The Power of Lexicographic Maximization: Beyond Cross-Monotonicity*

Yvonne Bleischwitz
yvonneb@upb.de

Burkhard Monien
bm@upb.de

Florian Schoppmann
fschopp@upb.de

Karsten Tiemann
tiemann@upb.de

Thursday 20th August, 2009

Department of Computer Science, Electrical Engineering and Mathematics
University of Paderborn, Fürstenallee 11, 33102 Paderborn, Germany

Abstract In a cost-sharing problem, finitely many players have an unknown preference for some non-rivalrous but excludable good (service). The task is to determine which set of players Q to serve and how to distribute the incurred cost $C(Q)$. Typical constraints of cost-sharing problems demand for mechanisms that elicit truthful bids, balance the budget (i.e., recover the incurred cost with the prices charged), and are economically efficient (i.e., trade off the service cost and the excluded players' valuations as good as possible).

The only known general technique for the design of cost-sharing mechanisms that are truthful in the strong sense, i.e., resilient also against manipulation by coalitions (group-strategyproof, GSP), is due to Moulin (1999). Unfortunately, there are several natural cost-sharing problems for which any Moulin mechanism inevitably suffers poor budget-balance and economic efficiency.

In this work, we make first important steps towards new general design techniques for GSP mechanisms: We devise a novel family of polynomial-time computable mechanisms that, for the class of subadditive symmetric cost functions, achieve provably better budget balance than any Moulin mechanism. We apply our findings to the problem of sharing the makespan cost when scheduling jobs on parallel related machines. Towards understanding the limitations imposed by GSP itself, we establish an impossibility result stating that for more than three players, exactly budget-balanced GSP mechanisms do not exist in general, even if the cost function is symmetric.

* This work was partially supported by the IST Program of the European Union under contract number IST-15964 (AEOLUS). The first, third, and fourth authors were supported by a fellowship by the International Graduate School of Dynamic Intelligent Systems, Paderborn. A preliminary version [6] of this work appeared in the Proceedings of the 32nd International Symposium on Mathematical Foundation of Computer Science, August 2007.

1 Introduction

In a *cost-sharing problem*, 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; 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 [32] and their references. 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.

1.1 Design Techniques for Cost-Sharing Mechanisms

Truthfulness As is common in the literature on cost sharing, we assume that 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 *strategyproof* (SP), requires that no player can improve his utility by false bidding when all other bids are kept fixed. Moreover, several concepts of resistance against coordinated manipulation are known in the literature: A mechanism is called *group-strategyproof* (GSP) if any defection by 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.

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* [41, 10, 18]. 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 [17] revealed already in the 1970's that under general assumptions[1], the VCG mechanisms are the only class of SP mechanisms with these properties.

[1] The set of possible bids is equal to the set of possible valuations, and utilities are quasi-linear (see Section 2.2).

Unfortunately, VCG mechanisms are not resistant against collusion and fail in general to provide any guarantees for cost recovery, even when ignoring computational complexity [32]. 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. [15] 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 [36] 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.

GSP Mechanisms Due to the unsuitability of VCG mechanisms for cost-sharing problems there is great need for other general design techniques, of which, however, only few are known up to the present day. Essentially the only one to obtain GSP mechanisms is due to Moulin [31]. Its main ingredient is *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. [22] and Roughgarden and Sundararajan [36, 37] 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. [22] and Penna and Ventre [35] gave a family of cost-sharing mechanisms that are GSP only if voluntary non-participation is not an option players have.[2] In this work, 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 work.

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.

[2] Formally, these mechanisms do not satisfy *strong consumer sovereignty* (see Section 2.2).

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 A general technique for the design of WGSP mechanisms, called *acyclic mechanisms*, is due to Mehta et al. [29]. 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.

Bleischwitz et al. [5] gave a special family of acyclic mechanisms that for the broad class of subadditive cost-sharing problems provide exact budget-balance and optimal (cf. Dobzinski et al. [11]) asymptotic efficiency. For some classes of problems, their mechanisms are computable in polynomial time; however, this does not necessarily hold in general.

1.2 Our Results

We devise a novel technique for the design of GSP mechanisms based on the following idea: Given fixed cost shares $\xi_i(Q)$, 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 Q 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; yet, this is only well-defined for cross-monotonic cost shares.

While we have to leave open an exact characterization of when cost-sharing methods induce GSP mechanisms under lexicographic maximization, we do give a sufficient condition by defining *valid symmetric* cost-sharing methods. Here, a player's cost-share only depends on his rank within the set of the served players and on the size of that set. We call any such mechanism a *symmetric mechanism*.

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 is never more costly than the sum of the two stand-alone costs. For cost functions that are both symmetric and subadditive, we give a family of valid symmetric 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 1.3). 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 above). 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 [7] 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.

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.

