TRADE-OFFS IN COST SHARING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Mukund Sundararajan

June 2009

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Tim Roughgarden) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(John Mitchell)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Ramesh Johari)

Approved for the Stanford University Committee on Graduate Studies.

Preface

To Economists: This thesis investigates a classic impossibility result from economics—the non-existence of cost-sharing mechanisms that are efficient, budget-balanced and incentive compatible—using an approach standard in computer science. For a specific cost-sharing problem, we ask: How much efficiency does the best incentive compatible, budget-balanced mechanism achieve in comparison to the efficient allocation?

Unlike an impossibility result, this approach is entirely constructive, and results in a concrete prescription for what mechanism we should deploy. This thesis identifies optimal mechanisms for a wide variety of cost-sharing problems, discusses how we can trade budget-balance for efficiency, and what makes some cost-sharing problems harder than others.

To Computer Scientists: This thesis investigates resource allocation where some of the problem parameters are privately held by agents that vie for the resources. We must find algorithms that elicit the privately held parameters, allocate resources optimally, and recover the cost of the allocated resources from the agents. Just as we must deal with lack of knowledge of future demand when designing online algorithms, we must now deal with economically and game-theoretically motivated constraints. Welcome to a parallel universe replete with hardness results, optimal algorithms and complexity classes.

Acknowledgements

This is the first time I have taken on a project that was meant to take longer than a few months. I could not have done it without support of the following people.

Fun friends at work—Adam, Aleksandra, Aneesh, Arash, Arnab, Damon, Hamid, Mike, Peerapong, Qiqi, Shaddin, Shubha, Ying. Efficient admins—Lynda Harris, Wendy Cardamone. Knowledgeable and approachable mentors on various research projects—Michael Brudno, Anupam Datta, Jason Hartline, Arpita Ghosh, and David Pennock. Bright collaborators on some of the work in this thesis—Shuchi Chawla, Aranyak Mehta, Shahar Dobzinksi. An intimidating, but ultimately friendly, orals committee—Serafim Batzoglou, Ramesh Johari, John Mitchell, Rajeev Motwani, and Tim Roughgarden. A subset of which is my reading committee—Ramesh Johari, John Mitchell, and Tim Roughgarden.

And then there are people who I cannot thank enough. Serafim Batzoglou, who took me on as a research assistant though I had no prior research experience. John Mitchell, who co-advised me throughout my stay at Stanford. I collaborated with John and his group on fun projects unrelated to this thesis. I shall take with me his truly wonderful Weltanschauung.

My thesis advisor, Tim Roughgarden, has been a wonderful mentor and guide. He is a co-author on all the work in this thesis. But most of all he is an inspiration as a consummate theoretical computer science researcher. My wife Anagha, who simply could not have been more supportive. Marrying her has easily been the best decision of my life. My parents S.S. Sundararajan and Geeta Sunder who raised me to think and act independently; what better preparation for a PhD?

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 A Motivating Example

How should the cost of a joint project be shared by its participants? Here is a motivating example paraphrased from Young et al. [67].

Example 1.1.1 In the early 1940s some municipalities in southern Sweden formed an association to tackle a potential water scarcity problem. In the late 1960s this group started to design a major project for obtaining water from a lake outside the region via a 80-km long tunnel. Besides the tunnel, water treatment systems at the source and water distribution systems also needed to be built.

The viability of this project depended on how many municipalities could be induced to participate and bear the cost of the project. This in turn depended on how much each of these municipalities would be obliged to pay for participation, bearing in mind the availability and costs of developing their own on-site resources.

The above example is characteristic of the circle of issues we investigate in this thesis: We would like to determine an economically efficient level of resource allocation in a public project that involves several potential participants. The cost of the project depends on who participates in it. In the above example, the necessary capacity, and hence the cost of the tunnel, depended on the set of municipalities that actually took part. Part of the problem data—namely, the value of the public project to

the various participants—is privately held. The participants are self-interested, and might lie about this information if it benefits them. For the project to be feasible, the cost incurred by it should be recovered from the eventual participants.

1.2 Mechanism Design

Such optimization settings, where part of the input is privately held by self-interested agents, are the topic of study of a field of economics called *mechanism design* (see for instance Chapter 23 of [58]). Mechanism design has numerous applications to, for example, auction design, pricing problems, and network protocol design [34, 42, 58, 66].

An illustrative, paradigmatic problem in mechanism design is allocating a single unit of an indivisible good to one of n potential buyers. Each bidder i has a *valuation* v_i, expressing its maximum willingness to pay for the good. We assume that this value is known only to the bidder, and not to the auctioneer. A *mechanism* for selling a single good is a protocol that determines the winner and the selling price. Each bidder i is "selfish" in that it wants to maximize its "net gain" $(v_i - p)x_i$ from the auction, where p is the price, and x_i is 1 if the bidder wins and 0 if the bidder loses.

What optimization problem underlies a single-good auction? One natural goal is *economic efficiency*, which in this context demands that the good is allocated to the bidder with the highest valuation. This goal is trivial to accomplish if the valuations are known a priori. Can it be achieved when the valuations are private?

Vickrey [74] provided an elegant solution. First, each player submits a sealed bid b_i to the seller, which is a proxy for its true valuation v_i. Second, the seller awards the good to the highest bidder. This achieves the efficient allocation *if* we can be sure that players bid their true valuations—if $b_i = v_i$ for every i. To encourage players to bid truthfully, we must charge the winner a non-zero price. (Otherwise, all players will bid gargantuan amounts in an effort to be the highest.) On the other hand, if we charge the winning player its bid, it encourages players to underbid. (Bidding your maximum willingness to pay ensures a net gain of zero, win or lose.) Vickrey [74] suggested charging the winner the value of the *second-highest* bid, and

proved that this price transforms truthful bidding into an optimal strategy for each bidder, independent of the bids of the other players. In turn, the Vickrey auction is guaranteed to produce an efficient allocation of the good, provided all players bid in the obvious, optimal way.

1.3 Cost-Sharing Problems

Unlike the single-item auctions, the problems we are interested in have no explicit limit on the number of winners (players who eventually participate). On the other hand, servicing many or all the players may require vast resources. So, we must take into account the *cost* of providing service. Formally, a *cost-sharing problem* is defined by a set U of players vying to receive some good or service, and a cost function $C : 2^U \to \mathcal{R}^+$ describing the cost incurred by the mechanism as a function of the outcome — the set S of winners. We assume that $C(\emptyset) = 0$ and that C is nondecreasing (i.e., $S \subseteq T$ implies $C(S) \leq C(T)$). In Example 1.1.1, for a set S of municipalities, $C(S)$ represents the total cost—incurred from the construction of a water treatment plant, a tunnel of sufficient capacity, and a distribution network—of delivering water of adequate quality and quantity to the municipalities.

As in the previous section, every player $i \in U$ has a private value v_i for service. And every bidder is self-interested, i.e. it maximizes its "net gain" $(v_i - p_i)x_i$ from the mechanism, where p_i is the price it is charged, and x_i is 1 if the bidder wins and 0 if the bidder loses. In Example 1.1.1, the value v_i represents municipality i's value for the water project.

For a given set U and function C, a *cost-sharing mechanism* is a protocol that takes the cost function C and bids as input and decides which players win (the x_i's) and at what prices (the p_i's).

1.3.1 Desiderata

As this is a public project, the primary goal should be to come up with a solution that is economically efficient for the society at large, i.e. for every valuation profile,

we would like to pick an allocation that embodies the optimal trade-off between total value serviced and total cost incurred. Formally, for a cost function C and a valuation profile $\{v_i\}_{i \in U}$, the *efficient allocation* is the subset that maximizes the *social welfare*:

$$W(S) = v(S) - C(S)$$

Here $v(S)$ denotes $\sum_{i \in S} v_i$. Of course, as in the single-item auction example, values are private information, and we would like mechanisms that are incentive compatible, i.e. the cost-sharing mechanism should encourage players to bid truthfully. However, there is an additional constraint: For the project to be feasible, the mechanism should recover the cost of the project from the winners, i.e., it should be *no-deficit*.

Summarizing, we have identified three natural goals in cost-sharing mechanism design: *incentive-compatibility*, meaning that every player's optimal strategy is to bid its true private value v_i for receiving the service; *no-deficit*, meaning that the mechanism recovers its incurred cost with the prices charged; and *efficiency*, meaning that the cost and valuations are traded off in an optimal way. (For the most part we will identify computationally efficiency mechanisms, though our hardness results will not reference this constraint.)

1.3.2 Motivating Related Work

Unfortunately, even for very simple cost-sharing problems, there are no mechanisms that simultaneously satisfy these three constraints (see Example 3.1.1). Intuitively, this impossibility stems from the payments having to play two distinct, incompatible roles—to ensure incentive compatibility, and to recover cost. Chapter 3 discusses this issue further.

This impossibility result motivates relaxing at least one of these properties. Until recently, nearly all work in cost-sharing mechanism design completely ignored either the cost-recover constraint or efficiency. Without the cost-recovery constraint, there is an extremely powerful and flexible mechanism that is incentive-compatible and efficient: the *VCG mechanism* (see e.g. [33, 63]). This mechanism specializes to the Vickrey auction when selling a single good, but it is far more general. The

VCG mechanism typically does not satisfy the cost-recovery constraint within any approximation factor (assuming "individually rational" prices, see e.g. [32] for details).

A second approach is to discard economic efficiency as an objective and insist on incentive-compatibility and the no-deficit condition. Until very recently (see Chapter 4)), the only general technique for designing mechanisms of this type was due to Moulin [62]. Researchers have developed numerous approximately no-deficit Moulin mechanisms for cost-sharing problems arising from different combinatorial optimization problems, including fixed-tree multicast problems [5, 32, 33]; more general submodular problems [62, 63]; scheduling problems [11, 17]; network design problems [37, 39, 47, 48, 51, 55, 68]; facility location problems [56, 68]; and various covering problems [26, 45]. With one exception discussed in Section 4.5.1, none of these works provided any guarantees on the economic efficiency achieved by the proposed mechanisms.

1.4 What Is in this Thesis?

Here is a brief overview of the results in this thesis.

1.4.1 Measuring Efficiency Loss via Approximation

Impossibility results are common in computer science. There are, for instance, the (conditional) impossibility of polynomial time implementation motivated by Cook's theorem [23], and the information theoretic lower bounds stemming from lack of information in restricted models of computation such as the online model [16] or the streaming model [64]. When faced with such hardness results computer scientists are accustomed to devising heuristics and proving worst-case guarantees about them using the notion of approximation.

Following this approach, we quantify the efficiency loss of incentive compatible, no-deficit mechanisms via approximation ratios. That is, we discuss, for a fixed cost-sharing problem, what approximation of the optimal welfare does a specific truthful, no-deficit mechanism achieve? Chapter 3 describes this approach.

1.4.2 Optimal Moulin Mechanisms

Given this quantitative measure of efficiency loss, we can rigorously compare the economic efficiency of different mechanisms for a cost-sharing problem. Given two mechanisms A and B, mechanism A has better economic efficiency that mechanism B if it provides a better worst-case approximation to economic efficiency. Given this ability to rank mechanisms, we can then identify a mechanism as "optimally efficient" subject to cost-recovery and incentive compatibility constraints.

We apply this approach to Moulin mechanisms, which until very recently were the only general framework for designing mechanisms that are no-deficit and incentive compatible. All Moulin mechanisms use carefully designed pricing oracles to simulate ascending auctions. Our analysis reveals that the a certain combinatorial property of the pricing oracle, called its *summability*, characterizes (lower and upper bounds) the worst-case efficiency loss of the Moulin mechanism that employs it. Chapter 5 uses this characterization to identify optimal Moulin mechanisms for various cost-sharing problems. (In general, different public projects will induce different cost functions; Chapter 2 defines several types of cost-sharing problems.) All of the mechanisms we identify as optimal Moulin mechanisms are polynomial-time implementable, and optimal even among Moulin mechanisms not restricted to run in polynomial time. Our proofs include ideas inspired by primal-dual and online algorithms.

1.4.3 A Hierarchy of Cost-Sharing Problems

Cost-sharing problems vary in difficulty (in terms of achieving our desiderata). We identify the intrinsic complexity of a cost-sharing problems with the worst-case approximation achieved by an optimal Moulin mechanism for it. As an analogy recall that the "difficulty" of an NP-hard optimization problem is often identified with the best-possible approximation ratio achievable by a polynomial-time algorithm for it, assuming $P \neq NP$. Different NP-Hard problems admit different approximation ratios (see e.g. [7]).

Our analysis reveals a hierarchy of cost-sharing problems: Problems involving submodular cost and facility location always admit a logarithmic approximation (in the

number of players), and in the worst case, nothing better. Network design problems that are variants of Steiner tree cost functions admit a polylogarithmic approximation, and in the worst case, no better. Finally, problems that involve variants of set-cover admit no better than a polynomial approximation in the worst case. See Section 4.3.

1.4.4 New Mechanisms: Acylic Mechanisms

Having identified optimal Moulin mechanisms for various cost-sharing problems, we ask if there is a better alternative to the Moulin framework. We identify a new framework for designing truthful and no-deficit cost-sharing mechanisms, called *acyclic mechanisms*.

Like Moulin mechanisms, acyclic mechanisms are ascending auctions. However, careful resolution of certain non-determinism present in Moulin mechanisms permits acyclic mechanisms to employ a wider class of pricing oracles. Thus, acyclic mechanisms strictly generalize Moulin mechanisms and offer two important advantages. First, it is easier to design acyclic mechanisms than Moulin mechanisms: many classical combinatorial algorithms (based on the primal-dual method) naturally induce non-Moulin, polynomial-time implementable acyclic mechanisms with good performance guarantees. Second, for important classes of cost-sharing problems, acyclic mechanisms have exponentially better economic efficiency than Moulin mechanisms. The only, minor drawback of acyclic mechanisms is that they sacrifice a modicum of incentive compatibility: Moulin mechanisms are slightly more robust to coalitional manipulations compared to acyclic mechanisms. See Chapters 6 and 7 for details.

1.4.5 Lower Bounds on Truthful Mechanisms

Are there mechanisms, not necessarily computationally efficient, that are superior to acyclic mechanisms? We identify a no-deficit, incentive compatible non-acyclic mechanism that achieves a logarithmic approximation (in the number of players) of the optimal efficiency for all cost-sharing problems with underlying monotone cost functions. In contrast, for acyclic mechanisms, we only know of an analogous result

for the narrower class of subadditive cost-sharing problems.

