8 (i) and therefore $q_{\ell+1}(\boldsymbol{b}^{[\ell+1]}) \ne q_{\ell+1}(\boldsymbol{b}^{[\ell]})$ due to Lemma 5.2.6. Together with $b_{\ell+1} \overset{\text{C1}}{>} v_{\ell+1}$ because of the fact that $\ell + 1 \in \{1 \dots \lambda - 1\}$, this implies $q_{\ell+1}(\boldsymbol{b}^{[\ell+1]}) \overset{\text{P2.1.8}}{=} 1$, $q_{\ell+1}(\boldsymbol{b}) \overset{\text{C1}}{=} 1$, $q_{\ell+1}(\boldsymbol{b}^{[\ell]}) = 0$, and $b_{\ell+1} \overset{\text{VP}}{\ge} x_{\ell+1}(\boldsymbol{b}^{[\ell+1]}) \overset{\text{L5.2.6}}{>} v_{\ell+1} = x_{\ell+1}(\boldsymbol{b})$. Consequently, $u_{\ell+1}(\boldsymbol{b} \mid b_{\ell+1}) > u_{\ell+1}(\boldsymbol{b}^{[\ell+1]} \mid b_{\ell+1})$. ■

We now have everything ready to disprove that $[k]$ is a successful coalition. If $\lambda \ge 3$, then Claim 2 implies that $\forall i \in [\lambda - 1] : u_i(\boldsymbol{b} \mid b_i) = u_i(\boldsymbol{b}^{[\lambda-1]} \mid b_i)$ whereas Claim 3 says $\exists i \in [\lambda - 1] : u_i(\boldsymbol{b} \mid b_i) > u_i(\boldsymbol{b}^{[\lambda-1]} \mid b_i)$. This is a contradiction.

On the other hand, if $\lambda \le 2$, then Claim 2 implies that $u_n(\boldsymbol{b}) \ge u_n(\boldsymbol{b}^1)$. Since also $u_{-1}(\boldsymbol{b}^1) \ge u_{-1}(\boldsymbol{v})$ by (5.3.9) and Lemma 5.2.8 (v), it holds that $u_n(\boldsymbol{b}) \ge u_n(\boldsymbol{v})$. Again a contradiction. This proves the theorem. □

Now recall that every GSP cost-sharing mechanism is separable. This has first been observed by Moulin [51] and is also the result of our more general Theorem 4.2.9. We thus obtain the following characterization:

Corollary 5.3.13. *Let M be a cost-sharing mechanism. Then, M is GSP if and only if it is 2-GSP and separable.*

5.4 Weak Group-Strategyproofness and Non-Bossiness

5.4.1 Separability and 2-WGSP Do Not Imply WGSP

A natural question is whether a statement similar to Theorem 5.3.8 holds also for WGSP. We give an example showing that this not the case. Consider the mechanism $M = (q, x)$ defined by Algorithm 5.2.

```
Input:   bid vector $\boldsymbol{b} \in \mathbb{R}^6$
Output:  service allocation $\boldsymbol{q} \in \{0,1\}^6$; cost shares $\boldsymbol{x} \in \mathbb{R}^6_{\geq 0}$
1: $\boldsymbol{q} := (0,\dots,0)$, $\boldsymbol{x} := (0,\dots,0)$
2: for all $i \in \{4,5,6\}$ with $b_i > 1$ or ($b_i = 1$ and $b_{1+(i-3 \bmod 3)} \geq 2$) do
3:     $q_i := 1$, $x_i := 1$
4: for all $i \in \{1,2,3\}$ with $b_i \geq 1 + q_{i+3}$ do
5:     $q_i := 1$, $x_i := 1 + q_{i+3}$
```

Algorithm 5.2: Separable mechanism that is 2-WGSP but not WGSP

The unique cost-sharing method $\xi : \{0,1\}^n \to \mathbb{R}^n_{\geq 0}$ of mechanism M is given by

$$\xi_i(\boldsymbol{s}) := \begin{cases} 0 & \text{if } s_i = 0 \\ 1 & \text{otherwise, if } i \in \{4,5,6\} \text{ or } (i \in \{1,2,3\} \text{ and } s_{i+3} = 0) \\ 2 & \text{otherwise, if } i \in \{1,2,3\} \text{ and } s_{i+3} = 1 \end{cases}$$

Note that the threshold property is fulfilled: For players $i = 1,2,3$ the threshold value is $\theta_i(\boldsymbol{b}_{-i}) = 2$ if $b_{i+3} > 1$ or ($b_{i+3} = 1$ and $b_{1+(i \bmod 3)} \geq 2$). It is $\theta_i(\boldsymbol{b}_{-i}) = 1$ otherwise. For players $i = 4,5,6$, the threshold value is constant, $\theta_i(\boldsymbol{b}_{-i}) = 1$.

The only players who could ever improve are 1, 2, and 3. However, no subset $S \subset \{1,2,3\}$ of size $|S| = 2$ can jointly improve because there is always a player $i \in S$ whose threshold value does not depend on $\boldsymbol{b}_S$. Hence, M is 2-WGSP. However, it is not 3-WGSP: Let $\boldsymbol{v} = (2,2,2,1,1,1)$ be the true valuations vector and consider $\boldsymbol{b} = (1,1,1,1,1,1)$. Then, $Q(\boldsymbol{v}) = \{1\dots6\}$ and $Q(\boldsymbol{b}) = \{1,2,3\}$, so $\{1,2,3\}$ is a successful coalition.

5.4.2 2-GSP Implies WGSP

We now completely drop separability and show that already 2-GSP alone implies WGSP. Hence, this is a case where a stronger notion of collusion resistance, yet only for players with limited communication abilities, implies a weaker collusion resistance against coalitions of arbitrary size.

Theorem 5.4.1. *Let $M = (q, x)$ be a 2-GSP cost-sharing mechanism. Then, M is also WGSP.*

Proof. The proof is by induction over the size $m \in [n]$ of successful coalitions. That is, we show for all $m \in [n]$ that M is m-WGSP. Clearly, the base cases $m = 1$ and $m = 2$ are fulfilled by definition. In the remainder of the proof we therefore show the induction step $m - 1 \to m$.