1.3 Related Work

Applications Besides general design techniques and obtaining characterization results (see 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 to combinatorial optimization problems, including the minimum spanning tree [25, 23, 24], Steiner tree [23, 36, 37, 29], fixed tree multicast [14, 15, 2], facility location [34, 28, 37, 22, 29], rent-or-buy-network design [34, 37, 20], Steiner forest [9, 19, 26], edge/vertex/set cover [22, 29], and minimum makespan or minimum sum of completion times when scheduling parallel machines [3, 7, 6, 4, 5, 8]. In this list, three results [29, 5, 8] fit into the framework of acyclic mechanisms, all other results develop Moulin mechanisms with cross-monotonic cost shares.

Sharing Makespan Costs 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 [3]. 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 [7] 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 [3]. This is tight up to a factor of 2, since d is a lower bound [3].

Characterizing the Limitations of GSP A complete characterization of the impact of submodular costs on GSP and 1-BB was given by Moulin [31]: 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 [40], and the egalitarian solution [12].

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. [22]: If a GSP mechanism is upper-continuous then it has cross-monotonic cost shares. Consequently, it is a Moulin mechanism. Our symmetric mechanisms are the first known general technique for GSP mechanisms that do not satisfy upper-continuity.

Relaxing GSP As noted before in Section 1.1, relaxing GSP to WGSP allows for much improved performance with regard to the other desirable properties of a cost-sharing mechanism. Roughly speaking, when not taking the computational complexity into account, WGSP does not impair the achievable performance more than SP does—a property that does not hold true for GSP. Hence, when designing WGSP mechanisms, the only remaining intricacy lies in achieving polynomial-time computability. In the case of several scheduling problems, this issue can be dealt with, though: Bleischwitz et al. [5] gave polynomial-time acyclic mechanisms for sharing the makespan cost and achieve 2-BB and $O(\log n)$-EFF even when neither jobs nor machines are identical.

While these results may suggest sacrificing GSP in favor of WGSP, a recent result by Schoppmann [38] reinforces that GSP may still be desirable in several circumstances. He considered relaxing GSP to resistance only against coalitions of size two ("2-GSP")—yet, it is shown that "natural" mechanisms[3] are resistant against smalls coalitions only if they are also GSP (i.e., resistant against arbitrarily large coalitions).

The Minimum-Makespan Problem Machine scheduling is one of the fundamental areas in combinatorial optimization. Since most multiprocessor problems are NP-hard, research has long focused on finding approximation algorithms (see, e.g., Hochbaum [21]). Almost a decade ago, Nisan and Ronen [33] proposed a new paradigmatic way of studying optimization problems, based on the idea that part of the input data is held by selfish agents. One of the problems they particularly considered is minimizing the makespan when scheduling unrelated machines: Here, the machines (and not the jobs as in our cost-sharing model) are controlled by selfish agents and thus have to be given monetary incentives to truthfully reveal the true time needed to process each job. The case of related machines was later studied by Archer and Tardos [1]. For a survey on this research area, we refer to Lavi [27].

1.4 Roadmap

All preliminaries, including our notational conventions and a formal definition of the cost-sharing model, are given in Section 2. We develop our novel technique for GSP mechanisms in Section 3 by giving a mathematical definition of symmetric mechanisms and an efficient algorithm for computing them. Afterwards, we devise concrete symmetric mechanisms for the case of symmetric subadditive costs in Section 4. As an application of our findings, we use our technique for sharing the makespan cost when scheduling jobs on parallel machines in Section 5. The impact of symmetry of costs on GSP and 1-BB is studied in Section 6.

[3] Technically: Whenever payments only depend on the set of served players but not directly on the bids, i.e., there is a cost-sharing method ξ so that always $x(\boldsymbol{b}) = \xi(Q(\boldsymbol{b}))$. See Definition 3 in Section 2.

2 The Model

2.1 Notation

For $n, m \in \mathbb{N}_0$, let $\{n \dots m\} := \{n, n+1, \dots, m\}$ and $[n] := \{1 \dots n\}$. Given a vector $\boldsymbol{x}$, we denote its components by $\boldsymbol{x} = (x_1, x_2, \dots)$. Two vectors $\boldsymbol{x}, \boldsymbol{y}$ of the same dimension are called *K-variants* if $x_i = y_i$ for all $i \notin K$. In this case, we write $\boldsymbol{y} = (\boldsymbol{x}_{-K}, \boldsymbol{y}_K)$. If $K = \{i\}$, then $\boldsymbol{x}$ and $\boldsymbol{y}$ are called *i-variants*. We then only write $\boldsymbol{y} = (\boldsymbol{x}_{-i}, y_i)$. We write $\boldsymbol{x} \leq \boldsymbol{y}$ to denote that in every component, $\boldsymbol{x}$ is no larger than $\boldsymbol{y}$. Moreover, $\boldsymbol{x} < \boldsymbol{y} :\Longleftrightarrow (\boldsymbol{x} \leq \boldsymbol{y}$ and $\boldsymbol{x} \neq \boldsymbol{y})$.