Are there no-deficit mechanisms, not necessarily computationally efficient, that achieve a constant factor approximation of the optimal efficiency? We show that, in the worst case, the answer is negative. We identify a logarithmic lower bound on the worst-case efficiency approximation of every truthful, no-deficit mechanism, applicable to a simple and central cost-sharing problem. The lower bound is robust and even applies to all truthful, randomized cost-sharing mechanisms, and randomized mechanisms that are only truthful in expectation. See Chapter 8.

We conclude with a list of open questions in Chapter 9.

Remark 1.4.1 This thesis is based on the following papers. Roughgarden and Sundararajan [72] formulates the approximation based measure of efficiency loss. Roughgarden and Sundararajan [72], Roughgarden and Sundararajan [71], and Chawla et al. [19] use this to identify optimal Moulin mechanisms for various cost-sharing problems. Mehta et al. [59] introduces acyclic mechanisms, and bounds their performance. Dobzinski et al. [27] establishes fundamental lower bounds on the efficiency loss of all incentive compatible, approximately budget-balanced mechanisms.

1.5 Notes

1.5.1 Binary Service

A key restriction of our model is that the mechanism offers each player only one of two levels of participation, service or no service. One could take instead model and determine the *extent* of players' participation in the project. (Clarke [22] and Groves [36]) allow for this possibility.)

However, this restriction offers two technical advantages. First, from a mechanism design point of view, players are single-parameter agents, i.e. each player's private information is a single number. Incentive compatible mechanisms for single-parameter settings are better understood than those for multi-parameter settings, a fact we leverage throughout this thesis, especially in Chapter 8 [1].

[1] Admittedly there do exist more general single-parameter models that allow for multiple levels of

Second, for cost functions that are defined implicitly as the optimal solution of an instance of a combinatorial optimization problem (see Section 2.2), we also hold the mechanism M responsible for constructing a feasible solution to the optimization problem induced by the served set S. When there is a binary notion of service, several previously developed algorithmic techniques are useful for this task.

Even with this restriction, we can model several natural resource allocation settings, and, further, our ideas have found application in settings with multiple levels of service. See [59, 14].

1.5.2 Excludability

Another restriction on our model is that we assume that the mechanism can exclude players from participation. The ability to exclude players is vital to recovering the cost of the constructed solution, without which we encounter the *free rider* problem, i.e., participants can enjoy the benefits of participation without contributing to the cost of the resources (see for instance [2]). Implementing exclusion may however require special effort:

Example 1.5.1 An interesting historical example of exclusion is Foothills Park, a park adjoining three cities, Palo Alto, Mountain View, and Los Altos. During the planning phase, all three cities considered participating in a project to turn this land into a park. Eventually, Mountain View and Los Altos felt their money could be better spent and refused to take part. As a result Palo Alto bought and developed the land, turning it into what is now Foothills Park. However, they also ensured that only residents of Palo Alto have access to the park. Proof of residency in Palo Alto is required at the gate [1]!

Of course, exclusion need not imply lack of service. For instance, in Example 1.1.1, exclusion may mean that municipalities develop and use on-site water resources; in the above example, residents of Mountain View and Los Altos have access to other parks.

service. However, these prevent us from leveraging the rich literature on combinatorial algorithms .

Admittedly, for some public projects, exclusion is not feasible, and our model is not applicable to these. For instance, consider a public project involving municipalities to ensure clean air. It is not (yet?) possible to exclude non-participating municipalities from enjoying the benefits of cleaner air.

1.5.3 Cooperative Game theory

Part of our model—the cost function on subsets of players—defines a *cost game* from cooperative game theory (see for instance Osborne and Rubinstein [66], Chapter 13).

A solution in cooperative game theory, given a set of potentially cooperating players, describes a putative assignment of prices to these players that cover the cost of service, i.e. for all sets $S \subseteq U$ of cooperating players, a set of numbers $\chi(i, S)$ for all $i \in S$ such that $\sum_{i \in S} \chi(i, S) = C(S)$. This corresponds to our cost-recovery requirement.

Unlike mechanism design, cooperative game theory does not deal with private information, in fact it does not include the notion of value for service, and presumes that all players desire service, no matter what they are charged. Consequently, there is no notion of computing an optimal trade-off between value and cost, i.e. no notion of economic efficiency. And, there is no notion of incentive compatibility as all information is considered public.

The key issues in cooperative game theory are *core stability*–which means no coalition has an incentive to secede from the project—and fairness. Both issues are valid concerns in our motivating examples, but we choose the mechanism design approach and focus on the private information and economic efficiency aspects. That said, some of the mechanisms we study inadvertently satisfy (approximate) versions of core stability and fairness.

Our work is related to cooperative game-theory in the broader sense of Moulin [61]. Moulin [61] proposes three modes of cooperation. The *direct agreement mode*—where players arrive at consensus by face-to-face bargaining, the *decentralized mode*—where the outcome results from players acting in a decentralized, self-interested way, and the *justice mode* where decision power is vested in a central arbitrator, whose choices

follow some normative principles, presumably arrived at by consensus among the players. The model that we study corresponds to the third mode, with the normative principles being efficiency, no-deficit and incentive compatibility.

1.5.4 Profit Maximization

An alternative route to the one we take in this thesis (maximizing social welfare and ensuring cost-recovery) is to maximize the mechanism's profit, like Myerson [65] does in the context of single-item auctions. A straightforward generalization of Myerson [65] identifies revenue-maximizing, incentive-compatible cost-sharing mechanisms under Bayesian assumptions on player valuations. Beyond this observation, we are unaware of any cost-sharing work that takes this route.

Chapter 2

Preliminaries

This chapter includes definitions and preliminary results used throughout this thesis. Treat this chapter as a reference. Section 2.1 describes our mechanism design setting. Section 2.2 defines several types of cost-sharing problems. Section 2.3 defines our desiderata—*budget-balance* and *efficiency*. Section 2.4 discusses and defines the various notions of incentive compatibility used in this thesis. Section 2.5 discusses *cost-sharing methods*, a combinatorial abstraction used extensively in the literature on cost-sharing mechanisms.

2.1 Mechanism Design Setting

This section describes the inputs to the mechanism, the outputs of the mechanism, the basic protocol, and the knowledge assumptions.

The problem input is a set U of n players and a cost function C that assigns a cost $C(S)$ to every set $S \subseteq U$ of players. We assume that $C(\emptyset) = 0$ and that $C(S) \leq C(T)$ for all $S \subseteq T \subseteq U$. In addition, every player $i \in U$ possesses a private, nonnegative *valuation* v_i, representing player i's maximum willingness to pay for being included in the chosen set S.

A *mechanism* collects a nonnegative bid b_i from each player $i \in U$, selects a set $S \subseteq U$ of players, and charges every player i a price p_i. For cost functions that are

defined implicitly as the optimal solution of an instance of a combinatorial optimization problem (see Section 2.2 for several examples), we also hold the mechanism M responsible for constructing a feasible solution to the optimization problem induced by the served set S. The cost $C_M(S)$ of this feasible solution is in general larger than the cost $C(S)$ of an optimal solution; indeed, many of the underlying combinatorial optimization problems we study are NP-Hard and the mechanisms we propose will run in polynomial time.

Remark 2.1.1 Mechanisms can be defined more generally, but the Revelation Principle [58, P.871] justifies restricting attention to the class of "direct-revelation mechanisms" defined above.

2.2 A Hierarchy of Cost-Sharing Problems

One of the main points of this thesis is to contrast the intrinsic difficulty of various cost-sharing problems, as a function of the underlying cost-function (recall Section 1.4.3). To this end we define, and will later study, several types of cost-sharing problems. Many of these are motivated by standard combinatorial optimization problems. We now define these cost-sharing problems via their underlying cost functions. These cost-sharing problems are in a hierarchy as Figure 2.1 shows; for instance every fixed-tree multicast problem (Example 2.2.8) is also a submodular cost-sharing problem (Example 2.2.2).

We start by defining a class of cost-sharing problems that encompasses all those we study. In several resource allocation settings, a feasible way to service a set S_1 of players can be combined with a feasible way to service a set S_2 of players to serve the union $S_1 \cup S_2$ of players, at no additional cost. The cost function defined by the optimal solutions (ranging over subsets of U) of such an optimization problem defines a subadditive cost function in the following sense.

Example 2.2.1 (Subadditive Cost Function) A *subadditive cost-sharing problem* is defined by a player set U and a nondecreasing cost function C such that, for every

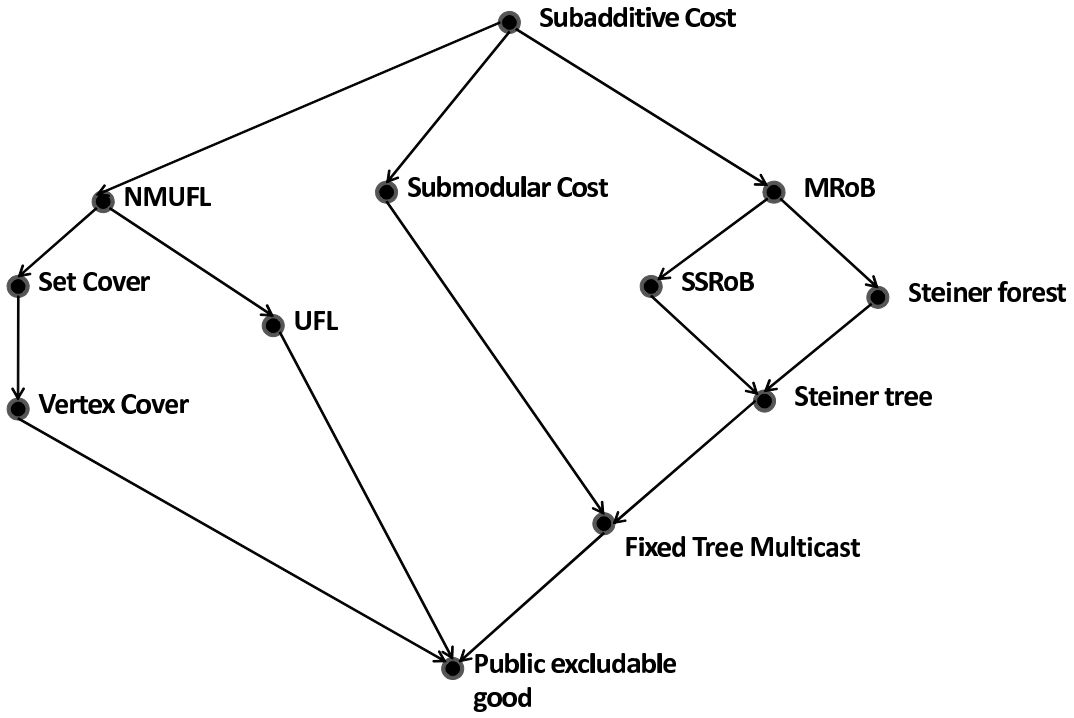


Figure 2.1: A hierarchy of cost-sharing problems. Nodes are families of cost-sharing problems. Edges between nodes assert that every problem instance of the destination node is also a problem instance of the source node.

$S_1 \subseteq U$, $S_2 \subseteq U$,

$$C(S_1) + C(S_2) \geq C(S_1 \cup S_2). \tag{2.1}$$

Next, we define a family of cost-sharing problems whose underlying cost function exhibits diminishing returns, i.e., the incremental cost of servicing a player falls as the set of players already serviced grows. If players are symmetric, this corresponds to a cost-sharing problem with a concave cost function. In general, we have:

Example 2.2.2 (Submodular Cost Function) A *submodular cost-sharing problem* is defined by a player set U and a nondecreasing cost function C such that, for

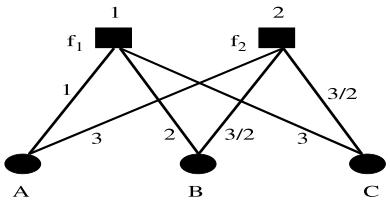


Figure 2.2: An instance of uncapacitated facility location (NMUFL)(Example 2.2.3)

every $S_1 \subseteq S_2 \subseteq U$ and $i \in U$, $i \notin S_2$,

$$C(S_2 \cup \{i\}) - C(S_2) \leq C(S_1 \cup \{i\}) - C(S_1). \tag{2.2}$$

We now discuss types of cost-sharing problems defined implicitly by combinatorial optimization problems. Though subadditive, these cost-sharing problems are not submodular in general (as depicted in Figure 2.1). We begin with a family of cost-sharing problems based on the well-known facility location problem.

Example 2.2.3 (Non-Metric Uncapacitated Facility Location (NMUFL)) An *uncapacitated facility location (NMUFL) cost-sharing problem* is defined by a player set U and a nondecreasing cost function C, defined implicitly by a set F of facilities, an opening cost f_q for each facility $q \in F$, and a nonnegative cost function c defined on $F \times U$. In Figure 2.2, for example, the universe contains three players, there are two facilities with opening costs $f_1 = 1$ and $f_2 = 2$, and the connection costs between facilities and players are as shown. For a subset $S \subseteq U$ of the players, the cost $C(S)$ is defined as the cost of the cheapest way to open a non-empty subset of facilities and connect all of the players in S to open facilities. Formally,

$$C(S) = \min_{\emptyset \neq F^* \subseteq F} \left(\sum_{q \in F^*} f_q + \sum_{i \in S} \min_{q \in F^*} c(q, i) \right).$$

For instance, in Figure 2.2, the cost $C(\{A, B, C\})$ of servicing all of the players is 7.

We pay special attention to NMUFL cost-sharing problems that have metric connection costs:

Example 2.2.4 (Metric Uncapacitated Facility Location) A *metric uncapacitated facility location cost-sharing problem* is a NMUFL cost-sharing problem in which the connection costs satisfy the triangle inequality: for every pair $i, i' \in U$ of demands and pair $q, q' \in F$ of facilities,

$$c(q, i) \leq c(q, i') + c(q', i') + c(q', i).$$

Another sub-class of NMUFL cost-sharing problems have connection costs that are either zero or ∞. The underlying cost functions are equivalent to the well-known set-cover problem, where the elements correspond to demands, sets and their costs correspond to facilities and their opening costs, and connection costs are either 0 (if the given element belongs to the given set) or $+\infty$ (otherwise).

Example 2.2.5 (Set Cover) A *set cover cost-sharing problem* is defined by a player set U and a nondecreasing cost function C defined as follows. There is a collection $\mathcal{C} = \{A_1, \ldots, A_m\}$ of subsets of U with nonnegative costs $c_1, \ldots, c_m$. For a subset $S \subseteq U$, $C(S)$ is defined as the cost of the cheapest way of covering the elements of S using subsets from $\mathcal{C}$.

Some instances of set cover cost-sharing problems are also instances of the vertex cover optimization problem: Edges correspond to elements, and sets of edges incident on a common vertex form the subsets.