Induction step Assume M is $(m-1)$-WGSP. W.l.o.g., let players be numbered such that a successful m-coalition consists of the first m players, i.e., $[m]$. Due to the induction hypothesis, we have that $b_i \neq v_i$ for all $i \in [m]$ as otherwise there would be a successful coalition of size $(m-1)$. By way of contradiction, assume now that M is *not* m-WGSP, i.e., there are true valuations $\boldsymbol{v} \in \mathbb{R}^n$ and an $[m]$-variant $\boldsymbol{b}$ of $\boldsymbol{v}$ such that for all players $i \in [m]$ it holds that $u_i(\boldsymbol{b}) > u_i(\boldsymbol{v})$.

Outline of contradiction For $i \in [m]$, denote $B(i) := \{j \in [m] \mid u_j(\boldsymbol{v}_{-i}, b_i) \geq u_j(\boldsymbol{b})\}$, i.e., all players in $B(i)$ benefit when player i deviates to bid b_i. Define the binary relation $\triangleleft := \{(i,j) \in [m]^2 \mid j \in B(i)\}$. We will show that $\triangleleft$ is irreflexive, transitive, and serial (i.e., without maximum elements). That is,

$$\forall i \in [m] : i \not\triangleleft i \,, \tag{IRR}$$

$$\forall i,j,k \in [m] : i \triangleleft j \text{ and } j \triangleleft k \Longrightarrow i \triangleleft k \,, \tag{TRA}$$

$$\forall i \in [m] : \exists j \in [m] : i \triangleleft j \,. \tag{SER}$$

This is a contradiction. Intuitively, consider the directed graph with node set $[m]$ and edge set $\triangleleft$. There is an edge (i,j) whenever player i's deviation to b_i would make player j at least as happy as at $\boldsymbol{b}$. Now, irreflexivity (IRR) requires $([m], \triangleleft)$ to be free of self-loops. Yet, transitivity (TRA) and seriality (SER) imply that self-loops do exist.

Irreflexivity and seriality Obviously, (IRR) holds due to SP. Moreover, (SER) holds by the induction hypothesis: Otherwise, if for some $i \in [m]$ there was no $j \in [m] \setminus i$ with $u_j(\boldsymbol{v}_{-i}, b_i) \geq u_j(\boldsymbol{b})$, then the coalition $[m] \setminus i$ could improve at $(\boldsymbol{v}_{-i}, b_i)$ by bidding as in $\boldsymbol{b}$, which contradicts $(m-1)$-WGSP. The remaining main part of the proof is thus to show (TRA).

Transitivity Assume there are $i, j, k \in [m]$ so that $i \lhd j$ and $j \lhd k$. Let $\boldsymbol{v}^i$ be the i-variant of $\boldsymbol{v}$ with $v_i^i := \theta_i(\boldsymbol{v}_{-i})$. Define $\boldsymbol{v}^j$ correspondingly. Moreover, for $i, j \in [m]$, let $\boldsymbol{v}^{i,j}$ be the $\{i, j\}$-variant of $\boldsymbol{v}$ with $v_i^{i,j} := \theta_i(\boldsymbol{v}_{-i})$ and $v_j^{i,j} := \theta_j(\boldsymbol{v}_{-j})$.

- By definition of $\boldsymbol{v}^i$, it holds that
$$u_j(\boldsymbol{v}^i) \overset{\text{L5.2.8(iv)}}{\geq} u_j(\boldsymbol{v}_{-i}, b_i) \geq u_j(\boldsymbol{b}) > u_j(\boldsymbol{v}). \tag{5.4.2}$$
Consequently, it follows by Lemma 5.2.8 (i) that $u_i(\boldsymbol{v}_{-i}, b_i) < u_i(\boldsymbol{v})$ and $u_i(\boldsymbol{v}^i) < u_i(\boldsymbol{v})$. Then, Lemma 5.2.6 implies $q_i(\boldsymbol{v}) \neq q_i(\boldsymbol{v}^i) = q_i(\boldsymbol{v}_{-i}, b_i)$. By the threshold property, we then have
$$M_i(\boldsymbol{v}_{-i}, b_i) = M_i(\boldsymbol{v}^i) \quad \text{and} \quad u_i(\boldsymbol{v}_{-i}, b_i) = u_i(\boldsymbol{v}^i) < u_i(\boldsymbol{v}). \tag{5.4.3}$$
Now Lemma 5.2.7 implies
$$u_k(\boldsymbol{v}_{-i}, b_i) = u_k(\boldsymbol{v}^i). \tag{5.4.4}$$
- By (5.4.2), we have $q_j(\boldsymbol{v}^i) = 1$ and $\theta_j(\boldsymbol{v}^i_{-j}) = x_j(\boldsymbol{v}^i) < v_j$. By Lemma 5.2.8 (vi), we have $\theta_j(\boldsymbol{v}^i_{-j}) < \theta_j(\boldsymbol{v}_{-j}) = v_j^{i,j}$. So by the threshold property,
$$M_j(\boldsymbol{v}^{i,j}) = M_j(\boldsymbol{v}^i). \tag{5.4.5}$$
Then
$$u_i(\boldsymbol{v}^{i,j} \mid v_i^i) \overset{\text{L5.2.7}}{=} u_i(\boldsymbol{v}^i \mid v_i^i) \quad \text{and} \quad u_k(\boldsymbol{v}^{i,j}) \overset{\text{L5.2.7}}{=} u_k(\boldsymbol{v}^i). \tag{5.4.6}$$
- Due to $u_j(\boldsymbol{v}^{i,j}) \overset{(5.4.5)}{=} u_j(\boldsymbol{v}^i) \overset{(5.4.2)}{>} u_j(\boldsymbol{v})$ it must hold by 2-GSP that $u_i(\boldsymbol{v}^{i,j}) < u_i(\boldsymbol{v})$. Consider now the two cases:
  - Case $q_i(\boldsymbol{v}) = 1$:
    Then $v_i^{i,j} = \theta_i(\boldsymbol{v}_{-i}) = x_i(\boldsymbol{v}) < v_i$ due to Lemma 5.2.6 (i) with (5.4.3). Hence, 2-GSP implies $q_i(\boldsymbol{v}^{i,j}) = 0$, because otherwise $u_i(\boldsymbol{v}^{i,j}) \geq u_i(\boldsymbol{v})$ due to VP. We have $u_i(\boldsymbol{v}^j) \overset{\text{L5.2.8(v)}}{\geq} u_i(\boldsymbol{v}) > 0$, so $q_i(\boldsymbol{v}^j) = 1$ and $\theta_i(\boldsymbol{v}^j_{-i}) = x_i(\boldsymbol{v}^j) \leq x_i(\boldsymbol{v}) = \theta_i(\boldsymbol{v}_{-i})$. Now if the inequality was strict, then i could improve at $\boldsymbol{v}^{i,j}$ by bidding v_i, because $q_i(\boldsymbol{v}^{i,j}) = 0$ and $v_i^{i,j} = \theta_i(\boldsymbol{v}_{-i})$. Hence, $\theta_i(\boldsymbol{v}^{i,j}_{-i}) = \theta_i(\boldsymbol{v}^j_{-i}) = \theta_i(\boldsymbol{v}_{-i})$.
  - Case $q_i(\boldsymbol{v}) = 0$:
    Then $u_i(\boldsymbol{v}^{i,j}) < u_i(\boldsymbol{v}) = 0$ and thus $q_i(\boldsymbol{v}^{i,j}) = q_i(\boldsymbol{v}^i) \overset{(5.4.3)}{=} 1$. Consequently, $\theta_i(\boldsymbol{v}^{i,j}_{-i}) = x_i(\boldsymbol{v}^{i,j}) \overset{(5.4.6)}{=} x_i(\boldsymbol{v}^i) = \theta_i(\boldsymbol{v}_{-i})$.