We use the symbol $\preceq$ for the binary relation "is lexicographically no larger than" (and $\prec$, $\succ$, $\succeq$ correspondingly). The maximum element of $\preceq$ on X is denoted $\operatorname{lex\,max} X$.

Given a finite set $S \subseteq \mathbb{N}$ 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, $MIN_k\, S := \{i \in S \mid \operatorname{rank}(i, S) \leq k\}$ is defined as the set of the k smallest elements in S, and $MAX_k\, S$ is likewise defined as the set of the k largest elements in S.

2.2 Cost-Sharing Problems and Mechanisms

A *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 costs. 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 $v_i \in \mathbb{R}$. *Utilities* are *quasi-linear*, i.e., player i's utility for outcome $(Q, \boldsymbol{x})$ is $v_i \cdot q_i - x_i$ where $q_i \in \{0, 1\}$, $q_i = 1 :\Leftrightarrow i \in Q$.

Definition 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* $(Q(\boldsymbol{b}), x(\boldsymbol{b}))$.

Given a fixed cost-sharing mechanism $M = (Q, x)$, we denote player i's utility by $u_i(\boldsymbol{b}) := v_i \cdot q_i(\boldsymbol{b}) - x_i(\boldsymbol{b})$. Unless otherwise noted, we will always require three standard axiomatic properties in this work:

- *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 $\mathfrak{b}^\infty \in \mathbb{R}_{\geq 0}$ such that if $b_i \geq \mathfrak{b}^\infty$ 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 [22].

2.3 Truthfulness

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 work, as well as in many related works on cost sharing, a stronger notion is required that also ensures resistance against coordinated manipulation.

Definition 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 $u_K(\boldsymbol{b}) > u_K(\boldsymbol{v})$.*

An interesting property of GSP is that it implies a separability condition in that cost shares only depend on the set of served players and not on the bids [31].

Definition 3. *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 $\xi_i(S) = 0$.*

Now technically, the aforementioned separability condition means that for any GSP mechanism $M = (Q, x)$ there is a cost-sharing method ξ so that $x = \xi \circ Q$, i.e., it always holds that $x(\boldsymbol{b}) = \xi(Q(\boldsymbol{b}))$.

For completeness (despite not the focus of this work), we remark that a *weaker* notion of collusion resistance is obtained by *strengthening* the requirements for successful coalitions: A mechanism M is *weakly group-strategyproof*, if for all coalitions $K \subseteq [n]$, all true valuations $\boldsymbol{v} \in \mathbb{R}^n$, and all their K-variants $\boldsymbol{b}$ it holds that $u_i(\boldsymbol{b}) \leq u_i(\boldsymbol{v})$ for at least one $i \in K$.

2.4 Budget Balance and Economic Efficiency

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

Definition 4. *A mechanism $M = (Q, x)$ is* β-budget-balanced (β-BB) with regard to actual costs C' and optimal costs C *if for all combinations of bids $\boldsymbol{b}$ 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 $C(\emptyset) = 0$.

As a measure for economic efficiency, the incurred cost and the rejected players' valuations should be traded off as good as possible:

Definition 5. *A Mechanism $M = (Q, x)$ is* α-efficient (α-EFF) with regard to actual costs C' and optimal costs C *if for all combinations of bids $\boldsymbol{b}$ it holds that*

$$C'(Q(\boldsymbol{b})) + \sum_{i \notin Q(\boldsymbol{b})} \overline{b_i} \leq \alpha \cdot \min_{P \subseteq [n]} \left\{ C(P) + \sum_{i \notin P} \overline{b_i} \right\},$$

where $\alpha \geq 1$ is a constant (independent of $\boldsymbol{b}$) and $\overline{b_i} := \max\{b_i, 0\}$.

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

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").

2.5 Special Cost Functions

In many cases, costs exhibit a special structure that can be exploited when designing cost-sharing mechanisms. In this work, 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 $S, T \subseteq [n] : C(S \cup T) \leq C(S) + C(T)$.
- *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 $S, T \subseteq [n]$: $C(S \cup T) + C(S \cap T) \geq C(S) + C(T)$.

3 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 $\mathcal{F}(\boldsymbol{b})$. 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, $Q(\boldsymbol{b}) = \max \mathcal{F}(\boldsymbol{b})$, i.e., $Q(\boldsymbol{b})$ is always the greatest element of $\mathcal{F}(\boldsymbol{b})$ 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 $Q(\boldsymbol{b}) = \max(\arg\max_{S \in \mathcal{F}(\boldsymbol{b})} \{P(\boldsymbol{b}, S)\})$. The crucial idea is now the following generalization.