Example 2.2.6 (Vertex Cover) A *vertex cover* cost-sharing problem is defined by a player set U and a nondecreasing cost function C defined implicitly by an undirected graph $G = (V, U)$ with nonnegative vertex weights. For a subset $S \subseteq U$, $C(S)$ is defined as the minimum weight vertex cover of the graph V, S—a subset of vertices that includes at least one endpoint of each edge in S.

An important class of non-NMUFL cost-sharing problems have the well-known Steiner tree problem, or a variant of it, as the underlying cost function:

Example 2.2.7 (Steiner Tree) A *Steiner tree cost-sharing problem* [47] is defined by a player set U and a nondecreasing cost function C implicitly defined by the

following. There is an undirected graph $G = (V, E)$ with non-negative weights on the edges. There is a designated vertex $t \in V$ called the root. Every player $i \in U$ is associated with a node in V. The cost $C(S)$ of a subset S of players is defined as that of a minimum-cost subgraph of G that spans all of the players of S as well as the root t.

While Steiner tree cost-sharing problems are not in general submodular, a subclass of these problems are:

Example 2.2.8 (Fixed-Tree Multicast) A *fixed-tree multicast cost-sharing problem* [33, 63], is a Steiner tree cost-sharing problem where the input graph is a tree.

We next list three generalizations of Steiner tree cost-sharing problems.

Example 2.2.9 (Steiner Forest) In a *Steiner forest cost-sharing problem* with player set U, the cost function is defined as follows. There is an undirected graph $G = (V, E)$, a non-negative cost function $c : E \to R^+$, and a set R of terminal pairs $\{(s_1, t_1), (s_2, t_2) \ldots (s_k, t_k)\} \subseteq V \times V$. The cost of servicing a subset $S \subseteq U$ of terminal pairs is the cost of the minimum weight subgraph of G that connects the terminal pairs together.

A different generalization of Steiner Tree problems models a setting where the cost of an edge is a function of the number of nodes that use it to connect to the root.

Example 2.2.10 (Single-Sink Rent-or-Buy (SSRoB)) In a *single-sink rent-or-buy (SSRoB) cost-sharing problem* with player set U, the cost function is defined as follows. There is a graph $G = (V, E)$ with edge costs that satisfy the Triangle Inequality, a root vertex t and a parameter $M \geq 1$. Each player $i \in U$ is located at a vertex of G. For any set $S \subseteq U$ of players, a feasible solution to the SSRoB problem induced by S is a way of installing sufficient capacity on the edges of G so that every player in S can simultaneously route one unit of flow to t. Installing x units of capacity on an edge e costs $c_e \cdot \min\{x, M\}$; the parameter M can be interpreted as the ratio between the cost of "buying" infinite capacity for a flat fee and the cost of "renting" a single unit of capacity. The cost $C(S)$ of a subset $S \subseteq U$ of players is

then defined as the cost of an optimal solution to the SSRoB problem induced by the set S.

We now define a set of cost-sharing problems that simultaneously generalize Steiner forest and SSRoB cost-sharing problems.

Example 2.2.11 (Multicommodity Rent-or-Buy(MRoB)) In a MRoB cost-sharing problem, each player i corresponds to a pair of nodes s_i, t_i. All other aspects of the problem are similar to SSRoB cost-sharing problems.

The following cost-sharing problem is fundamental, because it is an instance of every one of the cost-sharing problems defined above. Thus every negative result that applies to this problem automatically applies to all of the other cost-sharing problem families.

Example 2.2.12 (Excludable Public Good) In the *excludable public good cost-sharing problem* with player set U, the cost function is defined as follows. The cost of any non-empty subset $S \subseteq U$ of players is 1 and the cost of the empty set is 0.

To conclude, we mention a simple cost-sharing problem based on pure marginal cost.

Example 2.2.13 (Marginal Cost) In an *additive cost-sharing problem* with player set U, the cost function is defined as follows. Every player $i \in U$ is associated with a cost c_i. The cost of any non-empty subset $S \subseteq U$ of players is $\sum_{i \in S} c_i$ and the cost of the empty set is 0.

2.3 Budget-Balance and Efficiency

We now formally define two of our mechanism design objectives, budget-balance and efficiency.

Definition 2.3.1 A mechanism M for the cost-sharing problem C is (β, γ)-*budget-balanced* if

$$\frac{C_M(S)}{\gamma} \leq \sum_{i \in S} p_i \leq \beta \cdot C(S)$$

for every outcome — set S, prices p, and, if applicable, feasible solution with service cost $C_M(S)$ — of the mechanism. Recall from Section 2.1 that the cost of the solution produced by the mechanism may not be optimal, and hence, in general $C_M(S) \neq C(S)$.

A mechanism is *competitive* in the sense of Pál and Tardos [68] if it is $(1, \gamma)$ budget-balanced for some $\gamma \geq 1$. A *β-budget-balanced mechanism* is, by definition, $(\beta, 1)$-budget-balanced. A *no-deficit* mechanism is β-budget-balanced for some $\beta \geq 1$. A *budget-balanced* mechanism is 1-budget-balanced.

Remark 2.3.2 Our definition of budget-balance includes an upper and a lower bound on the total payment. The lower bound assert that we would like the mechanism to (approximately) recover the cost that it incurs, i.e., be budget-feasible. The upper bound asserts that the mechanism should not end up with a budget surplus, a secondary requirement that obviates the need to determine how to expend the surplus. For the most part we shall focus on no-deficit mechanisms.

Remark 2.3.3 Most previous works on approximately budget-balanced cost-sharing mechanisms define β-budget-balance to mean $(1, \beta)$-budget-balance rather than $(\beta, 1)$-budget-balance. For the cost-sharing mechanisms that we study, a mechanism meeting one definition can be modified to satisfy the other by scaling its prices accordingly, and thus the two definitions are in some sense equivalent. In this thesis, we adopt the definition that is more convenient for stating and proving efficiency guarantees. All our results have obvious analogs for the alternative definition.

Remark 2.3.4 There are two reasons to study approximate budget-balance. The first, is the impossibility result discussed in Section 3.1, which states that there are no efficient, no-deficit, truthful mechanisms. So, it is interesting to relax the cost-recovery constraint to see how efficiency improves. The second is the impossibility

results from [45], which identify several non-trivial, interesting cost-sharing problems that do not permit any budget-balanced Moulin mechanisms. (Prior to our work, this was the only framework for designing approximately budget-balanced mechanisms.)

Given a mechanism M, the *social welfare* from servicing a set T, $W(T)$, is defined as the difference between value generated by servicing the set $v(T) = \sum_{i \in T} v_i$ and the cost incurred C_M. Given valuations, the optimal social welfare W^* is

$$\max_{S \subseteq U} (v(S) - C(S))$$

This quantity is independent of any concrete mechanism. We shall often use S^* to denote an element of the set $\text{argmax}_{S \subseteq U} (v(S) - C(S))$. See Chapter 3, and specifically Section 3.2.3, for the definition of approximate efficiency.

Asymptotic Notation: Some of our approximation bounds use standard asymptotic notation. Here $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, and $f(n) = o(g(n))$ means that $\lim_{n \to \infty} f(n)/g(n)$ is bounded above by a positive constant, bounded below by a positive constant, and equal to zero, respectively.

2.4 Private Information and Incentive Compatibility

We implicitly impose two natural constraints on all our mechanisms. First, a mechanism satisfies *no positive transfers* if it never pays players, i.e., prices are always nonnegative. A mechanism is *individually rational* (or synonymously, satisfies *voluntary participation*), if truthful players derive non-negative utility from participation. Without the individual rationality constraints, players may sometimes prefer to not participate once the outcome is known, which is undesirable. For truthful, direct revelation mechanisms these requirements together imply: if the serviced set is S, then $p_i = 0$ for players $i \notin S$ and $p_i \leq b_i$ for players $i \in S$.

We now define our incentive compatibility constraints. As is standard, we assume that every player aims to maximize the quasilinear utility function $u_i(S, p_i) = v_i \cdot x_i -$

p_i, where $x_i = 1$ if $i \in S$ and $x_i = 0$ if $i \notin S$. Our incentive-compatibility constraint is the well-known strategyproof condition, stating that truthful bidding is a dominant strategy for every player.

Definition 2.4.1 (Strategyproofness) A mechanism is *strategyproof* (SP) or *truthful* if for every player i, every bid vector b with $b_i = v_i$, and every bid vector b' with $b'_j = b_j$ for all $j \neq i$, $u_i(S, p_i) \geq u_i(S', p'_i)$, where (S, p) and (S', p') denote the outputs of the mechanism for the bid vectors b and b', respectively.

Some of our mechanisms will also satisfy stronger incentive compatibility constraints.

Definition 2.4.2 (Groupstrategyproofness) A mechanism is *groupstrategyproof* (GSP) if for every subset of players $S \subseteq U$, bid vectors b, b', such that $b_i = v_i$ for $i \in S$ and $b'_i = b_i$ for all $i \notin S$, we have that if there exists a player $i \in S$ such that $u_i(S, p_i) < u_i(S', p'_i)$, then there exists a player $j \in S$ such that $u_j(S, p_j) > u_i(S', p'_j)$; here (S, p) and (S', p') denote the outputs of the mechanism for the bid vectors b and b', respectively.

Definition 2.4.3 (Weak Groupstrategyproofness) A mechanism is *weakly groupstrategyproof* (WGSP) if for every subset of players $S \subseteq U$, bid vectors b, b', such that $b_i = v_i$ for $i \in S$ and $b'_i = b_i$ for all $i \notin S$, we have that if there exists a player $i \in S$ such that $u_i(S, p_i) < u_i(S', p'_i)$, then there exists a player $j \in S$ such that $u_j(S, p_j) = u_i(S', p'_j)$; here (S, p) and (S', p') denote the outputs of the mechanism for the bid vectors b and b', respectively.

Informally, a mechanism is *groupstrategyproof* [63] if no coordinated false bid by a subset of players can ever strictly increase the utility of one of its members without strictly decreasing the utility of some other member; transfers between coalition members are not allowed. Informally, a mechanism is *weakly groupstrategyproof* [26] if no coordinated false bid by a subset of players can ever strictly increase the utility of every one of its members. Thus, in a WGSP mechanism, every deviating coalition has at least one indifferent member.

Note that GSP implies WGSP, and WGSP implies SP, however the converse does not generally hold. For example, VCG mechanisms (Section 2.4.1) are SP, but typically not WGSP. And acyclic mechanisms (Chapter 6) are WGSP, but not in general GSP.

We now mention two characterizations of truthful mechanisms. The first is a payment centric view that we use in Section 8.2. The forward direction is trivial, for the reverse consult e.g. [60].

Proposition 2.4.4 *Define property* Π *of a mechanism as follows: For every* $i \in U$ *and bid vector* b_{-i} *for players other than* i*, there is a* threshold $t_i(b_{-i})$ *such that: (i) if* i *bids more than* $t_i(b_{-i})$*, then it receives service at price* $t_i(b_{-i})$*; (ii) if* i *bids less than* $t_i(b_{-i})$*, then it does not receive service.*

Then, M *is a truthful cost-sharing mechanism that satisfies voluntary participation with the player set* U *if and only if it satisfies property* Π*.*

The second is an essentially equivalent allocation centric view that we use in Section 8.1. See Archer and Tardos [6] for a proof. An mechanism is *monotone* if for every player i and fixed bids b_{-i} of the other players, if player i receives service with a bid b_i, then it also receives service for a bid $b'_i > b_i$. A mechanism is a *threshold mechanism* if for every player i and fixed set of bids b_{-i} of the other players, it charges every winning player i the minimum bid that wins it service.

Proposition 2.4.5 *A mechanism is truthful and satisfies voluntary participation if and only if it is a monotone, threshold mechanism.*

Randomization makes a brief appearance in Chapter 8. A randomized mechanism is, by definition, a probability distribution over deterministic mechanisms. Such a mechanism is *universally truthful* if every mechanism in its support is truthful. Such a mechanism is *truthful in expectation* if no player can ever strictly increase its *expected* utility by misreporting its valuation. Every universally truthful mechanism is truthful in expectation, but the converse need not hold.

2.4.1 VCG Mechanisms

The well-known Vickrey-Clarke-Groves (VCG) mechanism is a truthful and efficient mechanism, applicable to several mechanism design settings. We define it in the context of cost-sharing (following Moulin and Shenker [63]). Given a profile v of bids, the mechanism allocates service to the set S^* that maximizes social welfare. Each player i is charged $h_i(v_{-i}) - \left(\sum_{j \in S^* \setminus \{i\}} v_j - C(S^*)\right)$. Intuitively the mechanism is truthful because the prices align the global objective (social welfare) with each player's utility function—so the player should bid truthfully to maximize utility, up to a term $h_i(v_{-i})$ that is independent of player i's bid.

An important type of VCG mechanism is the VCG mechanism with Clarke tax, where the functions h_i are defined to be the optimal social welfare without player i (alternatively with player i's value set to zero). As discussed in Moulin and Shenker [63], this mechanism is truthful, efficient, and also satisfies no positive transfers and voluntary participation; in fact it is the unique such mechanism. Unfortunately, as discussed in Chapter 3, it has a large budget deficit.

2.5 Cost-Sharing Methods

The mechanism design frameworks we study, i.e., acyclic mechanisms and Moulin mechanisms, localize the problem specific aspects into a pricing oracle called a cost-sharing method. A *cost-sharing method* χ is a function that assigns a non-negative *cost share* $\chi(i, S)$ for every subset $S \subseteq U$ of players and every player $i \in S$.

For cost functions induced by combinatorial optimization problems (such as Examples 2.2.7 and 2.2.8), a cost-sharing method outputs both cost shares and a feasible solution for the optimization problem induced by S. A cost-sharing method is (β, γ)-*budget balanced* for a cost function C and parameters $\beta, \gamma \geq 1$ if

$$\frac{C_\chi(S)}{\gamma} \leq \sum_{i \in S} \chi(i, S) \leq \beta \cdot C(S), \tag{2.3}$$

where $C_\chi(S)$ is the cost of the feasible solution produced by the method χ. As usual,

β-budget-balance is short for $(\beta, 1)$-budget-balance, and such methods are also called *no-deficit*. Moulin mechanisms (Chapter 4) require cost-sharing methods that are cross-monotonic in order to be SP.

Definition 2.5.1 A cost-sharing method is *cross-monotonic* if the cost share of a player only increases as other players are removed: for all $S \subseteq T \subseteq U$ and $i \in S$, $\chi(i, S) \geq \chi(i, T)$.

Here is an example that illustrates the definitions in this section.