  We have shown that $\theta_i(\boldsymbol{v}^{i,j}_{-i}) = \theta_i(\boldsymbol{v}_{-i}) = v_i^{i,j}$, so
$$u_k(\boldsymbol{v}^j) \overset{\text{L5.2.8(iv)}}{\leq} u_k(\boldsymbol{v}^{i,j}). \tag{5.4.7}$$

Putting everything together, we get

$$u_k(\boldsymbol{v}_{-i}, b_i) \overset{(5.4.4)}{=} u_k(\boldsymbol{v}^i) \overset{(5.4.6)}{=} u_k(\boldsymbol{v}^{i,j}) \overset{(5.4.7)}{\geq} u_k(\boldsymbol{v}^j) \overset{\text{L5.2.8(iv)}}{\geq} u_k(\boldsymbol{v}_{-j}, b_j) \geq u_k(\boldsymbol{b}).$$

Recall that the last inequality stems from our assumption that $j \lhd k$. Hence, $k \in B(i)$, i.e., $i \lhd k$. This completes the proof. □

5.4.3 Relationship Between Collusion-Resistance and Non-Bossiness Properties

Lemma 5.4.8. *Let $M = (q, x)$ be a SP and ONB cost-sharing mechanism. Then it is separable.*

Proof. Let $i \in [n]$. Suppose $\boldsymbol{b}, \boldsymbol{b}'$ are i-variants so that $b'_i = b^\infty$ if $q_i(\boldsymbol{b}) = 1$ and $b'_i = -1$ otherwise. The threshold property implies $M_i(\boldsymbol{b}) = M_i(\boldsymbol{b}')$, so $M(\boldsymbol{b}) = M(\boldsymbol{b}')$ by ONB. This argument can be used repeatedly: Let $\boldsymbol{b}^* \in \mathbb{R}^n$ be defined by $b^*_j := b^\infty$ if $q_j(\boldsymbol{b}) = 1$ and $b^*_j = -1$ otherwise. Then also $M(\boldsymbol{b}) = M(\boldsymbol{b}^*)$. □

Consequently, Theorem 5.3.8 can be seen as a generalization of Proposition 5.3.2 because the requirements of the latter (SP, ONB, and WUNB) imply 2-GSP and separability in a relatively straightforward manner.[3] The following example shows that Theorem 5.3.8 is *strictly* more general because ONB is not a necessary condition for GSP: Define mechanism $M = (q, x)$ by

$$q(\boldsymbol{b}) := \begin{cases} (1,1) & \text{if } (b_1 \geq 1 \text{ and } b_2 > 1) \text{ or } \boldsymbol{b} = (1,1) \\ (1,0) & \text{if } (b_1 \geq 1 \text{ and } b_2 \leq 1) \text{ and } \boldsymbol{b} \neq (1,1) \\ (0,1) & \text{if } b_1 < 1 \text{ and } b_2 > 1 \\ (0,0) & \text{if } b_1 < 1 \text{ and } b_2 \leq 1 \end{cases} \quad \text{and} \quad x(\boldsymbol{b}) := q(\boldsymbol{b}).$$

Obviously, the threshold value for both players is constantly 1, so neither of the two players could ever improve and M is GSP. However, the mechanism is not ONB because $M_1(1,1) = M_1(2,1)$ but $M_2(1,1) \neq M_2(2,1)$.

We conclude by stating another result by Mutuswami [55], which completes our overview of the various notions of non-manipulability and many of their implications (see Figure 5.2).

Proposition 5.4.9 (Mutuswami [55]). *Let M be a SP, ONB cost-sharing mechanism. Then it is also WGSP.*

[3] The fact that SP, ONB, and WUNB together imply also 2-GSP follows, of course, from Proposition 5.3.2. However, it can be easily seen directly: By way of contradiction, assume that $\{i, j\}$ is a GSP-successful coalition at $\boldsymbol{v}$ for some $\{i, j\}$-variant $\boldsymbol{b}$. W.l.o.g., let $u_j(\boldsymbol{b}) > u_j(\boldsymbol{v})$. Now, SP implies that $u_j(\boldsymbol{v}_{-i}, b_i) \geq u_j(\boldsymbol{b}) > u_j(\boldsymbol{v}) \geq 0$. Hence $q_j(\boldsymbol{b}) = q_j(\boldsymbol{v}_{-i}, b_i) = 1$, and $x_j(\boldsymbol{b}) = x_j(\boldsymbol{v}_{-i}, b_i)$ due to the threshold property. ONB implies $M(\boldsymbol{b}) = M(\boldsymbol{v}_{-i}, b_i)$. However, since SP and WUNB are fulfilled, Lemma 5.2.8 (i) implies $u_i(\boldsymbol{v}_{-i}, b_i) < u_i(\boldsymbol{v})$. This contradicts that player i is part of a successful coalition.