Definition 6. *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, $Q(\boldsymbol{b}) \in \arg\operatorname{lex}\max_{S \in \mathcal{F}(\boldsymbol{b})} \{P(\boldsymbol{b}, S)\}$.*

In order to be consistent with the preliminary version of this work [6], we take the lexicographic order with *reversed significance*: For instance, in this work $(1,2) \succ (2,1)$. 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, $\arg\operatorname{lex}\max_{S\in\mathcal{F}(\boldsymbol{b})}\{P(\boldsymbol{b},S)\}$ contains more than one feasible set.

3.1 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 7. *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 8. *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., $\xi_\lambda(p) < \xi_\lambda(p-1) \implies d_p \leq \lambda$.

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

p	l_p	d_p	$\xi_1(p), \ldots, \xi_p(p)$
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	$\xi_1(p), \ldots, \xi_p(p)$
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 $\{1 \ldots 5\}$, whereas the right method has the two segments $\{1 \ldots 4\}$, $\{5\}$. 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. $\Diamond$

Definition 9. *Let ξ be a valid symmetric cost-sharing method. The precedence mechanism uniquely defined by $Q(\boldsymbol{b}) := \operatorname{lex\,max}(\arg\operatorname{lex\,max}_{S \in \mathcal{F}(\boldsymbol{b})}\{P(\boldsymbol{b}, S)\})$ 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 1. *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 $\xi_\lambda(p) \geq \xi_{\lambda+1}(p+1)$.*

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

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

Lemma 2. *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 mechanisms 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, $\xi_j(S') \leq \xi_j(S)$ due to Lemma 1.

- 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 $\xi_j(S') \leq \xi_j(S) \leq b_j$. Hence, S' is $\boldsymbol{b}$-feasible and $P(\boldsymbol{b}, S') \succ P(\boldsymbol{b}, S)$. This completes the proof. $\square$

Theorem 1. *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 $u_K(\boldsymbol{b}) \geq u_K(\boldsymbol{v})$. 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

$$p := \min\left\{ r \in \{t' + s'' \dots |T \cup S''|\} \;\middle|\; \xi_{[t']}(r) \leq \xi_{[t']}(|T|) \right\}. \tag{3.1}$$

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

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

Note that T', S'', and $MIN_{r-t'-s''}\left(T'' \setminus S''\right)$ 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) $|T| \leq z$, $v_i > \xi_i(T) \geq l$, and $|T \cup S''| \leq z$

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, $|T| \leq z$ and we also have $v_i > \xi_i(T) \geq l$.

Now, all players in $T \cup S$ have a true valuation of at least $l \geq l_{z+1}$. 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) $y \leq s''$ and $y \leq |T|$

Define $S''_+ := \{j \in S'' \mid v_j > l\}$. By way of contradiction, assume $y > |S''_+|$. Define $R' := S''_+ \cup i \cup MAX_{y-|S''_+|-1}\left(S \setminus (S''_+ \cup i)\right)$. 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, $y \leq |S''_+| \leq s''$.

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 $R'' := T \cup MAX_{y-|T|}\left(S''_+ \setminus T\right)$. Then $|R''| = y$ because $y - |T| \leq |S''_+ \setminus T|$. 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 (3.1), 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

$$\begin{aligned} \xi_{t'}(p) &\leq \xi_{t'}(|T|) && \text{due to (3.1)} \\ &= \xi_i(T) < \xi_{s'}(|S|) && \text{as explained} \\ &\leq \xi_{t'}(|S|) && \text{due to (V1)}\,. \end{aligned}$$

  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 1.

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

  Then $\xi_{[t']}(p) < \xi_{[t']}(p-1)$ by (3.1) 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 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 1. We start by giving some intuition. Afterwards, we provide a formal verification of correctness.

Input: valid symmetric cost-sharing method ξ, 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: $I := \{i \in Q \mid b_i = l_{|Q|}\}$; $H := \emptyset$; $p := \max\big(\{r \in [|Q|] \mid d_r = 0\} \cup \{0\}\big)$
4: **while** $p < |Q \setminus I|$ **do**
5: $\quad j := \min(Q \setminus H)$; $I := I \setminus j$
6: $\quad$ **if** $b_j \geq \xi_j(Q)$ **then** $H := H \cup j$; $p := \max\{r \in [|Q|] \mid d_r = |H|\}$
7: $\quad$ **else** $Q := Q \setminus j$
8: $Q := Q \setminus \mathit{MIN}_{|Q|-p} I$; $\boldsymbol{x} := \xi(Q)$

Algorithm 1: Symmetric mechanisms

Algorithm 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.