Example 2.5.2 (Shapley and Sequential Cost-Sharing) Consider an instance of fixed-tree multicast (Example 2.2.8) with tree T and player set $U = \{1, 2, \ldots, n\}$. Two 1-budget-balanced cost-sharing methods are as follows. In the *sequential* cost-sharing method χ_{seq}, given a subset $S \subseteq U$, each player $i \in S$ pays the full cost of each edge of its (unique) path to the root of T that is not used by a player of S with lower index. In the *Shapley* method χ_{sh}, each player $i \in S$ pays a "fair share" of each of the edges in its path — c_e/n_e for an edge e of cost c_e, where n_e denotes the number of players of S using edge e to the reach the root of T. Since the amount a player pays for each edge in its path can only increase as other players are removed from S, both of these methods are cross-monotonic.

Chapter 3

Quantifying Inefficiency

In this chapter we discuss why efficiency loss arises in truthful, approximately budget-balanced mechanisms, and introduce our approximation based approach to measuring efficiency loss.

3.1 A Motivating Impossibility Result

Recall our desiderata from Chapter 1. Are there efficient, no-deficit, truthful mechanisms? Here is an example that shows that in general the answer is *negative*.

Example 3.1.1 Consider the excludable public good cost-sharing problem (Example 2.2.12) with $n > 1$ players. Fix any truthful, efficient mechanism. Recall that every truthful, individually-rational mechanism must offer a bid-independent take-it-or-leave it price in the sense of Proposition 2.4.4. Fix a player $i \in 1 \dots n$. Suppose that the sum of the valuations of all the players other than i ($\sum_{j \neq i} v_i$) is strictly larger than one. What take-it-or-leave-it price does the mechanism offer player i? For the mechanism to be optimally efficient, the player i should be serviced if it has any strictly positive valuation. So it must be offered a price of 0.

Now consider the valuation profile $1 + \epsilon, \dots, 1 + \epsilon$ for some positive ϵ. As the mechanism is efficient, it services all the players, and incurs a cost of 1. But, by the above argument, the mechanism does not collect *any* revenue, and thus has arbitrarily

bad deficit.

What just happened? When we insist on a truthful, efficient, budget-balanced mechanism, we are really asking if the prices can play two different roles simultaneously. Can they help incentivize an efficient allocation, while recovering cost? The above example shows that the unique prices that support the efficient allocation do not in general recover cost.

For readers familiar with the mechanism design literature, here is a slicker way to see where this impossibility comes from. Recall that every direct revelation mechanism consists of an allocation rule and a payment rule. Arguably, the central result of single-parameter mechanism design (see for instance Myerson [65] or Archer and Tardos [6]) states that the allocation rule of a truthful mechanism determines its payment function up to certain bid-independent terms. When we insist on voluntary participation, these bid-independent terms are uniformly zero. Consequently, there is a unique efficient mechanism that satisfies voluntary participation (Moulin and Shenker [63], Proposition 3 proves this uniqueness result from first principles). This happens to be the VCG mechanism with Clarke tax (Section 2.4.1). Unfortunately, this mechanism happens to have arbitrarily bad budget balance, even for simple cost-sharing problems such as the excludable public good cost-sharing problem. This mechanism charges a player only if it is pivotal—a player is pivotal if reducing its valuation to zero changes the allocation of at least one of the other players. For the excludable public good cost-sharing problem, when every player has a value larger than the cost of the good, no player is pivotal. (Our example above confirms this phenomenon.)

Finally, recall that the excludable public good cost-sharing problem is an instance of all the cost-sharing problems listed in the previous chapter other than the additive cost-sharing problem. So, the impossibility result applies to many interesting cost-sharing problem families.

3.2 How to Quantify Inefficiency?

How should we proceed?

3.2.1 Approximation

Impossibility results are common in optimization. Motivated by conditional impossibility results like Cook's Theorem [23], as well as information-theoretic lower bounds in restricted models of computation like online [16] and streaming algorithms [64], algorithm designers are accustomed to devising heuristics and proving worst-case guarantees about them using approximation measures. This approach can also be applied to cost-sharing mechanism design to quantify the inevitable efficiency loss in incentive-compatible, budget-balanced cost-sharing mechanisms. As worst-case approximation measures are rarely used in economics, this research direction has not been pursued previously.

3.2.2 The Obvious Approach Does Not Work

Several definitions of approximate efficiency are possible. Arguably, the most natural requirement is to insist that a mechanism always computes an outcome S that is a ρ-approximation of the social welfare: $W(S) \geq \rho \cdot W(S^*)$, where S^* is the economically efficient solution. Unfortunately, Feigenbaum et al. [32] shattered any hope for such a guarantee: For the excludable public good cost-sharing problem, for every $\gamma, \beta \geq 1$ and β, γ-budget-balanced incentive-compatible mechanism, there is a valuation profile such that the efficient solution has strictly positive welfare but the mechanism produces the empty outcome (with zero welfare). Thus every mechanism, no matter how intuitively "good" or "bad", is a 0-approximation algorithm for the social welfare objective. This inapproximability result is characteristic of mixed-sign objective functions such as the social welfare.

3.2.3 Our Approach, and Its Interpretations

We must therefore measure efficiency loss in a different way. Our basic efficiency guarantees have the following form, for a parameter $\rho \geq 0$ and a mechanism for the cost-sharing problem C: for every valuation profile,

$$W(S^*) - W(S) \leq \rho \cdot C(S^*), \tag{3.1}$$

where S is the output of the mechanism and S^* is an efficient outcome. In this case, we call the mechanism *ρ-approximate*.

We have chosen to present this efficiency guarantee in terms of additive welfare loss, but it is robust and admits several different interpretations. For example, the bound in (3.1) implies a relative approximation guarantee for a different formulation of economic efficiency. Precisely, define the *social cost* $\pi(S)$ of an outcome S to be the cost incurred by the mechanism plus the sum of the *excluded* valuations (i.e., opportunity cost):

$$\pi(S) = C(S) + v(U \setminus S). \tag{3.2}$$

Since social cost and social welfare are related by the affine transformation $\pi(S) = -W(S)+v(U)$, minimizing the social cost is ordinally equivalent to maximizing the social welfare. The two objective functions are not, of course, equivalent from an approximation perspective. Indeed, while the impossibility result in Feigenbaum et al. [32] precludes any relative approximation of the social welfare, *every ρ-approximate cost-sharing mechanism also $(\rho+1)$-approximates the social cost.* Such non-approximation-preserving transformations are common in applications with mixed-sign objective functions, including prize-collecting combinatorial optimization problems (e.g. [10]) and discrete maximum-likelihood problems (e.g. [54]).

A second interpretation of the bound in (3.1) is motivated by the examples used in the impossibility result in [32]. These examples are intuitively difficult because the optimal outcome S^* has large cost $C(S^*)$ and value $v(S^*)$ only slightly larger than $C(S^*)$, leaving the mechanism with no "margin for error". Can we obtain a relative approximation of welfare when the value of an optimal outcome is bounded away from its cost? To formalize this question, we say that an outcome S is *η-separated* if $W(S) \geq \eta \cdot C(S)$ or, equivalently, if $v(S) \geq (\eta + 1) \cdot C(S)$. The punchline, proved via a simple calculation, is this: if a mechanism is ρ-approximate, then *ρ is the separation threshold beyond which non-trivial welfare approximation is possible.* Precisely, a ρ-approximate mechanism extracts at least a $(1 - \rho/\eta)$ fraction of the optimal welfare when the optimal outcome is η-separated.

3.2.4 Justifying our Approach

We next establish the robustness of such an approximation bound by demonstrating its consequences for alternative definitions of approximate economic efficiency.

Not all definitions of approximate efficiency provide meaningful information for cost-sharing mechanism design. As noted in Section 3.2, for each $\beta, \gamma \geq 1$ there are simple cost-sharing problems such that no incentive-compatible, β, γ-budget-balanced mechanism obtains a non-zero fraction of the optimal welfare [32]. Thus, if we insist on adopting a relative approximation measure — by far the most ubiquitous kind across theoretical computer science — we must either change the objective function or restrict the allowable instances. We explore these two approaches in turn.

What is the "smallest perturbation" of the welfare objective that admits non-trivial approximation results? A minimal requirement for a credible reformulation is *ordinal equivalence* — for a fixed cost-sharing function and valuation profile, a subset S should be "better" than a subset T if and only if S has higher welfare than T. This requirement suggests either maximizing $f(W(S))$ for a strictly increasing function f or minimizing $f(W(S))$ for a strictly decreasing function f. Affine functions are in some sense the "least distorting" candidate functions f, and for relative approximation guarantees there is no loss of generality in considering only: (1) minimizing $-W(S) + g(C, v) = C(S) - v(S) + g(C, v)$, where the additive term $g(C, v)$ is positive and independent of S; and (2) maximizing $v(S) - C(S) + h(C, v)$ for a positive additive term $h(C, v)$. Since costs and valuations already occur positively in (1) and (2), respectively, we take g to be independent of C and h to be independent of v. The examples in [32] are strong enough to imply that no non-trivial relative approximation is possible for these objectives unless $g(C, v) \geq v(S^*)$ and $h(C, v) \geq C(S^*)$. To avoid the awkwardness of referencing the optimal solution in the objective function itself, we take $g(C, v) = v(U)$ and $h(C, v) = C(U)$, leading to the objectives of *minimizing social cost*:

$$\min_{S \subseteq U} \pi(S) \equiv -W(S) + v(U) = C(S) + v(U \setminus S); \tag{3.3}$$

and *maximizing social reward*:

$$\max_{S \subseteq U} R(S) \equiv W(S) + C(U) = v(S) + [C(U) - C(S)]. \tag{3.4}$$

These answers to our initial question conform to previous approaches to approximating mixed-sign objective functions in other application domains, including prize-collecting combinatorial optimization (e.g. [10]) and maximum-likelihood inference (e.g. [54]).

Simple algebra shows that an efficiency guarantee of the form (3.1) implies relative approximation guarantees for the social cost and social reward objectives.

Proposition 3.2.1 (From Additive to Relative Approximation) *If M is a ρ-approximate mechanism for a cost-sharing problem C, then, assuming truthful bids:*

(a) M is a $(\rho + 1)$-approximation algorithm for minimizing social cost; and

(b) M is a $1/(\rho + 1)$-approximation algorithm for maximizing social reward.

The guarantees in Proposition 3.2.1 hold even if the constants $g(C, v)$ and $h(C, v)$ in the definitions of social cost (3.3) and social reward (3.4) are reduced to $v(S^*)$ and $C(S^*)$, respectively.

A second approach to efficiency guarantees is to seek a relative approximation of welfare for the widest class of problems possible. The impossibility result from [32] applies to the excludable public good cost-sharing problem, which is an instance of almost all other problem families (recall Figure 2.1). So, restricting only the cost function is in general uninteresting for non-trivial relative welfare guarantees.

We instead study "promise problems" in which the value served by an optimal solution is bounded away from its service cost. Recall from the Introduction that an outcome S is *η-separated* for a parameter $\eta \geq 0$ if $W(S) \geq \eta \cdot C(S)$. Call a valuation profile *η-separated* if there is an η-separated efficient outcome. Simple algebra implies the following.

Proposition 3.2.2 (From Additive Approximation to Promise Problems) *If M is a ρ-approximate mechanism for a cost-sharing problem C, then, assuming truthful bids, M is a $(1 - \frac{\rho}{\eta})$-approximation algorithm for social welfare for η-separated valuation profiles.*

Thus the approximation factor ρ is the separation threshold beyond which the mechanism is guaranteed to approximate the social welfare.

Finally, recall that our critique of the social welfare objective was rooted in the fact that it fails to differentiate between "better" and "worse" cost-sharing mechanisms. Does the approximation framework detailed in this section suffer the same flaw? The answer is "no": the approximation factors (in the sense of (3.1)) of different mechanisms for a problem can vary widely (Example 2.2.12 and Proposition 4.2.12), and the best-achievable approximation factor is different for different types of cost-sharing problems (Section 5.1 and Theorem 5.3.10).

3.3 Notes

3.3.1 Beyond Impossibility: A Bayesian Approach

Dropping the voluntary participation constraint makes the impossibility result go away—there is an efficient, truthful, no-deficit mechanism (VCG with a large, bid-independent tax). However, arguably, the voluntary participation constraint is vital. Suppose that the valuation distributions of the players are known to the mechanism. Then, there is a mechanism that is efficient, truthful, budget-balanced in expectation over the valuation distributions (ex-ante budget-balanced), and that satisfies voluntary participation in expectation over the valuation distributions (ex-ante voluntary participation).

We start from the basic VCG mechanism. For every bid vector v, every player j is paid an amount $\sum_{j \neq i, j \in S^*} v_j - C(S^*)$ to align its utility to the global objective (social welfare); here S^* is the set that maximizes social welfare. So, each player enjoys an expected utility equal to $\mathrm{E}[\sum_{j \in S^*} v_j - C(S^*)]$, where the expectations are over the valuations. The mechanism suffers an expected deficit of $E[(n-1) \cdot \sum_{j \in S^*} v_j - n \cdot$

$C(S^*) + C(S^*)]$.

As this mechanism optimizes social welfare, we have that $\mathrm{E}[\sum_{i \in S^*} v_i] \geq \mathrm{E}[C(S^*)]$. So the expected deficit and the expected surplus are both non-negative. Introducing an additional bid-independent tax for each player leaves the mechanism efficient and truthful. Each player can be charged an additional amount $(n-1)/n \cdot \mathrm{E}[\sum_{j \in S^*} v_j - C(S^*)]$, resulting in ex-ante budget-balance, ex-ante voluntary participation.

Can we improve on this? d'Aspremont, Gerard-Varet [24] propose a mechanism which is efficient, Bayesian incentive compatible (truthtelling maximizes expected utility where the expectation is over the other players' valuations), and satisfies the property that the payments sum to zero; this needs an assumption that the players' valuations are distributed independently. As in VCG the idea is to pay each player an amount that, in *expectation* over other players' values, aligns its utility with the welfare objective, yielding bayesian incentive compatibility. Call this payment to player i, $X_i(v_i)$. Now, each player is charged further amount $\sum_{j \neq i} X_j(v_j)$. These additional payments balance the budget, and do not violate bayesian incentive compatibility as they are bid-independent (the terms $X_i(v_i)$ are in expectation over the other players' valuations).

Can we instead get the payments to sum to the cost of the efficient solution, $C(S^*)$? The problem is that the quantity $C(S^*)$ depends on the players' bids. Following the above paragraph we could achieve ex-ante budget-balance, i.e. recover $E[C(S^*)]$, a bid independent quantity. However, this gives us a result weaker than the one above based on VCG.