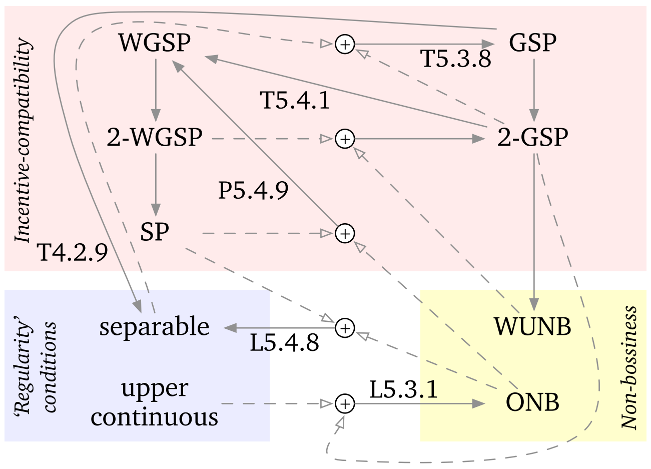


Figure 5.2: Overview of the various non-manipulability properties

5.5 Conclusion

GSP is a very strong axiom. It implies that players have full information of all other players' valuations and essentially unbounded ability to communicate and make bindings agreements. In particular, players would abandon a dominant strategy—telling the truth—even if they did not benefit from the deviation themselves. There are scenarios, in particular when the number of players is large, where these assumptions seem not appropriate.

In this chapter, we proposed relaxing GSP to k-GSP, which implies that the players' ability to coordinate deviations is limited to small coalitions (of size at most k). Somewhat surprisingly, however, we showed that already 2-GSP is equivalent to GSP once we require cost-sharing mechanisms to be separable—which is a very natural property and fulfilled by all known general techniques for the design of truthful cost-sharing mechanisms. Hence, our result gives some justification that GSP may, after all, still be desirable in several scenarios. Moreover, we proved that even without separability, 2-GSP implies WGSP. While, to the best of our knowledge, restrictions on coalition sizes have not been considered before in the cost-sharing literature, 2-GSP bears some resemblance to notions of non-bossiness. We also shed light on the relationship to these notions. Finally, various open problems remain.

- We believe that our results facilitate developing and understanding new GSP mechanisms. So, how can the fact that we only have to counter coalitions of size two be exploited for the open questions from Section 3.7?

- Characterizations of WGSP mechanisms are still needed. Are there "reasonable" $o(\log(n))$-WGSP mechanisms that are computable in polynomial time and perform better than acyclic mechanisms?

Chapter 6

Generalizing the Model

6.1 Overview of Contribution

In this last chapter, we generalize cost-sharing problems to a general-demand setting where each player may have demand for *multiple levels* of service. Correspondingly, a mechanism now has to output a *multiset* of served players. This is particularly useful in scenarios where multiple levels of service correspond to increased fault tolerance and a higher quality of service.

We show that the idea of Moulin mechanisms, i.e., serving the largest feasible set, can be generalized to serving the largest feasible multiset. However, cross-monotonic cost shares are not alone sufficient any more to imply GSP: Instead, constraints have to be imposed also on the *marginal* cost shares of the players. In fact, getting these constraints "right" is the major difficulty here. We therefore define *valid marginal cost-sharing methods* and thus obtain the first general technique for the design of general-demand cost-sharing mechanisms that are GSP.

6.2 General-Demand Cost Sharing

Formally, a *general-demand cost-sharing problem* is specified by the maximum service level $L \in \mathbb{N}$ available to each player and a cost function $C : [L]_0^n \to \mathbb{R}_{\geq 0}$. Each player $i \in [n]$ is characterized by a valuation vector $\boldsymbol{v}_i \in \mathbb{R}^L$ where $v_{i,l}$ indicates the *marginal valuation* of receiving level l additionally to levels $1, \dots, l-1$. For technical reasons that we will discuss later, we always require non-increasing marginal valuations $v_{i,1} \geq \cdots \geq v_{i,L}$. An outcome now consists of a *service-allocation vector* $\boldsymbol{q}^* \in [L]_0^n$, which represents the multiset of served players, and a vector of cost shares $\boldsymbol{x}^* \in \mathbb{R}^n$. The utility of a player $i \in [n]$ for outcome $(\boldsymbol{q}^*, \boldsymbol{x}^*)$ is

$$\sum_{l=1}^{q_i^*} v_{i,l} - x_i^* .$$

Note that a cost-sharing problem is binary-demand if $L = 1$.

A *general-demand* mechanism $M = (q, x)$ is a pair of functions $q : \mathbb{R}^{n \times L} \to [L]_0^n$ and $x : \mathbb{R}^{n \times L} \to \mathbb{R}^n$. Truthfulness, budget-balance, and economic efficiency are all generalized in the obvious way. A *marginal cost-sharing method* $\chi : [L]_0^n \to \mathbb{R}_{\geq 0}^{n \times L}$ associates service allocations to their marginal cost shares, where for all service allocations $\boldsymbol{s} \in [L]_0^n$, all players $i \in [n]$ and all service levels $l > s_i$, we require that $\xi_{i,l}(\boldsymbol{s}) = 0$. Now given a marginal cost-sharing method χ and a bid vector $\boldsymbol{b} \in \mathbb{R}^{n \times L}$, we say a service allocation $\boldsymbol{s}$ is $\boldsymbol{b}$-feasible if for all $i \in [n]$ and all $l \in [s_i]$ it holds that $\chi_{i,l}(\boldsymbol{s}) \le b_{i,l}$. Trivially, the empty service allocation $(0,\dots,0)$ is always $\boldsymbol{b}$-feasible.

6.2.1 Generalized Moulin Mechanisms

The following notation will be convenient. For any two service-allocation vectors $\boldsymbol{s}, \boldsymbol{t} \in [L]_0^n$, we define $\boldsymbol{s} \vee \boldsymbol{t} := (\max\{s_i, t_i\})_{i \in [n]}$.[1] Moreover, for $l \in [L]_0$, we define $\boldsymbol{s}^{\le l} := (\min\{s_i, l\})_{i \in [n]}$.

Now, we define a property that will take the role of cross-monotonicity in the binary-demand setting.

Definition 6.2.1. *A marginal cost-sharing method χ is*

- cross-monotonic *if for all service-allocation vectors $\boldsymbol{s} \le \boldsymbol{t}$, all players i, and all service levels $l \le s_i$ it holds that $\chi_{i,l}(\boldsymbol{s}) \ge \chi_{i,l}(\boldsymbol{t})$;*
- non-decreasing *if for all service-allocation vectors $\boldsymbol{t}$ and all players i it holds that $\chi_{i,1}(\boldsymbol{t}) \le \chi_{i,2}(\boldsymbol{t}) \le \dots \le \chi_{i,t_i}(\boldsymbol{t})$.*
- level-restricted *if for all service-allocation vectors $\boldsymbol{t}$, all players i, and all service levels l it holds that $\chi_{i,l}(\boldsymbol{t}) = \chi_{i,l}(\boldsymbol{t}^{\le l})$;*

We say χ is valid *if it satisfies all of the above three properties.*

We briefly comment on the new properties “non-decreasing” and “level-restricted”: Non-decreasing marginal cost shares imply that the more service levels a player gets, the more expensive each additional level becomes. This is in contrast to the marginal valuations, which are non-increasing. It will turn out that this reversed growth is crucial for ensuring that generalized Moulin mechanisms are GSP.