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. *Suppose Algorithm 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 $\{y \dots z\}$ 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: "$\forall r \in \{p+1 \dots |Q|\} : |H| < d_r$". 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 2. *Let ξ be a valid symmetric cost-sharing method. Then, for all bid vectors $\boldsymbol{b}$, Algorithm 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 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 (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. □

4 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 10. *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.

4.1 Computing Two-Price Cost-Sharing Forms

We compute 2P-CSFs with Algorithm 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.

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** $p := 2, \ldots, n$ **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 2: Two-price cost-sharing forms

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).

Lemma 4. *Suppose Algorithm 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 $(l_p < l_{p-1} \implies d_p = 0)$

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

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

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 $(h_p < h_{p-1} \implies d_p \leq 1)$

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. *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 2 efficiently computes a valid and β-BB 2P-CSF $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$.*

Proof. Due to Lemma 4, 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

$$h_p \geq (\beta^2 - \beta) \cdot c(s)\,. \tag{4.1}$$

Obviously, (4.1) 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 (4.1).

Next, we show

$$c(p) \leq 2 \cdot c(s)\,. \tag{4.2}$$

Let $t := \min(\{r \in \{p+1 \dots n\} \mid \beta \cdot \frac{c(r)}{r} \leq l_p\} \cup \{n+1\})$ 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, $s \leq y < p' < p$, $d_{y+1} = 1$, $d_{p'+1} = 2$, and $h_{y+1} \geq h_p = h_{p'}$. We show

  $$2 \cdot c(s) > h_{y+1} + y \cdot l_{y+1}\,. \tag{4.3}$$

  By way of contradiction, assume $2 \cdot c(s) \leq h_{y+1} + y \cdot l_{y+1}$. Then, $c(p'+1) \leq c(p) \leq 2 \cdot c(s) \leq h_{y+1} + y \cdot l_{y+1}$ due to (4.2). 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, (4.3) holds.

  As a next step, we show

  $$h_p = (\beta^2 - \beta) \cdot c(s)\,. \tag{4.4}$$

  Recall (4.1) and assume, by way of contradiction, that $h_p > (\beta^2 - \beta) \cdot c(s)$. Since $h_{y+1} \geq h_p$, line 10 must have evaluated to false for cardinality $(y+1)$. Since the other two inequalities of line 10 are fulfilled according to (4.3) 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 $\beta^2 \cdot c(s) = \beta^2 \cdot c(s) - \beta \cdot c(s) + s \cdot l_{y+1} \leq (\beta^2 - \beta) \cdot c(s) + y \cdot l_{y+1} < c(y+1)$. 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 (4.3), which then proves (4.4)

  Now, we get as lower bound

  $$\begin{aligned}
  \gamma(p) &= 2h_p + (p-2) \cdot l_p \\
  &= 2h_p + (p-2) \cdot \beta \cdot \frac{c(s)}{s} && \text{due to the definition of } s \\
  &\geq (2\beta^2 - \beta) \cdot c(s) && \text{due to (4.4) and } p - 2 \geq s \\
  &\geq 2 \cdot c(s) && \text{due to } \beta \geq \tfrac{\sqrt{17}+1}{4} \\
  &\geq c(p) && \text{due to (4.2)}\,.
  \end{aligned}$$

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

$$\begin{aligned}
\gamma(p) &= 2h_p + (p-2) \cdot l_p \\
&= h_p - l_p + h_p + (p-1) \cdot l_p \\
&< h_p + c(p) && \text{due to line 8} \\
&= (\beta - 1) \cdot \beta \cdot c(s) + c(p) && \text{due to (4.4)} \\
&< (\beta - 1) \cdot c(p) + c(p) && \text{as explained} \\
&= \beta \cdot c(p)\,.
\end{aligned}$$

□

4.2 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 4. *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
$$\begin{aligned} h_{y+1} &\leq \alpha \cdot c(r) - c(r-1) = \alpha \cdot \beta - \left(\beta - \frac{\beta-1}{\beta^l}\right) \\ &< \alpha \cdot \beta - (\beta - \varepsilon) = \alpha \cdot (\beta - 1) < \beta^2 - \beta\,. \end{aligned}$$
□

4.3 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 [39] and the egalitarian solution [13].

Lemma 5. *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 6. *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 has a unique cost-sharing method [31]. 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 $v_i := \mathfrak{b}^\infty$ for all other players i. It has to hold that $Q(\boldsymbol{v}_{-1}, \mathfrak{b}^\infty) = [n] \setminus 2$. Otherwise, if $Q(\boldsymbol{v}_{-1}, \mathfrak{b}^\infty) = [n]$, player 2 could help all other players by bidding -1. Correspondingly, $Q(\boldsymbol{v}_{-2}, \mathfrak{b}^\infty) = [n] \setminus 1$.