Chapter 4

Moulin Mechanisms

Until recently almost all known approximately budget-balanced cost-sharing mechanisms were Moulin mechanisms [11, 17, 37, 39, 45, 47, 48, 51, 55, 56, 68]. As Section 4.1 reviews, Moulin mechanisms are ascending auctions that consist of a problem-independent protocol together with a problem-dependent pricing oracle (i.e. cost-sharing method). Section 4.2 characterizes the efficiency loss of Moulin mechanisms in terms of a combinatorial property of the cost-sharing method, called its summability. In the next chapter we use this characterization to identify optimal Moulin mechanisms for a wide variety of cost-sharing problems. Section 4.3 summarizes the results of our analysis and compares the hardness of various cost-sharing problems. Section 4.4 shows how we can trade cost-recovery for increased efficiency.

4.1 Moulin Mechanisms

We first review *Moulin mechanisms*. A Moulin mechanism is driven by a cross-monotonic cost-sharing method (recall Section 2.5)—a function χ that assigns a non-negative *cost share* $\chi(i, S)$ for every subset $S \subseteq U$ of players and every player $i \in S$.

Given a cross-monotonic cost-sharing method χ for a cost function C, we obtain the corresponding Moulin mechanism by simulating an iterative ascending auction, with the method χ suggesting prices for the remaining players at each iteration.

Definition 4.1.1 (Moulin Mechanisms) Let U be a universe of players and χ a cross-monotonic cost-sharing method defined on U. The *Moulin mechanism* $M(\chi)$ *induced by* χ is the following.

1. Collect a bid b_i from each player $i \in U$.
2. Initialize $S := U$.
3. If $b_i \geq \chi(i, S)$ for every $i \in S$, then halt. Output the set S, the feasible solution constructed by χ, and charge each player $i \in S$ the price $p_i = \chi(i, S)$.
4. Let $i^* \in S$ be a player with $b_{i^*} < \chi(i^*, S)$.
5. Set $S := S \setminus \{i^*\}$ and return to Step 3.

The cross-monotonicity constraint ensures that the simulated auction is ascending, in the sense that the prices offered to a player progressively increase with time. This implies that the outcome of a Moulin mechanism is uniquely defined, independent of the choices made in Step 4. Also, the Moulin mechanism $M(\chi)$ clearly inherits the budget-balance factors of the cost-sharing method χ. Finally, Moulin [62] proved the following.

Theorem 4.1.2 (Strategyproofness of Moulin Mechanisms [62]) *If χ is a cross-monotonic cost-sharing method, then the corresponding Moulin mechanism $M(\chi)$ is strategyproof.*

Theorem 4.1.2 reduces the problem of designing an strategyproof (Definition 2.4.1), (β, γ)-budget-balanced (Definition 2.3.1) cost-sharing mechanism to that of designing a cross-monotonic, (β, γ)-budget-balanced cost-sharing method (see Section 2.5 for definitions).

Remark 4.1.3 Moulin mechanisms also satisfy a stronger notion of incentive compatibility called *groupstrategyproofness* [62, 63], which states that every coordinated set of false bids by a coalition should decrease the utility of some player in the coalition (or should have no effect).

By Theorem 4.1.2, the sequential and Shapley cost-sharing methods of Example 2.5.2 induce strategyproof and fully budget-balanced mechanisms for fixed-tree multicast cost-sharing problems. The impossibility result discussed in Section 3.1 implies that neither mechanism can be fully efficient. We conclude the section with concrete examples demonstrating this.

Example 4.1.4 Recall the excludable public good cost-sharing problem (Example 2.2.12). For a valuation profile v, the efficient outcome is U if $v(U) > 1$ and $\emptyset$ otherwise. The idea is to determine "worst-case valuations" for the Moulin mechanisms $M(\chi_{seq})$ and $M(\chi_{sh})$ induced by the sequential and Shapley cost-sharing methods (recall Section 2.5), respectively. We do this by setting the valuations of players to be as large as possible, subject to the constraint that the mechanism terminates with the empty outcome.

Fix a small positive number ϵ. If all players have valuation $1 - \epsilon$ and bid truthfully, then $M(\chi_{seq})$ outputs the empty outcome. If player i has valuation $1/i - \epsilon$ for $i \in \{1, 2, \ldots, n\}$ and players bid truthfully, then $M(\chi_{sh})$ outputs the empty outcome. These examples show (for arbitrarily small ϵ) that the first mechanism is no better than $\approx (n-1)$-approximate, while the second is no better than $\approx (\mathcal{H}_n - 1)$-approximate (in the sense of Equation (3.1)), where $\mathcal{H}_n = \sum_{i=1}^{n} 1/i$ denotes the nth Harmonic number.

4.2 Summability Characterizes Approximate Efficiency

This section proves that the summability of a cost-sharing method characterizes the approximate efficiency of the corresponding Moulin mechanism. After Section 4.2.1 defines summability, Section 4.2.2 proves that it upper bounds approximate efficiency and Section 4.2.3 explores the senses in which this bound is tight.

4.2.1 Summability

Intuitively, summability quantifies the efficiency loss from the overly aggressive removal of players by a Moulin mechanism. We motivate the formal definition via a generalization of Example 2.2.12, which strongly suggests that summability lower bounds the approximate efficiency of a Moulin mechanism.

Example 4.2.1 (Generic Lower Bound on Efficiency Loss) Let χ be a cross-monotonic cost-sharing method for the cost function C, defined on the universe U. Assume for simplicity that the method only assigns positive cost shares: $\chi(i, S) > 0$ for all $S \subseteq U$ and $i \in S$. Pick an ordering σ of the players of U and a subset S. Let i_ℓ denote the ℓth player and S_ℓ the first ℓ players of S with respect to σ and define the parameter $\alpha_{S,\sigma}$ by

$$\alpha_{S,\sigma} = \frac{1}{C(S)} \sum_{\ell=1}^{|S|} \chi(i_\ell, S_\ell). \tag{4.1}$$

In other words, we start with the empty set, add players of S one-by-one according to σ, and consider the cost share of the ℓth player when it is initially added. The parameter $\alpha_{S,\sigma}$ is the factor by which the sum of these cost shares overestimates the cost $C(S)$ of serving all of the players.

We claim that the Moulin mechanism $M(\chi)$ is no better than $(\alpha_{S,\sigma}-1)$-approximate for C. To see this, define the valuation v_ℓ of the ℓth player of S (according to σ) to be $\chi(i_\ell, S_\ell) - \epsilon$, where $\epsilon > 0$ is arbitrarily small. Give players of $U \setminus S$ zero valuations. The Moulin mechanism $M(\chi)$ will output the empty set. The optimal welfare is bounded below by $v(S) - C(S) \approx \alpha_{S,\sigma} \cdot C(S) - C(S) = (\alpha_{S,\sigma} - 1) \cdot C(S)$. Since valuations outside S are zero, there is an efficient outcome $S^* \subseteq S$. Further, C is non-decreasing, and hence the welfare loss of $M(\chi)$ on this valuation profile is at least $(\alpha_{S,\sigma} - 1) \cdot C(S^*)$.

The *summability* of a cost-sharing method is then defined as the worst-case ratio of the form (4.1) over choices of sets S and orderings σ.

Definition 4.2.2 (Summability) Let C and χ be a cost function and a cost-sharing method, respectively, defined on a common universe U of n players. The method χ

is α-*summable for* C for a function $\alpha : \{0, 1, 2, \ldots, n\} \to \mathcal{R}^+$ if

$$\sum_{\ell=1}^{|S|} \chi(i_\ell, S_\ell) \leq \alpha(|S|) \cdot C(S) \tag{4.2}$$

for every ordering σ of U and every set $S \subseteq U$, where S_ℓ and i_ℓ denote the set of the first ℓ players of S and the ℓth player of S (with respect to σ), respectively.

Remark 4.2.3 We define summability as a function rather than a scalar in order to parametrize our efficiency guarantees by the number k of players served in an efficient outcome (which can be much smaller than the universe size). For example, in Chapter 5 Sections 5.1 and 5.3 we establish summability bounds of the form $\alpha(|S|) \leq \mathcal{H}_{|S|}$ and $\alpha(|S|) = O(\log^2 |S|)$ for all $S \subseteq U$, which will lead to Moulin mechanisms that are $\mathcal{H}_k - 1$ and $O(\log^2 k)$-approximate, respectively.

4.2.2 Efficiency Guarantees

The central result of this section is the following efficiency guarantee for Moulin mechanisms derived from cost-sharing methods with bounded summability.

Theorem 4.2.4 (Summability Upper Bounds Approximate Efficiency) *Let C be a cost function defined on a universe U and χ a cross-monotonic, no-deficit, α-summable cost-sharing method for C. Then $M(\chi)$ is an $(\alpha(k)-1)$-approximate mechanism, where k is the size of an efficient outcome.*

Propositions 3.2.1 and 3.2.2 immediately give the following corollaries.

Corollary 4.2.5 *Let C be a cost function defined on a universe U and χ a cross-monotonic, no-deficit, α-summable cost-sharing method for C. Then $M(\chi)$ is:*

(a) an $\alpha(k)$-approximation algorithm for minimizing the social cost (3.2)*;*

(b) a $1/\alpha(k)$-approximation algorithm for maximizing the social reward;

(c) a $[1 - (\alpha(k) - 1)/\eta]$-approximation algorithm for maximizing welfare for η-separated valuation profiles.

We emphasize that Theorem 4.2.4 is completely problem-independent. Together with Definition 4.2.2, it distills the problem-specific aspect of simultaneously achieving good budget-balance and efficiency in Moulin mechanisms: designing a cross-monotonic, approximately budget-balanced cost-sharing method with small summability. The generality of Theorem 4.2.4 is evident from its application to identify optimal Moulin mechanisms for various cost-sharing problems (Section 4.3), and to quantifiable trade-offs between budget-balance and economic efficiency (Section 4.4).

We now build up to a proof of Theorem 4.2.4. Fix a cost function C defined on a universe U, a valuation profile v, and an α-summable and a no-deficit cross-monotonic cost-sharing method for C. Let σ denote the reversal of the order in which the mechanism $M(\chi)$ deletes players (in some fixed trajectory), with players in the final output set S^M ordered arbitrarily among the first $|S^M|$ positions.

A crucial tool in our proof is the following *potential function* Φ_σ, which we define for each subset $S \subseteq U$ as

$$\Phi_\sigma(S) = v(U \setminus S) + \sum_{i_\ell \in S} \chi(i_\ell, S_\ell), \tag{4.3}$$

where for every $\ell \in \{1, 2, \ldots, |S|\}$, S_ℓ denotes the first ℓ players of S and i_ℓ the ℓth player of S according to σ.

The ordering σ and the potential function Φ_σ are defined to ensure that the potential function value decreases with each iteration in our fixed trajectory of $M(\chi)$. We use this fact in the next lemma.

Lemma 4.2.6 *If S^M is the final output of $M(\chi)$ and S^* is an efficient outcome for a valuation profile v, then*

$$\Phi_\sigma(S^M \cap S^*) \leq \Phi_\sigma(S^*).$$

Proof: The idea is to delete players from S^* in the same order as $M(\chi)$ to obtain the set $S^M \cap S^*$. More precisely, order the players $i_1, i_2, \ldots, i_m$ of $S^* \setminus S^M$ according to their deletion by $M(\chi)$, with player i_1 deleted first. This ordering is consistent with σ. For a player $i_j \in S^* \setminus S^M$, let S_j denote the set of players from which it

was removed by $M(\chi)$, and let S_j^* denote $S^* \setminus \{i_1, \ldots, i_{j-1}\}$. Note that $S_j \supseteq S_j^*$ for every j. By the definition of $M(\chi)$, the valuation v_j of player i_j is less than $\chi(i_j, S_j)$. Cross-monotonicity of χ then implies that $v_j < \chi(i_j, S_j^*)$ for every player $i_j \in S^* \setminus S^M$. Using the definition of Φ_σ, we have

$$\Phi_\sigma(S^*) = \Phi_\sigma(S_1^*) > \Phi_\sigma(S_2^*) > \cdots > \Phi_\sigma(S_{m+1}^*) = \Phi_\sigma(S^M \cap S^*).$$

■

Also, by definition, summability (4.2) bounds the distance between the potential function (4.3) and the social cost (3.2) in the following sense.

Lemma 4.2.7 *For every subset $S \subseteq U$,*

$$\Phi_\sigma(S) \leq v(U \setminus S) + \alpha(|S|) \cdot C(S).$$

We are now prepared to prove Theorem 4.2.4.

Proof of Theorem 4.2.4: Fix a universe U, a cost function C, and a set v of truthful bids. Let S^* be an efficient outcome. Let χ be an α-summable, no-deficit, cross-monotonic cost-sharing method for C and S^M the output of the corresponding Moulin mechanism $M(\chi)$ for the profile v. Define the player ordering σ and the potential function Φ_σ as in (4.3). We can then derive

$$\begin{aligned}
v(U \setminus S^M) + C(S^M) &\leq v(U \setminus S^M) + \sum_{i \in S^M} \chi(i, S^M) \\
&\leq v(U \setminus S^M) + v(S^M \setminus S^*) + \sum_{i \in S^M \cap S^*} \chi(i, S^M) \\
&\leq \Phi_\sigma(S^M \cap S^*) \\
&\leq \Phi_\sigma(S^*) \\
&\leq v(U \setminus S^*) + \alpha(|S^*|) \cdot C(S^*),
\end{aligned}$$

where the first inequality follows from the no-deficit condition (2.3), the second from the fact that $\chi(i, S^M) \leq v_i$ for every $i \in S^M$, the third from the cross-monotonicity

of χ, the fourth from Lemma 4.2.6, and the fifth from Lemma 4.2.7. Rearranging terms then proves the theorem. ■

Remark 4.2.8 When the method χ is the Shapley cost-sharing method (see Section 5.1), our definition (4.3) of the potential function Φ_σ essentially coincides with that of Hart and Mas-Colell [41] for cooperative games.

Remark 4.2.9 The results of this section can be interpreted as efficiency guarantees for the noncooperative *participation games* studied by Monderer and Shapley [63] and Moulin [62]. For example, Corollary 4.2.5(a) implies that for the social cost objective (3.3), the "strong price of anarchy" [4] in such a game is at most the summability of the underlying cost-sharing method.

4.2.3 Matching Lower Bounds

We now discuss the senses in which the bound in Theorem 4.2.4 is tight. The argument in Example 4.2.1 implies the following lower bound for strictly positive cost-sharing methods.