Level-restrictedness has an interesting implication: It ensures that one could iteratively invoke binary-demand cost-sharing mechanisms: In each iteration $i = 1,\dots,L$, determine an allocation of service level i among those players that have also received service level $i-1$. By level-restrictedness, a later iteration cannot change the marginal cost shares assigned in previous iterations.

Definition 6.2.2. *A mechanism M is a* generalized Moulin mechanism *if it has a valid marginal cost-sharing method and it associates all bid vectors $\boldsymbol{b} \in \mathbb{R}^{n \times L}$ to the maximal $\boldsymbol{b}$-feasible service allocation.*

We remark that generalized Moulin mechanisms are well-defined because there is always a *unique* maximal $\boldsymbol{b}$-feasible allocation. This can be verified as follows: Assume that both $\boldsymbol{s}$ and $\boldsymbol{t}$ are maximal $\boldsymbol{b}$-feasible allocations. Define $\boldsymbol{r} := \boldsymbol{s} \vee \boldsymbol{t}$. Then, $\boldsymbol{r}$ is also $\boldsymbol{b}$-feasible

[1] This is the usual notation when considering $[L]^n$ together with the partial order “$\le$” as a lattice.

because M has cross-monotonic cost shares. Moreover, it holds that $\boldsymbol{r} \geq \boldsymbol{s}$ and $\boldsymbol{r} \geq \boldsymbol{t}$. Now, since both $\boldsymbol{s}$ and $\boldsymbol{t}$ are maximal by assumption, it hence follows that $\boldsymbol{r} = \boldsymbol{s} = \boldsymbol{t}$.

Equivalently to Definition 6.2.2, $M = (q, x)$ is a generalized Moulin mechanism if and only if there is a marginal cost-sharing method χ so that for all $\boldsymbol{b} \in \mathbb{R}^{n \times L}$ it holds that

$$q(\boldsymbol{b}) = \max\left\{\boldsymbol{s} \in [L]_0^n \mid \forall i \in [n] : \forall l \in [s_i] : b_{i,l} \geq \chi_{i,l}(\boldsymbol{s})\right\} \quad \text{and}$$
$$x(\boldsymbol{b}) = (\textstyle\sum_{l=1}^{L} \chi_{i,l}(q(\boldsymbol{b})))_{i \in [n]}.$$

Theorem 6.2.3. *Generalized Moulin mechanisms are GSP.*

Proof. Let $M = (q, x)$ be a generalized Moulin mechanism. Let $\boldsymbol{v}$ contain the true valuations, $K \subseteq [n]$ be a non-empty coalition, and $\boldsymbol{b}$ be a K-variant of $\boldsymbol{v}$ with $u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$. We will show that then $u_K(\boldsymbol{b}) = u_K(\boldsymbol{v})$.

Let $\boldsymbol{s} := q(\boldsymbol{v})$ and $\boldsymbol{t} := q(\boldsymbol{b})$. If $\boldsymbol{t} \leq \boldsymbol{s}$, then cross-monotonicity implies $u(\boldsymbol{b}) \leq u(\boldsymbol{v})$ and we are done. Hence, by way of contradiction, assume that $\exists i \in [n] : t_i > s_i$. Define level $l := \min\{s_i \mid i \in [n] \text{ and } t_i > s_i\}$ and $A := \{i \in [n] \mid t_i > l \text{ and } l = s_i\}$. Define $\boldsymbol{r}$ by

$$r_i := \begin{cases} l+1 & \text{if } i \in A \\ s_i & \text{otherwise}. \end{cases}$$

Note that for all players $i \in A$ and for all levels $k \in [l]$, we have $\boldsymbol{s}^{\leq k} = \boldsymbol{r}^{\leq k} = (\boldsymbol{s} \vee \boldsymbol{t})^{\leq k}$ and thus $\chi_{i,k}(\boldsymbol{s}) = \chi_{i,k}(\boldsymbol{r}) = \chi_{i,k}(\boldsymbol{s} \vee \boldsymbol{t})$ because χ is level-restricted. Similarly, $\chi_{i,l+1}(\boldsymbol{r}) = \chi_{i,l+1}(\boldsymbol{s} \vee \boldsymbol{t})$.

We now show for all $i \in A$ that $v_{i,l+1} \geq \chi_{i,l+1}(\boldsymbol{t})$. By way of contradiction, assume this is not the case. Then, due to non-increasing marginal utilities, non-decreasing cost shares, and cross-monotonicity, there is an $i \in A$ so that for all $k \in \{l+1 \dots t_i\} : v_{i,k} < \chi_{i,k}(\boldsymbol{t}) \leq b_{i,k}$. Consequently, $i \in K$ and

$$\begin{aligned} u_i(\boldsymbol{b}) = \sum_{k=1}^{t_i}\big(v_{i,k} - \chi_{i,k}(\boldsymbol{t})\big) &< \sum_{k=1}^{l}\big(v_{i,k} - \chi_{i,k}(\boldsymbol{t})\big) && \text{as explained} \\ &\leq \sum_{k=1}^{l}\big(v_{i,k} - \chi_{i,k}(\boldsymbol{s} \vee \boldsymbol{t})\big) && \text{due to cross-monotonicity} \\ &= \sum_{k=1}^{l}\big(v_{i,k} - \chi_{i,k}(\boldsymbol{s})\big) = u_i(\boldsymbol{v}) && \text{as explained}. \end{aligned}$$

This is a contradiction to $i \in K$.