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 $\mathfrak{b}^\infty$. Correspondingly, if $Q(\boldsymbol{v}) = [n] \setminus 2$, then player 2 could improve by bidding $\mathfrak{b}^\infty$. A contradiction to GSP. □

5 Applications

Suppose a computing center with a large cluster of parallel machines wants to offer processing times to potential customers. First, it asks for reports of the maximum payments that each customer would be willing to contribute in order for having his jobs processed. Only then, the computing center uses a commonly known cost-sharing mechanism to determine both the set of served customers and their payments. In addition, it computes a cost-efficient schedule for the accepted jobs.

The motivation here is as follows: Deciding on prices only after receiving *binding bids* by customers enables the computing center to determine an outcome that both ensures recovery of its own cost as well as competitive prices in that its surplus is always relatively small. In fact, if the computing center is a public institution, this *budget-balance* might even be a legal requirement.

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. The processing time of player i's job (in short: job i) is denoted by $p_i \in \mathbb{N}$. All processing times are publicly known and not part of the input of the mechanism. The speed of machine j is denoted by $s_j \in \mathbb{N}$. We say jobs are *identical* if $p_1 = \ldots p_n = 1$; machines are *identical* if $s_1 = \cdots = s_m = 1$. The *completion time* of a machine $j \in [m]$ is the sum of the processing times of the jobs assigned to j, divided by j's speed. A schedule for the set Q is a denoted $\boldsymbol{a} \in [m]^Q$, its makespan cost is the maximum completion time. Hence, the optimal cost C are defined as

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

It is a simple observation that C is a subadditive function. Moreover, if jobs are identical, then C is symmetric and $C(Q)$ can be computed in time $O(n \cdot \log m)$ using the LPT (longest processing time first) algorithm [16]. It processes the jobs in decreasing order and assigns each job to the machine on which its completion time will be smallest.

If jobs are not identical, C is not symmetric anymore and an optimal allocation is in general NP-hard to compute. However, to keep finding an allocation computationally tractable, we want algorithms to be polynomial-time computable in the size of the scheduling instance plus the players' bids. We thus assume that the service provider uses an approximation algorithm

ALG. For any algorithm ALG that computes feasible solutions, we define the approximation cost function as C_{ALG}. The objective is now to find mechanisms that are β-BB with regard to the actual approximation cost C_{ALG} and the optimal cost C.

5.1 Makespan Minimization with Identical Jobs

Theorem 5. *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. 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 2 can be done in time $O(n)$. Finally, computing the outcome of the symmetric mechanism with Algorithm 1 takes time $O(n^2)$. Summing up yields the desired result. □

Lemma 7. *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 2 and dividing all cost shares by $\frac{\sqrt{17}+1}{4}$. □

Lemma 8. *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 2 (or as in Lemma 7) 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 $p \in \{m+1 \ldots 2m\}$. Let $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$ as in Lemma 7 (as noted before, this is the same 2P-CSF as the one computed by Algorithm 2 but divided by $\frac{\sqrt{17}+1}{4}$). We have:

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

Suppose the true valuations are $\boldsymbol{v} = (\frac{1}{m} - \varepsilon, \frac{2}{m} - \varepsilon, \ldots, 1 - \varepsilon, \frac{1}{m} + \varepsilon, \ldots, \frac{1}{m} + \varepsilon)$ for some fixed $\varepsilon > 0$. Let $M = (Q, x)$ be the symmetric mechanism defined by $(n, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{l})$. Then $Q(\boldsymbol{v}) = \{m+1 \ldots m\}$, $C(Q(\boldsymbol{v})) = 1$, 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. □

5.2 Makespan Minimization with Non-Identical Jobs

Theorem 6. *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 $c : [n] \to \mathbb{R}_{\geq 0}$ 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 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 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 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 $C(Q)$:

$$
\begin{aligned}
C'(Q) &\leq \sum_{\phi \in P} \phi \cdot c(|Q_\phi|) && \text{because merging is subadditive} \\
&\leq \sum_{\phi \in P} \sum_{i \in Q_\phi} x_i^\phi && \text{by Theorem 5} \\
&\leq \sum_{\phi \in P} \frac{\sqrt{17}+1}{4} \cdot C(Q_\phi) && \text{by Theorem 5} \\
&\leq d \cdot \frac{\sqrt{17}+1}{4} \cdot C(Q)\,. &&
\end{aligned}
$$

□

6 Characterizing Symmetry and 1-BB

6.1 An Impossibility Result

Theorem 7. *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 7, we need the subsequent Propositions 1, 2, and 3, together with Lemma 9.