Proposition 4.2.10 (Summability Lower Bounds Approximate Efficiency I) *Let χ be a cross-monotonic cost-sharing method for a cost-sharing problem C with universe U that is everywhere positive and at least α-summable. Then $M(\chi)$ is no better than $(\alpha(k)-1)$-approximate, where k is the size of an efficient outcome.*

The assumption that all cost shares are positive is similar to the "strong consumer sovereignty" assumption in Moulin [62], which states that each player has winning and losing bids for every fixed set of bids of the other players.

For technical reasons, summability need not lower bound the approximate efficiency of cost-sharing methods that can employ zero cost shares. To informally illustrate the issue, consider a cost-sharing problem with universe $U = \{1, 2, \ldots, n\}$ and two cost-sharing methods χ_1, χ_2 defined for the restriction of this problem to $U \setminus \{1\}$, where the summability of χ_2 is much larger than that of χ_1. Define χ on U by setting cost shares equals to those of χ_1 for sets that include the first player and

equal to those of χ_2 for sets that do not; the first player always receives a zero cost share. The summability of χ is as large as that of χ_2, but the Moulin mechanism $M(\chi)$ will never delete the first player and will therefore only assign cost shares according to the method χ_1 that has small summability. Thus the summability of χ is strictly larger than the approximate efficiency of the induced Moulin mechanism.

There is nevertheless a variant of Proposition 4.2.10 for non-positive cost-sharing methods. To state it, note that a Moulin mechanism $M(\chi)$ for a cost-sharing problem naturally induces a Moulin mechanism for each induced sub-universe U' of U (via the restriction of χ to this set U'). We say that a Moulin mechanism $M(\chi)$ is *strongly ρ-approximate* if every induced mechanism is ρ-approximate for the corresponding induced cost-sharing problem. The proof of Theorem 4.2.4 extends directly to this notion of strong approximation.

Corollary 4.2.11 *Let C be a cost function defined on a universe U and χ a cross-monotonic, no-deficit, α-summable cost-sharing method for C. Then $M(\chi)$ is a strongly $(\alpha(k) - 1)$-approximate mechanism, where k is the size of an efficient outcome.*

Summability is a valid lower bound for strong approximate efficiency, even for cost-sharing methods that use zero cost shares.

Proposition 4.2.12 (Summability Lower Bounds Approximate Efficiency II) *Let χ be a cross-monotonic cost-sharing method for a cost-sharing problem C with universe U that is at least α-summable. Then $M(\chi)$ is no better than strongly $(\alpha(k)-1)$-approximate, where k is the size of an efficient outcome.*

Proof Sketch: Choose k, a set S with $|S| = k$, and an ordering of the players of S so that $\sum_{\ell=1}^{k} \chi(i_\ell, S_\ell) \geq \alpha(k) \cdot C(S)$, where S_ℓ and i_ℓ are defined in the usual way. Obtain R from S by discarding players with $\chi(i_\ell, S_\ell) = 0$. Since χ is cross-monotonic and C is nondecreasing, the induced ordering on R satisfies $\sum_{\ell=1}^{|R|} \chi(i_\ell, R_\ell) \geq \alpha(k) \cdot C(R)$ with all cost shares positive. Mimicking Example 4.2.1 in the problem induced by R, the welfare loss of the induced Moulin mechanism is at least $(\alpha(k)-1) \cdot C(R^*)$, where R^* denotes an optimal outcome to this induced problem. ■

The construction in Example 4.2.1 also demonstrates the tightness of the alternative guarantees in Corollary 4.2.5.

Proposition 4.2.13 *Let χ be a cross-monotonic cost-sharing method for a cost-sharing problem C with universe U that is everywhere positive and at least α-summable. Then:*

(a) $M(\chi)$ is no better than an $\alpha(k)$-approximation algorithm for minimizing social cost;

(b) $M(\chi)$ is no better than a $1/\alpha(k)$-approximation algorithm for maximizing social reward;

(c) there are $(\alpha(k) - 1)$-separated valuation profiles for which $M(\chi)$ obtains zero welfare.

Similar results apply for non-positive cost-sharing methods and "strong" versions of these three types of efficiency guarantees.

4.3 Applications of the Summability Framework

Theorem 4.2.4 and Propositions 4.2.10 and 4.2.12 show that summability of the cost-sharing method χ characterizes (i.e., lower and upper bounds) the efficiency loss of the Moulin mechanism, $M(\chi)$. In the next chapter we identify optimal Moulin mechanisms for various cost-sharing problem families (see Section 2.2 for the definitions of these families). That is, for a specific cost-sharing problem family, we establish lower bounds on the summability attainable by any no-deficit, cross-monotonic cost-sharing method, and we identify cost-sharing methods that match this lower bound up to a constant factor. The table below summarizes our results. All the results are worst case with respect to the valuation profile and the cost function in the problem family; k is the size of the optimally efficient solution and recall that $\mathcal{H}_k \approx \ln k$.

Problem	Optimal Worst Case Efficiency Loss	Optimal mechanism
Marginal Cost	1 (below)	Trivial
Excludable Public Good	$\mathcal{H}_k - 1$ (Section 5.1)	Shapley [63]
Fixed Tree Multicast	$\mathcal{H}_k - 1$ (Section 5.1)	Shapley [63]
Submodular Cost	$\mathcal{H}_k - 1$ (Section 5.1)	Shapley [63]
Metric UFL	$\Theta(\log k)$ (Section 5.2)	Pal and Tardos [68]
Steiner Tree	$\Theta(\log^2 k)$ (Section 5.3)	Jain and Vazirani [48]
Steiner Forest	$\Theta(\log^2 k)$ (Section 5.4)	Könemann, Leonardi and Schäfer [55]
SSRoB	$\Theta(\log^2 k)$ (Section 5.5)	Gupta, Srinivasan and Tardos [39]

Marginal Cost cost-sharing problems admit a Moulin mechanism that is truthful, budget-balanced and efficient—the (obviously cross-monotonic and budget-balanced) cost-sharing method always offers each player a price equal to its marginal cost c_i. In contrast, the excludable public good cost-sharing problem, which models pure fixed cost, only admits a $\Theta(\log k)$-approximate no-deficit mechanism; the optimal Moulin mechanism uses the Shapley value cost-sharing method. This result extends to all submodular cost-sharing problems, including fixed tree multicast problems.

It is now tempting to think of pure fixed cost and pure marginal cost as two extremes, and extrapolate that all cost-sharing problems admit $O(\log k)$-approximate Moulin mechanisms via the Shapley value cost-sharing method. In fact, the proof of Proposition 2 from Moulin and Shenker [63] essentially shows that *if* the Shapley value cost-sharing method is cross-monotonic for a cost-sharing problem, then it has optimal summability, and the resulting Moulin mechanism is $O(\log k)$-approximate. The fly in the ointment is that the Shapley value cost-sharing method is not in general cross-monotonic for non-submodular cost-sharing problems.

Metric facility location problems admit a $O(\log k)$ approximate Moulin mechanism via the cost-sharing method of Pal and Tardos [68]. Steiner tree cost-sharing problems, and their generalizations (Steiner forest and SSRoB) only admit $\Theta(\log^2 k)$-approximate Moulin mechanisms.

See Section 4.5.2 for further applications of this summability framework to other cost-sharing problems. In particular, there exist cost-sharing problems do not admit any $O(\text{polylog}(k))$-approximate Moulin mechanisms, a fact that motivates the search for better mechanism design frameworks such as acyclic mechanisms (Chapter 6).

In conclusion, if we define the complexity of a problem family by the optimal worst-case approximation of efficiency achievable by a Moulin mechanism (recall Section 1.4.3), problem families fall into four classes. There are those with $\Theta(1)$ complexity like edge-cover [45] and marginal cost. Others have $\Theta(\log k)$ complexity like metric UFL and submodular cost. Some have $\Theta(\log^2 k)$ complexity like Steiner tree, Steiner forest, SSRoB and MRoB. Finally, there are those with $\Omega(poly(k))$ complexity like Vertex cover and Set cover.

4.4 Budget-Balance vs. Economic Efficiency Trade-Offs

No-deficit Moulin mechanisms are inefficient because of their overzealous removal of players that cannot pay their cost share (cf., Examples 2.2.12 and 4.2.1). This suggests a possible trade-off between budget-balance and economic efficiency: if we relax the requirement that the prices charged cover the cost incurred, then a Moulin mechanism can employ smaller cost shares and reduce the worst-case efficiency loss from regrettable player deletions. This section extends the efficiency guarantees of Section 4.2 to mechanisms that need not cover the incurred cost, and uses these extensions to quantify the trade-off between budget-balance and economic efficiency in Moulin mechanisms. In particular, we show that relaxing budget-balance permits mechanisms with strictly better efficiency guarantees than those possible for no-deficit Moulin mechanisms.

Recall that a Moulin mechanism is (β, γ)-*budget-balanced* if the sum of the prices charged is at least $1/\gamma$ and at most β times the incurred service cost. When $\gamma > 1$, Moulin mechanisms can suffer efficiency loss from the unjustified service of players with low valuations. (See Example 4.4.4 below.) For this reason, an efficiency

guarantee for a (β, γ)-budget-balanced Moulin mechanism must reference both the parameter γ and the summability of its underlying cost-sharing method. We provide such a guarantee next.

Theorem 4.4.1 *Let C be a cost function defined on a universe U and χ a cross-monotonic, (β, γ)-budget-balanced, α-summable cost-sharing method for C. Let S^M and S^* denote the outcome chosen by $M(\chi)$ and an optimal outcome, respectively, for a valuation profile v. Then,*

$$W(S^*) - W(S^M) \leq (\alpha(|S^*|) - 1 + \beta(\gamma - 1)) \cdot C(S^*) + (\gamma - 1) \cdot v(S^M \setminus S^*).$$

Proof: Define an ordering σ on U and a potential function Φ_σ as in the proof of Theorem 4.2.4. By following the steps in that proof and using the (β, γ)-budget-balance of χ, we obtain

$$\begin{aligned}
v(U \setminus S^M) + C(S^M) &\leq v(U \setminus S^M) + \gamma \sum_{i \in S^M} \chi(i, S^M) \\
&\leq v(U \setminus S^M) + \gamma \cdot v(S^M \setminus S^*) + \gamma \sum_{i \in S^M \cap S^*} \chi(i, S^M) \\
&\leq \Phi_\sigma(S^M \cap S^*) + (\gamma - 1) \cdot v(S^M \setminus S^*) + (\gamma - 1) \sum_{i \in S^M \cap S^*} \chi(i, S^M \cap S^*) \\
&\leq \Phi_\sigma(S^*) + (\gamma - 1) \cdot v(S^M \setminus S^*) + (\gamma - 1)\beta \cdot C(S^*) \\
&\leq (\alpha(|S^*|) + \beta(\gamma - 1)) \cdot C(S^*) + v(U \setminus S^*) + (\gamma - 1) \cdot v(S^M \setminus S^*).
\end{aligned}$$

Rearranging terms proves the theorem. ■

Like Theorem 4.2.4, the guarantee on additive welfare loss in Theorem 4.4.1 can be interpreted in several different ways. We mention only the cleanest such interpretation, in terms of minimizing the social cost objective (3.2).

Corollary 4.4.2 *Let C be a cost function defined on a universe U and χ a cross-monotonic, (β, γ)-budget-balanced, α-summable cost-sharing method for C. Then $M(\chi)$ is a $(\max\{\alpha(k) + \beta(\gamma - 1), \gamma\})$-approximation algorithm for the social cost objective, where k is the size of an efficient outcome.*

We can use Theorem 4.4.1 and Corollary 4.4.2 to trade cost-recovery for increased efficiency. For example, an easy consequence of the upper bound on summability of Shapley cost-shares from Section 5.1 is that for a submodular cost-sharing problem with n players, dividing the corresponding Shapley cost shares by an $\sqrt{\mathcal{H}_n}$ factor yields a $(1, \sqrt{\mathcal{H}_n})$-budget-balanced and $\sqrt{\mathcal{H}_n}$-summable cost-sharing method.

Corollary 4.4.2 implies the following guarantee for the induced Moulin mechanism (the *scaled Shapley mechanism*).

Corollary 4.4.3 *For every n-player submodular cost-sharing problem, the scaled Shapley mechanism is $(1, \sqrt{\mathcal{H}_n})$-budget-balanced and a $(2\sqrt{\mathcal{H}_n} - 1)$-approximation algorithm for the social cost objective.*

The efficiency guarantee in Corollary 4.4.3 is better than the best possible for no-deficit Moulin mechanisms (see Section 5.1). There are analogous improvements possible for the other cost-sharing problems, also via scaling the no-deficit cost-sharing method with the optimal summability.

Corollary 4.4.3 is optimal in the following senses. First, a simple example shows that a Moulin mechanism that is no better than (β, γ)-budget-balanced is no better than a γ-approximation algorithm for the social cost objective.

Example 4.4.4 Let χ be a cross-monotonic cost-sharing method for a cost function C defined on a universe U, and suppose that χ is no better than (β, γ)-budget-balanced for C. By definition, there is a subset $S \subseteq U$ of players with $\sum_{i \in S} \chi(i, S) \leq C(S)/\gamma$. Give each player $i \in S$ the valuation $\chi(i, S)$ and other players zero valuations. With this valuation profile, the Moulin mechanism $M(\chi)$ outputs a set containing all of the players of S, with social cost at least $C(S)$. The optimal social cost is at most that of the empty set, which is at most $C(S)/\gamma$.

Second, the lower bound proofs in Section 5.1 and Theorem 5.3.10 extend easily to show that all (β, γ)-budget-balanced Moulin mechanisms for the excludable public good and the Steiner tree cost-sharing problems are $\Omega((\log k)/\gamma)$- and $\Omega((\log^2 k)/\gamma)$-approximation algorithms for the social cost, respectively. Thus no Moulin mechanism, no matter how poor its budget-balance, obtains an $o(\sqrt{\log k})$-approximation of

the social cost for submodular or UFL cost-sharing problems or an $o(\log k)$-approximation of the social cost for Steiner tree, Steiner Forest of SSRoB cost-sharing problems.

4.5 Notes

4.5.1 Prior Work on the Efficiency of Moulin Mechanisms

The sole previous work on quantifying efficiency loss in no-deficit cost-sharing mechanisms is Moulin and Shenker [63], which studies submodular cost-sharing problems. Their results successfully rank different no-deficit mechanisms for an arbitrary but fixed submodular cost-sharing problem according to worst-case efficiency loss (see also Section 5.1). However, it is not obvious how to use their efficiency loss measure to make comparisons between different cost-sharing problems. Additionally, the approach in [63] has not yet been extended beyond submodular cost-sharing problems, and many problems studied in the computer science literature fall outside of this class [11, 17, 37, 39, 45, 47, 48, 51, 55, 56, 68].