Now, we have shown for all $i \in A$ that $v_{i,l+1} \geq \chi_{i,l+1}(\boldsymbol{t}) \geq \chi_{i,l+1}(\boldsymbol{s} \vee \boldsymbol{t}) = \chi_{i,l+1}(\boldsymbol{r})$. Therefore, by cross-monotonicity, $\boldsymbol{r}$ is $\boldsymbol{v}$-feasible. This is again a contradiction because $\boldsymbol{r} > \boldsymbol{s}$, but $\boldsymbol{s}$ is the maximal $\boldsymbol{v}$-feasible allocation due to the definition of generalized Moulin mechanisms. □

Similar to the binary-demand case, the outcome of generalized Moulin mechanisms can be computed efficiently. Specifically, using the same arguments as for Lemma 2.2.6,

```
Input:  marginal cost-sharing method $\chi : [L]_0^n \to \mathbb{R}_{\geq 0}^{n \times L}$, bid vector $\boldsymbol{b} \in \mathbb{R}^{n \times L}$
Output: service allocation $\boldsymbol{q} \in [L]_0^n$, cost distribution $\boldsymbol{x} \in \mathbb{R}_{\geq 0}^n$
1: $\boldsymbol{q} := (L, \dots, L)$
2: while there is a player $i$ with $q_i > 0$ and $b_{i,q_i} < \chi_{i,q_i}(\boldsymbol{q})$ do
3:     $q_i := q_i - 1$
4: $\boldsymbol{x} := (\sum_{l=1}^{L} \chi_{i,l}(\boldsymbol{q}))_{i \in [n]}$
```

Algorithm 6.1: Generalized Moulin mechanisms

it is easy to see that for every marginal cost-sharing method χ and for all bid vectors $\boldsymbol{b}$, Algorithm 6.1 computes the outcome of the respective generalized Moulin mechanism.

In our SAGT'08 paper [7], we design valid marginal cost-sharing methods for the *fault-tolerant facility location problem*. An instance of this problem is specified as for the usual facility location problem (see Section 2.3.3); yet in addiction, we are also given the number of facilities each customer wants to be connected to.

Theorem 6.2.4 (Bleischwitz and Schoppmann [7]). *For sharing the cost induced by fault-tolerant facility location, there is a generalized Moulin mechanism that guarantees* $3L$*-BB and* $(3L \cdot (1 + H_n))$*-EFF. Its outcome can be computed in polynomial time.*

We remark that the only other truthful cost-sharing mechanisms for this problem are due to Mehta et al. [50]. However, their mechanisms are only WGSP, and depending on the size of n and L, both the budget balance and economic efficiency of our mechanisms are better.

6.3 Conclusion

Obviously, generalizing the cost-sharing model to general demand opens the door to numerous new problems—essentially all research questions addressed in this thesis are also interesting in the generalized setting.

The requirements of generalized Moulin mechanisms are somewhat limiting: While the motivation for cross-monotonicity is similar to the binary-demand case [74, 51, 52], also all the arguments against it (see Section 1.3) still hold. Similarly, the properties "non-decreasing" and "level-restricted" may seem implausible in various practical settings. Finally, the requirement that marginal valuations are non-increasing is a strong restriction: E.g., players cannot express that they need to be connected to at least two facilities—possibly due to security concerns—and that only one connection would be almost useless to them. We therefore state as open problems:

- Are there other techniques for designing GSP general-demand cost-sharing mechanisms? In particular, are there techniques when marginal valuations may be increasing?

- What are lower bounds on the performance of generalized Moulin mechanisms, similar to the results by Immorlica et al. [36]? Are there general lower bounds for GSP mechanisms, independent of the design technique?

Fault-tolerant facility location is the only problem for which both acyclic mechanisms and generalized Moulin mechanisms are known. Even though acyclic mechanisms are only WGSP, there is no clear indication that they allow for significantly improved performance.

- Can the approximation guarantees for general-demand cost-sharing be improved, possibly by sacrificing some collusion resistance?
- Would relaxing GSP to 2-GSP help for general-demand cost sharing?

Bibliography

Note: Each entry is followed by a list of the pages from which there was a reference.

[1] S. Aland, D. Dumrauf, M. Gairing, B. Monien, and F. Schoppmann. Exact price of anarchy for polynomial congestion games. In *Proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS'06)*, volume 3884 of *LNCS*, pages 218–229, 2006. DOI: `10.1007/11672142_17`. Extended version available at `http://wwwcs.upb.de/cs/ag-monien/PERSONAL/FSCHOPP/pdf/ExactPoA_extended.pdf`. → 15

[2] N. Andelman, M. Feldman, and Y. Mansour. Strong price of anarchy. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'07)*, pages 189–198. Society for Industrial and Applied Mathematics, 2007. → 13

[3] A. Archer, J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Approximation and collusion in multicast cost sharing. *Games and Economic Behaviour*, 47(1): 36–71, 2004. DOI: `10.1016/S0899-8256(03)00176-3`. → 11

[4] R. J. Aumann. Acceptable points in general cooperative *n*-person games. In *Contributions to the theory of games, Vol. IV*, Annals of Mathematics Studies, No. 40, pages 287–324. Princeton University Press, Princeton, N.J., 1959. → 5

[5] 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 (CIAC'06)*, volume 3998 of *LNCS*, pages 175–186, 2006. DOI: `10.1007/11758471_19`. → 10, 11

[6] Y. Bleischwitz and F. Schoppmann. New efficiency results for makespan cost sharing. *Information Processing Letters*, 107(2):64–70, July 2008. DOI: `10.1016/j.ipl.2008.01.005`. → 10, 11, 15

[7] Y. Bleischwitz and F. Schoppmann. Group-strategyproof cost sharing for metric fault tolerant facility location. In *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)*, volume 4997 of *LNCS*, pages 350–361, 2008. DOI: `10.1007/978-3-540-79309-0_31`. → 15, 104

[8] Y. Bleischwitz, B. Monien, and F. Schoppmann. To be or not to be (served). In *Proceedings of the 3rd International Workshop on Internet and Network Economics (WINE'07)*, volume 4858 of *LNCS*, pages 515–528,

2007. DOI: 10.1007/978-3-540-77105-0_55. Extended version available at http://wwwcs.upb.de/cs/ag-monien/PERSONAL/FSCHOPP/pdf/ToBeOrNotToBeServed_extended.pdf. → 11, 14, 53

[9] Y. Bleischwitz, B. Monien, F. Schoppmann, and K. Tiemann. The power of two prices: Beyond cross-monotonicity. In *Proceedings of the 32nd International Symposium on Mathematical Foundations of Computer Science (MFCS'07)*, volume 4708 of *LNCS*, pages 657–668, 2007. DOI: 10.1007/978-3-540-74456-6_58. Extended version available at http://wwwcs.upb.de/cs/ag-monien/PERSONAL/FSCHOPP/pdf/LexicographicMaximization_extended.pdf. → 11, 14, 29