Proposition 1 (Moulin [31]). *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 1 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:

$$\xi_j(S \cup \{j,k\}) > \xi_j(S \cup \{j\}) \Longrightarrow \xi_k(S \cup \{k\}) = \xi_k(S \cup \{j,k\})\,. \tag{6.1}$$

Proposition 2 (Immorlica et al. [22]). *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 (Immorlica et al. [22]). *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 9. *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	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
$\mu(U)$	$(0,0,0)$	$(1,0,0)$	$(0,1,0)$	$(0,0,1)$	$(2,1,0)$	$(2,0,1)$	$(0,2,1)$	$(3,2,1)$
$\nu(U)$								$(2,3,1)$

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 (6.1), 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 (6.1) 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 $Q(2,\mathfrak{b}^\infty,\mathfrak{b}^\infty) = \{2,3\}$. A contradiction to GSP.

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

Assume first that $\xi_3\{1,2,3\} > 1$. By (6.1) 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 $\boldsymbol{v} = (2,\mathfrak{b}^\infty,\mathfrak{b}^\infty)$. Then player 1 is indifferent to getting the service and he could help player 3 by bidding -1 or help player 2 by bidding $\mathfrak{b}^\infty$, 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 (6.1)), we have that $\xi_2\{1,2,3\} \leq 1$ in this case. Now consider the true valuation vector $\boldsymbol{v} = (\mathfrak{b}^\infty, \xi_2\{1,2,3\}, v_3)$ 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 $\mathfrak{b}^\infty$ (using Proposition 2) 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 $\boldsymbol{v} = (\mathfrak{b}^\infty, v_2, \mathfrak{b}^\infty)$ with $v_2 \in (\xi_2\{1,2,3\}, 2)$, all players are served for $\boldsymbol{v}$ (Proposition 2). However, player 1 would be better off by bidding some $b_1 \in (2, \xi_1\{1,2,3\})$ instead of $\mathfrak{b}^\infty$, 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 (6.1), $\xi_1\{1,2,3\} = \xi_1\{1,3\} = 2$, 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 7). 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 9 hold for all subset $U \subset [4]$ with $|U| = 3$.

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

Let $\triangleleft$ denote the binary relation $\triangleleft := \{(a,b) \mid \xi_a\{a,b\} \geq \xi_b\{a,b\}\} \subseteq [4]^2$. By Lemma 9, for any 3-element-subset $U \subset [4]$, the restriction of $\triangleleft$ 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 9, 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 (6.1) 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 (6.1) 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 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 $\boldsymbol{v} = (\mathfrak{b}^\infty, 2, \frac{3}{2}, \mathfrak{b}^\infty)$: He could bid $\mathfrak{b}^\infty$ or -1 because $Q(\boldsymbol{v}_{-2}, \mathfrak{b}^\infty) = \{1,2,3,4\}$ 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 $\boldsymbol{v} = (\mathfrak{b}^\infty, 2, \mathfrak{b}^\infty, \mathfrak{b}^\infty)$, by bidding $\mathfrak{b}^\infty$ 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 $\boldsymbol{v} = (2, \mathfrak{b}^\infty, \frac{3}{2}, \mathfrak{b}^\infty)$: He could bid $\mathfrak{b}^\infty$ or -1 because $Q(\boldsymbol{v}_{-1}, \mathfrak{b}^\infty) = \{1,2,3,4\}$ 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 $\boldsymbol{v} = (2, \mathfrak{b}^\infty, \mathfrak{b}^\infty, \mathfrak{b}^\infty)$, by bidding $\mathfrak{b}^\infty$ or -1.

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

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

Theorem 8. *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	$\xi_1(p), \ldots, \xi_p(p)$	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 1, 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* [31]. Somewhat counterintuitively, if costs are supermodular but not necessarily symmetric, then sequential stand-alone mechanisms are in general not GSP [31, p. 297, second remark after Proposition 1].

7 Conclusion

In this work, 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 scheduling problem as an application and also to a setting with non-symmetric costs. Clearly, our work has to leave open many issues:

- For symmetric and/or subadditive costs, we still need an exact characterization with respect to the *best* possible BB that GSP mechanisms can achieve.
- Can our techniques be generalized to all/most non-symmetric cost functions? What is the potential of Algorithm 1? How would more general precedence mechanisms have to be like?
- Finally: Is it possible to design GSP mechanisms that improve on the BB *and* EFF of Moulin mechanisms?

References

[1] Aaron Archer and Éva Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS'01)*, pages 482–491, 2001. DOI: 10.1109/SFCS.2001.959924. → 6

[2] Aaron Archer, Joan Feigenbaum, Arvind Krishnamurthy, Rahul Sami, and Scott 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. → 5

[3] Yvonne Bleischwitz and Burkhard Monien. Fair cost-sharing methods for scheduling jobs on parallel machines. In Tiziana Calamoneri, Irene Finocchi, and Giuseppe F. Italiano, editors, *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. → 5

[4] Yvonne Bleischwitz and Florian 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. → 5

[5] Yvonne Bleischwitz, Burkhard Monien, and Florian Schoppmann. To be or not to be (served). In Xiaotie Deng and Fan Chung Graham, editors, *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. → 4, 5, 6