4.5.2 Other Applications of the Summability Framework

We mention some applications of the summability based framework for measuring the efficiency loss of Moulin mechanisms, besides the ones mentioned in Section 4.3. All of the results are worst-case approximation bounds over valuation profiles and cost functions in the problem family; k is the size of the optimally efficient solution.

Problem	Summability Bounds	Paper
Identical Machines	$\Theta(\log k)$	See Brenner and Schäfer [17]
Related Machines	$\Theta(\log k)$,	See Bleischwitz and Schoppmann [15]
Prize Collecting	$\Theta(\log^2 k)$	See Gupta et al. [37]
Steiner Forest		
MRoB	$\Theta(\log^2 k)$	See Roughgarden and Sundararajan [71]
Vertex Cover	$\Omega(k^{1/3})$	See Immorlica et al. [45]
Set Cover	$\Omega(\sqrt{k})$	See Immorlica et al. [45]

Brenner and Schäfer [17] identify an optimal mechanism for scheduling identical machines. Bleischwitz and Schoppmann [15] generalizes this to related machines. Gupta et al. [37] and Roughgarden and Sundararajan [71] identify optimal mechanisms for two generalizations of the Steiner tree cost-sharing problem. Immorlica et al. [45] in conjunction with Theorem 4.2.4 shows that not all subadditive cost functions admit Moulin mechanism with polylog approximate efficiency: There are no $o(k^{1/3})$-budget balanced Moulin mechanism for Vertex cover, or $o(\sqrt{k})$-budget balanced Moulin mechanism for set-cover [45]. Further, an easy consequence of cross-monotonicity is that the budget-balance factor is a lower bound on summability and hence by Theorem 4.2.4 these lower bounds apply to the efficiency approximations achievable for these cost-sharing problems.

4.5.3 Groupstrategyproofness

Recall from Section 2.4 the definition of groupstrategyproofness. As mentioned in Remark 4.1.3, Moulin mechanisms are groupstrategyproof. In fact, the converse is almost true: Theorem 4.2 from Immorlica et al. [45] states that the only groupstrategyproof mechanisms that satisfy an additional continuity condition and a strong version of consumer sovereignty (i.e. every player has a winning and losing bid, no matter what the bids of the other players), are Moulin mechanisms. Alternatively, Theorem 2 from Moulin [62] (also Proposition 1 from Moulin and Shenker [63]) states that the only groupstrategyproof, budget-balanced, voluntary mechanisms that satisfy an additional natural technical condition (consumer sovereignty) are Moulin mechanisms.

4.5.4 Multiple Levels of Service

Most of the literature on Moulin mechanisms focuses on a binary notion of service. A notable exception is Bleischwitz and Schoppmann [14], which generalizes Moulin mechanisms to settings with multiple levels of service, and applies it to generalizations of the UFL and Steiner tree cost-sharing problems where players demand redundancy in connectivity; they quantify efficiency loss of the mechanisms they propose using a generalization of our summability framework.

Chapter 5

Summability Bounds

Theorem 4.2.4 and Propositions 4.2.10 and 4.2.12 from the previous chapter show that summability of the cost-sharing method χ characterizes (lower bounds and upper bounds) the efficiency loss of the Moulin mechanism, $M(\chi)$. In this chapter we identify optimal Moulin mechanisms for various cost-sharing problem families (recall Section 2.2). That is, for a specific cost-sharing problem family, we establish lower bounds on the summability attainable by any no-deficit, cross-monotonic cost-sharing method. We then identify cost-sharing methods that match this lower bound, up to constant factors.

Section 5.1 studies submodular cost-sharing problems (Example 2.2.2), Section 5.2 studies metric UFL problems (Example 2.2.4), Section 5.3 studies Steiner tree problems (Example 2.2.7), Section 5.4 studies Steiner forest problems (Example 2.2.9) and Section 5.5 studies SSRoB problems (Example 2.2.10).

5.1 Submodular Cost-Sharing Problems

We show how existing results of Moulin and Shenker [63] imply approximation bounds for submodular cost-sharing problems, and also derive identical bounds using the summability approach of Section 4.2.

We first recall a mechanism based on a generalization of the Shapley method χ_{sh} described in Example 2.5.2. Let C be a submodular cost function (recall (2.2))

defined on a player set U. The *Shapley cost share* $\chi_{sh}(i, S)$ of player i in the set S is defined as follows. For a permutation σ of the players of S, let $\Delta_\sigma(i)$ denote the increase $C(A \cup \{i\}) - C(A)$ in cost due to i's arrival, where $A \subseteq S$ is the set of players that precede i in σ. The Shapley cost share $\chi_{sh}(i, S)$ is then the expected value of $\Delta_\sigma(i)$, where the expectation is over the (uniform at random) choice of σ. As is well known and easily checked, Shapley cost shares are 1-budget-balanced, and are cross-monotonic when the function C is submodular. The corresponding Moulin mechanism $M(\chi_{sh})$ is called the *Shapley mechanism for* C [63].

Moulin and Shenker [63, Proposition 2] proved that, for every submodular cost function C defined on a universe U of n players, the corresponding Shapley mechanism minimizes the worst-case (over valuation profiles) additive welfare loss, over all 1-budget-balanced Moulin mechanisms. Precisely, they showed that this worst-case welfare loss, compared to an efficient solution, is at least

$$\sum_{S \subseteq U} \frac{(|S|-1)!(n-|S|)!}{n!} C(S) - C(U) \tag{5.1}$$

for every 1-budget-balanced Moulin mechanism, with equality holding for the Shapley mechanism. Since $C(S) \leq C(U)$ for every $S \subseteq U$, the worst-case welfare loss for the Shapley mechanism is at most

$$C(U) \cdot \left(\sum_{|S|=1}^{n} \binom{n}{|S|} \frac{(|S|-1)!(n-|S|)!}{n!} \right) - C(U) = C(U) \cdot \left(\sum_{|S|=1}^{n} \frac{1}{|S|} \right) - C(U)$$

$$= (\mathcal{H}_n - 1) \cdot C(U),$$

and thus this mechanism is at most $(\mathcal{H}_n - 1)$-approximate for every submodular cost-sharing problem. Since $C(S) = C(U)$ for every non-empty set $S \subseteq U$ in the excludable public good problem (Example 2.2.12), it provides a matching lower bound: there is a submodular cost-sharing problem for which every 1-budget-balanced Moulin mechanism is no better than $(\mathcal{H}_n - 1)$-approximate.

These bounds can also be derived from summability arguments, and in the process extended to all no-deficit (not necessarily 1-budget-balanced) Moulin mechanisms.

The lower bound is again for the special case of an excludable public good with n players. For every Moulin mechanism $M(\chi)$ induced by a cross-monotonic, no-deficit cost-sharing method χ, we can inductively order the players $1, 2, \ldots, n$ such that $\chi(i, \{i, i+1, \ldots, n\}) \geq 1/(n-i+1)$ for every i. Defining valuations as in Example 4.2.1 then shows that $M(\chi)$ is no better than $(\mathcal{H}_n - 1)$-approximate. Formally we have:

Proposition 5.1.1 *For every $\beta \geq 1$, no β-budget-balanced Moulin mechanism is better than $(\mathcal{H}_k - 1)$-approximate for the excludable public good cost-sharing problem, where k is the size of an efficient outcome. Here $\mathcal{H}_m = 1 + 1/2 + \ldots 1/m$ is the mth harmonic number.*

To obtain an upper bound of $(\mathcal{H}_k - 1)$ for the approximation factor of the Shapley mechanism, where k is the number of players served in an optimal solution, fix a submodular cost function C with players U, with χ_{sh} the corresponding Shapley cost-sharing method. By Definition 4.2.2 and Theorem 4.2.4, we only need to show that

$$\sum_{\ell=1}^{|S|} \chi_{sh}(i_\ell, S_\ell) \leq \mathcal{H}_{|S|} \cdot C(S) \tag{5.2}$$

for every $S \subseteq U$ and ordering σ of U, where S_ℓ and i_ℓ are defined in the usual way. A remarkable result of Hart and Mas-Colell [41, Footnote 7], a variant of which is also used in [63] to establish (5.1), implies that the left-hand side of (5.2) is *independent of the ordering* induced by σ on the players of S. (This can also be established directly by a counting argument.) Choosing an ordering of the players of S uniformly at random, the facts that C is nondecreasing and χ_{sh} is 1-budget-balanced imply that $E[\chi_{sh}(i_\ell, S_\ell)] = E[C(S_\ell)]/\ell \leq C(S)/\ell$ for each ℓ. Summing over all ℓ and using the linearity of expectation shows that the expected value under a random ordering (and hence the value under every ordering) of the left-hand side of (5.2) is at most $\mathcal{H}_{|S|} \cdot C(S)$, completing the argument.

Remark 5.1.2 While the approximation bound of $\mathcal{H}_k - 1$ is tight for an excludable public good, both of the derivations above can obviously be sharpened for particular cost functions. For example, for the cost function $C(S) = |S|^d$ with $d \in (0, 1]$ and n

large, the Shapley mechanism remains optimal and is roughly $(\frac{1}{d} - 1)$-approximate. See Brenner and Schäfer [17] for a related discussion.

Remark 5.1.3 We note in passing that Shapley cost shares are generally hard to compute, in myriad senses, even for monotone and submodular cost functions [8]. The following randomized variant of the Shapley cost-sharing method is polynomial-time computable, cross-monotonic with probability 1, and arbitrarily close to $\mathcal{H}_k$-summable with high probability: choose in advance a sufficiently large polynomial number of player permutations uniformly at random, and estimate every expectation of the form $E[\Delta_\sigma(i)]$ by the average value of $\Delta_\sigma(i)$ over the randomly chosen permutations.

5.2 Metric Facility Location Cost-Sharing Problems

In this section we identify an optimal Moulin mechanism for metric uncapacitated facility location (UFL) cost-sharing problem defined by Example 2.2.4. We seek a no-deficit Moulin mechanism for UFL with the best-possible approximate efficiency. Section 5.2.1 describes the previously proposed Pál and Tardos [68] mechanism for the UFL cost-sharing problem. Section 5.2.2 bounds the efficiency of this mechanism and shows that this mechanism is optimal, up to constants.

5.2.1 The PT UFL Mechanism

Pál and Tardos [68] showed that every UFL cost function admits a 3-budget-balanced (in the sense of Section 2.3) cross-monotonic cost-sharing method χ_{PT}. We call this the *PT method*, and the induced Moulin mechanism the *PT mechanism*. Immorlica et al. [45] shows that no cross-monotonic cost-sharing methods can be $(3-\epsilon)$-budget-balanced, for any $\epsilon > 0$.

Given an instance of the UFL cost-sharing problem defined by players U, facilities F with opening costs f, and a metric c on $U \cup F$, the corresponding PT cost-sharing

method χ_{PT} is defined as follows. Fix an arbitrary subset $S \subseteq U$ of players. First, there is a notion of *time*, which is initially 0 and increases at a uniform rate. At a time $t \geq 0$, we associate with each player $i \in S$ a ball of radius t with center i, where distances are with respect to the given metric c; a ball centered at player i with radius r is defined as the set of facilities at a distance at most r from i. Once a ball includes a facility $q \in F$, the subsequent growth of this ball contributes toward "filling" this facility. Once these contributions equal the facility's opening cost f_q, we declare the facility q to be *full*. Precisely, facility q becomes full at the time t^q defined by the equation

$$\sum_{i \in S} \max\{0, t^q - c(q,i)\} = f_q. \tag{5.3}$$

The PT cost share $\chi_{PT}(i,S)$ of a player i in S, as defined by [68], is the length of time during which there is no full facility in player i's ball. We multiply these cost-shares by a factor 3 to ensure that the cost of the constructed solution is recovered. Proofs from [68] along with this scaling show that the PT cost-sharing method is 3-budget-balanced, and cross-monotone.

5.2.2 The PT mechanism is $O(\log k)$-approximate

In this section we bound the worst-case efficiency of the PT mechanism. We start by identifying a lower bound. We first note that the excludable public good cost-sharing problem defined by Example 2.2.12 is an instance of a UFL cost-sharing problem: There is single facility q with opening cost 1 and the metric $c(q,i)$ is 0 for all $i \in U$. Thus, by Proposition 5.1.1 every no-deficit Moulin mechanism is at least $\mathcal{H}_k$-approximate. The main result of this section is that the PT mechanism matches this lower bound, up to a constant factor. Recall that $\mathcal{H}_k \approx \ln k$).

Theorem 5.2.1 *The PT mechanism is $O(\log k)$-approximate for every UFL cost-sharing problem, where k is the size of an efficient outcome.*

By Theorem 4.2.4 it suffices to show that PT cost-shares are $O(\log k)$-summable (Definition 4.2.2). This will follow from Lemma 5.2.3, which shows that single-facility instances supply worst-case examples for the summability of the PT cost-sharing

method, and Lemma 5.2.4, which bounds the summability of single-facility instances. We bulld up towards the proof of Lemma 5.2.3. We establish that increasing distances between demands and facilities can only increase PT cost shares.

Lemma 5.2.2 *Let $\mathcal{I}$ and $\mathcal{I}'$ denote two instances of uncapacitated facility location with the same player set U, facility sets F, and facility opening costs f. Assume that the metric c' on $U \cup F$ of the second instance dominates the first, in that $c'(i,j) \geq c(i,j)$ for every $i, j \in U \cup F$. Let χ_{PT} and χ'_{PT} be the PT cost-sharing methods corresponding to $\mathcal{I}$ and $\mathcal{I}'$, respectively. Then $\chi'_{PT}(i,S) \geq \chi_{PT}(i,S)$ for every set $S \subseteq U$ and player $i \in S$.*

Proof: Fix a set $S \subseteq U$. First, equation (5.3) immediately implies that facilities can only become full later in the instance $\mathcal{I}'$ than in $\mathcal{I}$. Second, note that the PT cost share of a player $i \in S$ is defined as thrice the earliest time at which a full facility lies in player i's ball. It follows that PT cost shares for $\mathcal{I}'$ can only be larger than those for $\mathcal{I}$. ■

This monotonicity property allows us to argue that in worst-case UFL instances, players are partitioned into non-interacting groups, each clustered around one facility.

Lemma 5.2.3 *For any monotone function $\alpha : \{0, 1, 2, \ldots n\} \to R^*$, if the PT cost-sharing method is α-summable for all single facility UFL cost functions, then it is α-summable for every UFL cost function.*

Proof: Fix an arbitrary UFL cost function C^0, given by a player set U, a facility set F, facility opening costs f, and a metric c^0 on $U \cup F$. Suppose that single facility instances are α-summable. We must show for an arbitrary set $S \subseteq U$, an arbitrary ordering σ of the players in S that:

$$\sum_{\ell=1}^{|S|} \chi^0_{PT}(i_\ell, S_\ell) \leq \alpha(|S|) \cdot C^0(S)$$

Here S_ℓ and i_ℓ denote the set of the first ℓ players and the ℓth player of S in the ordering σ, respectively.