[10] J. Brenner and G. Schäfer. Cost sharing methods for makespan and completion time scheduling. In *Proceedings of the 24th International Symposium on Theoretical Aspects of Computer Science (STACS'07)*, volume 4393 of *LNCS*, pages 670–681, 2007. DOI: 10.1007/978-3-540-70918-3_57. → 10, 11, 28, 84

[11] J. Brenner and G. Schäfer. Singleton acyclic mechanisms and their applications to scheduling problems. In *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)*, volume 4997 of *LNCS*, pages 315–326, 2008. DOI: 10.1007/978-3-540-79309-0_28. → 11, 12, 54, 66, 71, 72, 83, 84

[12] P. Brucker. *Scheduling Algorithms*. Springer Verlag, 5th edition, 2007. DOI: 10.1007/978-3-540-69516-5. → 24, 25, 82

[13] S. Chawla, T. Roughgarden, and M. Sundararajan. Optimal cost-sharing mechanisms for Steiner forest problems. In *Proceedings of the 2nd International Workshop on Internet and Network Economics (WINE'06)*, volume 4286 of *LNCS*, pages 112–123, 2006. DOI: 10.1007/11944874_11. → 11

[14] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971. DOI: 10.1007/BF01726210. → 5, 22

[15] 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. DOI: 10.1016/0885-064X(87)90009-4. → 26, 79

[16] R. Deb and L. Razzolini. Voluntary cost sharing for an excludable public project. *Mathematical Social Sciences*, 37(2):123–138, 1999. DOI: 10.1016/S0165-4896(98)00026-2. → 20

[17] R. Deb and L. Razzolini. Auction-like mechanisms for pricing excludable public goods. *Journal of Economic Theory*, 88(2):340–368, 1999. DOI: 10.1006/jeth.1999.2603. → 11, 12

[18] 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. DOI: 10.1016/j.dss.2004.08.004. → 13

[19] S. Dobzinski, A. Mehta, T. Roughgarden, and M. Sundararajan. Is Shapley cost sharing optimal? In *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)*, volume 4997 of *LNCS*, pages 327–336, 2008. DOI: 10.1007/978-3-540-79309-0_29. → 11

[20] G. Dósa. The tight bound of first fit decreasing bin-packing algorithm is FFD(I) ≤ (11/9) OPT(I) + 6/9. In *Proceedings of the First International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies (ESCAPE'07)*, volume 4614 of *LNCS*, pages 1–11, 2007. DOI: 10.1007/978-3-540-74450-4_1. → 26, 79

[21] B. Dutta. The egalitarian solution and reduced game properties in convex games. *International Journal of Game Theory*, 19(2):153–169, June 1990. DOI: 10.1007/BF01761074. → 11, 66

[22] B. Dutta and D. Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57(3):615–635, 1989. URL: http://www.jstor.org/stable/1911055. → 6, 9, 40, 54, 62, 66

[23] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, August 2001. DOI: 10.1006/jcss.2001.1754. → 11

[24] J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Hardness results for multicast cost sharing. *Theoretical Computer Science*, 304(1–3):215–236, July 2003. DOI: 10.1016/S0304-3975(03)00085-9. → 5, 11

[25] M. Gairing and F. Schoppmann. Total latency in singleton congestion games. In *Proceedings of the 3rd International Workshop on Internet and Network Economics (WINE'07)*, volume 4858 of *LNCS*, pages 381–387, 2007. DOI: 10.1007/978-3-540-77105-0_42. → 15

[26] M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979. → 14, 26

[27] R. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969. URL: http://www.jstor.org/stable/2099572. → 25, 41, 77

[28] R. Graham, E. Lawler, J.-K. Lenstra, and A. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979. DOI: 10.1016/S0167-5060(08)70356-X. → 25

[29] J. Green and J.-J. Laffont. Characterizations of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica*, 45(2):427–438, 1977. URL: http://www.jstor.org/stable/1911219. → 5, 22

[30] T. Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973. URL: `http://www.jstor.org/stable/1914085`. → 5, 22

[31] 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 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'07)*, pages 1153–1162, 2007. → 11

[32] A. Gupta, A. Srinivasan, and É. Tardos. Cost-sharing mechanisms for network design. *Algorithmica*, 50(1):98–119, January 2008. DOI: `10.1007/s00453-007-9065-y`. → 11

[33] D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, 1997. → 14, 26

[34] D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems: Theoretical and practical results. *Journal of the ACM*, 34(1): 144–162, 1987. DOI: `10.1145/7531.7535`. → 25, 80

[35] D. S. Hochbaum and D. B. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988. DOI: `10.1137/0217033`. → 25

[36] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. *ACM Transactions on Algorithms*, 4(2):1–25, May 2008. DOI: `10.1145/1361192.1361201`. → 6, 7, 11, 12, 18, 45, 105

[37] K. Jain and V. Vazirani. Applications of approximate algorithms to cooperative games. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC'01)*, pages 364–372, 2001. DOI: `10.1145/380752.380825`. → 11

[38] K. Jain and V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC'02)*, pages 313–321, 2002. DOI: `10.1145/509907.509956`. → 11

[39] R. Juarez. Collusion-proof cost sharing mechanisms (draft). Available at `http://www2.hawaii.edu/~rubenj/groupstrategyproof_indiff3.pdf`, November 2007. → 53, 56

[40] T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow-time problem. *SIAM Journal on Computing*, 15(4):1119–1129, 1986. DOI: `10.1137/0215081`. → 83

[41] K. Kent and D. Skorin-Kapov. Population monotonic cost allocation on MSTs. In *Proceedings of the 6th International Conference on Operations Research (KOI'96)*, pages 43–48, Rovinj, Croatia, 1996. Croatian Operational Research Society. → 10

[42] J. Könemann, S. Leonardi, G. Schäfer, and S. van Zwam. A group-strategyproof cost sharing mechanism for the Steiner forest game. *SIAM Journal on Computing*, 37(5):1319–1341, January 2008. DOI: `10.1137/050646408`. → 11, 54

[43] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th International Symposium on Theoretical Aspects of Computer Science (STACS'99)*, volume 1563 of *LNCS*, pages 404–413. Springer Verlag, 1999. DOI: `10.1007/3-540-49116-3_38`. → 1

[44] A. Kovács. New approximation bounds for LPT scheduling. *Algorithmica*, 2008. DOI: `10.1007/s00453-008-9224-9`. Year refers to online publishing date. → 25

[45] J.-K. Lenstra, A. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:343–362, 1977. DOI: `10.1016/S0167-5060(08)70743-X`. → 25