[6] Yvonne Bleischwitz, Burkhard Monien, Florian Schoppmann, and Karsten Tiemann. The power of two prices: Beyond cross-monotonicity. In Luděk Kučera and Antonín Kučera, editors, *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. → 1, 5, 10

[7] Janina Brenner and Guido Schäfer. Cost sharing methods for makespan and completion time scheduling. In Wolfgang Thomas and Pascal Weil, editors, *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. → 5

[8] Janina Brenner and Guido Schäfer. Singleton acyclic mechanisms and their applications to scheduling problems. In Monien and Schroeder [30], pages 315–326. DOI: 10.1007/978-3-540-79309-0_28. → 5

[9] Shuchi Chawla, Tim Roughgarden, and Mukund Sundararajan. Optimal cost-sharing mechanisms for Steiner forest problems. In Paul Spirakis, Marios Mavronicolas, and Spyros Kontogiannis, editors, *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. → 5

[10] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971. DOI: 10.1007/BF01726210. → 2

[11] Shahar Dobzinski, Aranyak Mehta, Tim Roughgarden, and Mukund Sundararajan. Is Shapley cost sharing optimal? In Monien and Schroeder [30], pages 327–336. DOI: 10.1007/978-3-540-79309-0_29. → 4

[12] Bhaskar 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. → 5

[13] Bhaskar Dutta and Debraj Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57(3):615–635, 1989. URL http://www.jstor.org/stable/1911055. → 20

[14] Joan Feigenbaum, Christos Papadimitriou, and Scott 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. → 5

[15] Joan Feigenbaum, Arvind Krishnamurthy, Rahul Sami, and Scott 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. → 3, 5

[16] Ronald Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969. URL http://www.jstor.org/stable/2099572. → 21

[17] Jerry Green and Jean-Jacques 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. → 2

[18] Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973. URL http://www.jstor.org/stable/1914085. → 2

[19] Anupam Gupta, Jochen Könemann, Stefano Leonardi, R. Ravi, and Guido 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. → 5

[20] Anupam Gupta, Aravind Srinivasan, and Éva Tardos. Cost-sharing mechanisms for network design. *Algorithmica*, 50(1):98–119, January 2008. DOI: 10.1007/s00453-007-9065-y. → 5

[21] Dorit S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, 1997. → 6

[22] Nicole Immorlica, Mohammad Mahdian, and Vahab S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. *ACM Transactions on Algorithms*, 4(2):1–25, May 2008. DOI: 10.1145/1361192.1361201. → 3, 5, 6, 7, 24

[23] Kamal Jain and Vijay 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. → 5

[24] Kamal Jain and Vijay 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. → 5

[25] Kathryn Kent and Darko Skorin-Kapov. Population monotonic cost allocation on MSTs. In T. Hunjak, Lj. Martic, and L. Neralic, editors, *Proceedings of the 6th International Conference on Operations Research (KOI'96)*, pages 43–48, Rovinj, Croatia, 1996. Croatian Operational Research Society. → 5

[26] Jochen Könemann, Stefano Leonardi, Guido Schäfer, and Stefan 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. → 5

[27] Ron Lavi. Computationally efficient approximation mechanisms. In Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory*, chapter 12. Cambridge University Press, 2007. → 6

[28] Stefano Leonardi and Guido 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. → 5

[29] Aranyak Mehta, Tim Roughgarden, and Mukund 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. → 4, 5

[30] Burkhard Monien and Ulf-Peter Schroeder, editors. *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)*, volume 4997 of *LNCS*, 2008. DOI: 10.1007/978-3-540-79309-0. → 28, 29

[31] Hervé Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999. DOI: 10.1007/s003550050145. → 1, 3, 5, 8, 21, 24, 27

[32] Hervé Moulin and Scott Shenker. Strategyproof sharing of submodular costs: budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001. DOI: 10.1007/PL00004200. → 2, 3

[33] Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behaviour*, 35:166–196, 2001. DOI: 10.1006/game.1999.0790. → 6

[34] Martin Pál and Éva 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. → 5

[35] Paolo Penna and Carmine 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. → 3

[36] Tim Roughgarden and Mukund 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. → 3, 5

[37] Tim Roughgarden and Mukund Sundararajan. Approximately efficient cost-sharing mechanisms. *arXiv report*, June 2006. URL http://arxiv.org/abs/cs/0606127. → 3, 5

[38] Florian Schoppmann. The power of small coalitions in cost sharing. In Christos Papadimitriou and Shuzhong Zhang, editors, *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. → 6

[39] Lloyd S. Shapley. A value for n-person games. In Harold W. Kuhn and Albert 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. → 20

[40] Yves 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. → 5

[41] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961. URL http://www.jstor.org/stable/2977633. → 2