[46] S. Leonardi and G. Schäfer. Cross-monotonic cost sharing methods for connected facility location games. *Theoretical Computer Science*, 326(1–3):431–442, 2004. DOI: `10.1016/j.tcs.2004.07.033`. → 11

[47] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995. → 14

[48] M. Mavronicolas, B. Monien, V. G. Papadopoulou, and F. Schoppmann. Voronoi games on cycle graphs. In *Proceedings of the 33rd International Symposium on Mathematical Foundations of Computer Science (MFCS'08)*, volume 5162 of *LNCS*, 2008. DOI: `10.1007/978-3-540-85238-4_41`. → 15

[49] V. Mazalov, B. Monien, F. Schoppmann, and K. Tiemann. Wardrop equilibria and price of stability for bottleneck games with splittable traffic. In *Proceedings of the 2nd International Workshop on Internet and Network Economics (WINE'06)*, volume 4286 of *LNCS*, pages 331–342, 2006. DOI: `10.1007/11944874_30`. → 15

[50] A. Mehta, T. Roughgarden, and M. Sundararajan. Beyond Moulin mechanisms. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC'07)*, pages 1–10, 2007. DOI: `10.1145/1250910.1250912`. Full version available at `http://theory.stanford.edu/~tim/papers/bmm.pdf`. → 7, 11, 13, 22, 54, 66, 67, 68, 104

[51] H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999. DOI: `10.1007/s003550050145`. → 6, 11, 13, 19, 22, 23, 41, 44, 48, 60, 70, 95, 104

[52] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001. DOI: `10.1007/PL00004200`. → 4, 5, 11, 22, 23, 104

[53] F. D. Murgolo. Anomalous behavior in bin packing algorithms. *Discrete Applied Mathematics*, 21(3):229–243, October 1988. DOI: `10.1016/0166-218X(88)90069-8`. → 72, 79

[54] S. Mutuswami. Strategyproof cost sharing of a binary good and the egalitarian solution. *Mathematical Social Sciences*, 48(3):271–280, 2004. DOI: `10.1016/j.mathsocsci.2004.03.002`. → 11

[55] S. Mutuswami. Strategyproofness, non-bossiness and group strategyproofness in a cost sharing model. *Economics Letters*, 89(1):83–88, 2005. DOI: `10.1016/j.econlet.2005.05.012`. → 12, 19, 85, 89, 90, 98

[56] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behaviour*, 35:166–196, 2001. DOI: `10.1006/game.1999.0790`. → 1, 3, 13

[57] N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007. → 14

[58] M. J. Osborne and A. Rubinstein. *A course in game theory*. MIT Press, 1994. → 6

[59] M. Pál and É. Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proceedings of the 44th Annual Symposium on Foundations of Computer Science (FOCS'03)*, pages 584–593, 2003. DOI: `10.1109/SFCS.2003.1238231`. → 11

[60] C. Papadimitriou. Algorithms, games, and the Internet. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC'01)*, pages 749–753, 2001. DOI: `10.1145/380752.380883`. → 1, 2

[61] D. C. Parkes. *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, May 2001. URL: `http://www.eecs.harvard.edu/~parkes/diss.html`. → 14

[62] P. Penna and C. Ventre. More powerful and simpler cost-sharing methods. In *Proceedings of the 2nd International Workshop on Approximation and Online Algorithms (WAOA'04)*, volume 3351 of *LNCS*, pages 97–110, 2005. DOI: `10.1007/b106130`. → 7

[63] T. Roughgarden. Algorithmic game theory: Some greatest hits and future directions. In *Proceedings of the 5th IFIP International Conference on Theoretical Computer Science*, pages 21–42, 2008. DOI: `10.1007/978-0-387-09680-3_2`. Revised version available at `http://theory.stanford.edu/~tim/papers/tcs08.pdf`. → 13

[64] T. Roughgarden and M. Sundararajan. New trade-offs in cost-sharing mechanisms. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC'06)*, pages 79–88, 2006. DOI: `10.1145/1132516.1132528`. Extended version available at `http://theory.stanford.edu/~tim/papers/trade.pdf`. → 5, 6, 11, 43

[65] T. Roughgarden and M. Sundararajan. Approximately efficient cost-sharing mechanisms. *arXiv report*, June 2006. URL: `http://arxiv.org/abs/cs/0606127`. → 6, 11, 54

[66] M. A. Satterthwaite and H. Sonnenschein. Strategy-proof allocation mechanisms at differentiable points. *Review of Economic Studies*, 48(4):587–597, October 1981. URL: `http://www.jstor.org/stable/2297198`. → 12, 19

[67] F. Schoppmann. The power of small coalitions in cost sharing. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE'08)*, volume 5385 of *LNCS*, pages 665–674, 2008. DOI: `10.1007/978-3-540-92185-1_72`. → 14

[68] A. S. Schulz and N. A. Uhan. Encouraging cooperation in sharing supermodular costs. In *APPROX '07/RANDOM '07: Proceedings of the 10th International Workshop on Approximation and the 11th International Workshop on Randomization, and Combinatorial Optimization*, pages 271–285. Springer Verlag, 2007. DOI: `10.1007/978-3-540-74208-1_20`. → 71, 83, 84

[69] J. Schummer. Manipulation through bribes. *Journal of Economic Theory*, 91(2):180–198, 2000. DOI: `10.1006/jeth.1999.2618`. → 13, 86, 87

[70] S. Serizawa. Pairwise strategy-proofness and self-enforcing manipulation. *Social Choice and Welfare*, 26(2):305–331, 2006. DOI: `10.1007/s00355-006-0099-x`. → 12, 86

[71] L. S. Shapley. A value for *n*-person games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the theory of games, Vol. II*, Annals of Mathematics Studies, No. 28, pages 307–317. Princeton University Press, Princeton, N.J., 1953. → 6, 40

[72] S. Shenker. Some technical results on continuity, strategy-proofness, and related strategic concepts. Available at `ftp://parcftp.parc.xerox.com/pub/net-research/str.ps`, 1993. → 12

[73] W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956. DOI: `10.1002/nav.3800030106`. → 83

[74] Y. Sprumont. Population monotonic allocation schemes for cooperative games with transferable utility. *Games and Economic Behavior*, 2(4):378–394, 1990. DOI: `10.1016/0899-8256(90)90006-G`. → 11, 104

[75] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961. URL: `http://www.jstor.org/stable/2977633`. → 2, 5, 22

Index

Note: **Bold** page numbers refer to (formal) definitions.