Fix an optimal solution to the facility location instance induced by the players in S. Let F^* denote the facilities opened by this solution, and let S^q denote the players of S assigned to the facility $q \in F^*$ in this solution. This solution has cost

$$C^0(S) = \sum_{q \in F^*} \left(f_q + \sum_{i \in S^q} c^0(i,q) \right).$$

We obtain the instance $\mathcal{I}^1$ from $\mathcal{I}^0$ by modifying distances as follows. If a player $i \in S$ is assigned to the facility $q \in F^*$ in the fixed optimal solution to the instance induced by S, then set $c^1(i,q) = c^0(i,q)$. All other distances between players and facilities are set to a sufficiently large number. Let C^1 denote the cost function corresponding to $\mathcal{I}^1$ and χ^1_{PT} the corresponding PT cost-sharing method. By construction, $C^1(S) = C^0(S)$. Lemma 5.2.2 implies that

$$\sum_{\ell=1}^{|S|} \chi^1_{PT}(i_\ell, S_\ell) \geq \sum_{\ell=1}^{|S|} \chi^0_{PT}(i_\ell, S_\ell) \tag{5.4}$$

Thus it suffices to prove the claim that $\sum_{\ell=1}^{|S|} \chi^1_{PT}(i_\ell, S_\ell) \leq \alpha(|S|) \cdot C^1(S)$

The instance $\mathcal{I}^1$ is essentially a collection of independent single-facility instances. To make this precise, for a facility $q \in F^*$, let $\mathcal{I}^q$ denote the facility location instance with player set S^q, facility set $\{q\}$, opening cost f_q, and with distances inherited from $\mathcal{I}_1$. Let C^q and χ^q_{PT} denote the corresponding cost function and PT cost-sharing method, respectively. By construction, we have

$$C^1(S) = \sum_{q \in F^*} C^q(S^q). \tag{5.5}$$

Further, our definition of the distances in $\mathcal{I}^1$ ensures that the PT cost share $\chi^1(i_\ell, S_\ell)$ for a player $i_\ell \in S^q$ is a function only of the set of players of S^q that precede i_ℓ in the ordering σ. (Only players of S^q contribute to the filling of facility q.) Because of this, we have

$$\sum_{\ell=1}^{|S|} \chi^1_{PT}(i_\ell, S_\ell) = \sum_{q \in F^*} \sum_{p=1}^{|S^q|} \chi^q_{PT}(i_p, S^q_p), \tag{5.6}$$

where S_p^q and i_p denote the first p players and the pth player of S_q, respectively, according to σ.

Finally, as PT cost-shares are by assumption α-summable for every single facility instance, and because α is non-decreasing, for every $q \in F^*$ we have that:

$$\sum_{p=1}^{|S^q|} \chi_{PT}^q(i_p, S_p^q) \leq \alpha(|S|) \cdot C^q(S^q) \tag{5.7}$$

Equations (5.5), (5.6), (5.7) easily prove the claim. ■

We can now focus our attention on single facility instances. The next lemma bounds the summability of the PT cost-sharing method on such instances.

Lemma 5.2.4 *The PT cost-sharing method is $3 \cdot \mathcal{H}_k$-summable for every single facility UFL cost-funciton. Here $\mathcal{H}_m = 1 + 1/2 + \ldots 1/m$ is the mth harmonic number.*

Proof: Consider an arbitrary facility location problem with a single facility q and a player set U. Let f_q denote the opening cost for q and $c(i, q)$ denote the distance between the player i and the facility q. For a set S of players, let $c(S, q)$ denote the sum of the distances of the demands in the set S to the facility q: $c(S, q) = \sum_{i \in S} c(i, q)$. Because there is only one facility, we have $C(S) = f_q + c(S, q)$ for every $S \subseteq U$.

Fix an arbitrary ordering σ of the players in U and a subset $S \subseteq U$. Let S_ℓ be the first ℓ players of S in the ordering and i_ℓ the ℓth player. By Definition 4.2.2, we need to show that

$$\sum_{\ell=1}^{|S|} \chi_{PT}(i_\ell, S_\ell) \leq H_{|S|} \cdot C(S). \tag{5.8}$$

Fix an $\ell \in \{1, 2, \ldots, |S|\}$ and consider the run of the PT algorithm on the set S_l. Recall from the definition of PT cost shares that there is a time t at which the facility q becomes full. There are two cases. If $c(i_\ell, q) > t$, then we have that:

$$\chi(i_\ell, S_\ell) = 3 \cdot c(i_\ell, q) \tag{5.9}$$

This is because, by the time q lies in player i_ℓ's ball, it is already full. If $c(i_\ell, q) \leq t$, then the PT cost share $\chi_{PT}(i_\ell, S_\ell)$ is $3 \cdot t$—by the time facility q is full, it already lies

in player i_ℓ's ball.

In the latter case, the growth of player i_ℓ's ball contributes toward filling the facility q during the time interval $[c(i_\ell, q), t]$. Since q is the only facility and it is not full during this time, the cost shares of all of the players in S_ℓ are accumulating during this time. All but $c(i, q)$ of the increase in the cost share of a player $i \in S_{\ell-1}$ during this time must contribute toward the filling of facility q. Thus,

$$[t - c(i_\ell, q)] + \sum_{i \in S_{\ell-1}} [t - c(i_\ell, q) - c(i, q)] \leq f_q.$$

Rewriting,

$$t - c(i_\ell, q) \leq \frac{1}{\ell} \left(f_q + \sum_{i \in S_{\ell-1}} c(i, q) \right). \tag{5.10}$$

Combining Equations (5.9) and (5.10), we can bound the PT cost share $\chi_{PT}(i_\ell, S_\ell)$ of player i_ℓ by

$$\chi_{PT}(i_\ell, S_\ell) \leq 3 \cdot c(i_\ell, q) + \frac{3}{\ell} \left(f_q + \sum_{i \in S_{\ell-1}} c(i, q) \right).$$

Summing over all $\ell \in \{1, 2, \ldots, |S|\}$ then gives

$$\begin{aligned}
\sum_{\ell=1}^{|S|} \chi_{PT}(i_\ell, S_\ell) &\leq \sum_{\ell=1}^{|S|} \left[3 \cdot c(i_\ell, q) + \frac{3}{\ell} \left(f_q + \sum_{i \in S_{\ell-1}} c(i, q) \right) \right] \\
&= f_q \sum_{\ell=1}^{|S|} \frac{3}{\ell} + \sum_{\ell=1}^{|S|} 3 \cdot c(i_\ell, q) \cdot \left(1 + \sum_{p=\ell+1}^{|S|} \frac{1}{p} \right) \\
&\leq 3 \cdot \mathcal{H}_k \cdot \left(f_q + \sum_{\ell=1}^{|S|} c(i_\ell, q) \right) \\
&= 3 \cdot \mathcal{H}_k \cdot C(S),
\end{aligned}$$

completing the proof of (5.8) and hence the lemma. ■

Though we are primarily interested in studying the worst-case approximation of efficiency over facility location instances, we point out that when facility opening costs are zero, we have an instance of a marginal cost cost-sharing problem (Example 2.2.13), for which we have a truthful, budget-balanced, optimally efficient mechanism (recall Section 4.3). On the flip side, Proposition 5.1.1 shows that the worst-case efficiency is achieved by a single facility instance with zero connetion costs.

5.3 Steiner Tree Cost-Sharing Problems

This section uses the summability framework of Section 4.2 to prove matching upper and lower bounds on the best-possible approximate efficiency of no-deficit Moulin mechanisms for Steiner tree cost-sharing problems (Example 2.2.7). Both the upper and lower bounds are much more intricate than those for submodular or UFL cost-sharing problems. Section 5.3.1 reviews a mechanism of Jain and Vazirani [47], and Section 5.3.2 proves that this mechanism is $O(\log^2 k)$-approximate for all Steiner tree problems. Section 5.3.3 proves that this mechanism is optimally approximately efficient (up to constant factors).

5.3.1 The JV Steiner Tree Mechanism

Recall that a Steiner tree cost-sharing problem (Example 2.2.7) is defined via an undirected graph $G = (V, E)$ with nonnegative edge costs, a root vertex t, and a set U of players that inhabit the vertices of G. The cost $C(S)$ of a subset $S \subseteq U$ is defined as the cost of an optimal Steiner tree of G that spans $S \cup \{t\}$. Such cost functions are not generally submodular, and the corresponding Shapley cost-sharing methods are not generally cross-monotonic. Several researchers have designed 2-budget-balanced and cross-monotonic Steiner tree cost-sharing methods [47, 48, 55], and no cross-monotonic method can have better budget-balance [45, 55]. We work with the first of these, designed by Jain and Vazirani [47].

Put succinctly, the *JV cost-sharing method* χ_{JV} for a Steiner tree problem is defined by equally sharing the dual growth that occurs in Edmonds's primal-dual

branching algorithm [29]. In more detail, this method works as follows.

First, given a subset $S \subseteq U$, form a complete directed graph $H = (V_H, A_H)$. The vertices V_H are t and the vertices of G that contain at least one player of S. The cost c_{uw} of an arc (u, w) of H equals the length of a minimum-cost u-w path in G. (Since G is undirected, arcs (u, w) and (w, u) of H have equal cost.) We then define both a feasible Steiner tree and cost shares using *Edmonds's algorithm*, as follows. Initialize a timer to time $\tau = 0$ and increase time at a uniform rate. Initialize a subset $F \subseteq A_H$ to $\emptyset$. At every moment in time, the algorithm increases at unit rate a variable y_A for every weakly connected component A of (V_H, F) other than the one containing the root t. When an inequality of the form

$$\sum_{A \subseteq V_H \,:\, u \in A, w \notin A} y_A \leq c_{uw}$$

first holds with equality, the corresponding arc (u, w) is added to F and the algorithm continues. (When this occurs for several inequalities simultaneously, all of the corresponding arcs are added.) When the algorithm terminates, the graph (V_H, F) contains a directed path from every vertex to the root t. To obtain a subgraph of G that spans t and the players of S, select an arbitrary branching B (a spanning tree directed toward t) of (V_H, F) and output the union of the minimum-cost paths of G that correspond to the arcs of B. To obtain cost shares, let u_i denote the vertex of V_H at which player i resides and set

$$\chi_{JV}(i, S) = \sum_{A \subseteq V_H \,:\, u_i \in A} \frac{y_A}{\kappa(A)},$$

where $\kappa(A)$ is the population of S in A. Equivalently, cost shares can be defined in tandem with the above algorithm: whenever a variable y_A is increased, this increase is distributed equally among the cost shares of the players of S contained in A.

Jain and Vazirani [47] proved that the method χ_{JV} is cross-monotonic and 2-budget-balanced in the sense of the inequalities (2.3). The next proposition summarizes the additional properties of the JV cost-sharing method that are important for bounding its summability. To state them, we say that a player $i \in S$ is *active at time*

τ in Edmonds's algorithm if it is not in the same weakly connected component as the root t at time τ. The *activity time* of a player is the latest moment in time at which it is active. The notation $d_G(i, j)$ refers to the minimum cost of an i-j path in the graph G.

Proposition 5.3.1 *Let $G = (V, E)$ be a Steiner tree instance with root t and player set S.*

(a) While player i is active in Edmonds's algorithm and belongs to a component with $m-1$ other (active) players, it accumulates an instantaneous cost share of $\frac{dt}{m}$. The final JV cost share for player i equals the integral of its instantaneous cost share up to its activity time.

(b) The activity time of a player $i \in S$ in Edmonds's algorithm is at most the length of a shortest i-t path in G.

(c) For every pair $i, j \in S$, by the time $d_G(i, j)$ in Edmonds's algorithm, players i and j are in the same weakly connected component.

Proposition 5.3.1 follows easily from the definition of Edmonds's algorithm and the JV cost shares.

5.3.2 The JV Mechanism is $O(\log^2 k)$-Approximate

The main result in this section is that, for every Steiner tree cost-sharing problem, the Moulin mechanism induced by the corresponding JV method is $O(\log^2 k)$-approximate.

Theorem 5.3.2 *There are constants $a, b > 0$ such that the following statement holds: for every Steiner tree cost-sharing problem, the Moulin mechanism induced by the corresponding JV method is $(a \log^2 k + b)$-approximate, where k is the size of an efficient outcome.*

Next we discuss our high-level proof approach. By Theorem 4.2.4, it suffices to show that

$$\sum_{\ell=1}^{|S|} \chi_{JV}(i_\ell, S_\ell) = O(\log^2 |S|) \cdot C(S)$$

for every Steiner tree problem C with JV method χ_{JV}, every subset S of players, and every ordering of the players (where i_ℓ and S_ℓ are defined in the usual way). The challenge in proving this stems from the adversarial ordering of the players (cf., Example 5.3.9 below). Our proof of Theorem 5.3.2 resolves this difficulty with the following three-step approach. First, we build a tree T on the player set, with the same root as the given Steiner tree problem, that intuitively "inverts" an arbitrary ordering so that players closer to the root in T appear earlier in the ordering than their descendants. We pay a price for this inversion: the sum of the edge costs of T is $O(\log |S|)$ times the cost of an optimal Steiner tree.

In the second step we define "artificial cost shares" for the players. These cost shares will approximate the JV cost shares of players in G, but it will also be straightforward to upper bound their sum. More precisely, we define the artificial cost share of the ith player (according to the given adversarial ordering) as its Shapley cost share in the tree T, assuming that precisely the first i players are present. By inequality (5.2), the sum of these artificial cost shares is at most $\mathcal{H}_{|S|}$ times the sum of the edge costs of T, which in turn is $O(\log^2 |S|)$ times the cost of an optimal Steiner tree in G.

In the third step, we prove that Shapley cost shares in T approximate JV cost shares in G: for every player, the former is at least a constant fraction of the latter. We feel that this final step is by far the most surprising, as it relates two sets of cost shares that are defined by different methods as well as in different graphs. This final step uses properties of both the JV dual growth process and the edge cost structure in the tree T.

We now supply the details. Fix a Steiner tree cost-sharing problem with universe U, graph $G = (V, E)$ with edge costs c, and root vertex $t \in V$. We begin with the construction of the tree T, given a subset $S \subseteq U$ of players and an ordering σ of the players. The tree T will contain a root vertex t_0 that corresponds to t, and will

contain one additional vertex for each player in S. We refer to a non-root node of T and to the corresponding player of S in G interchangeably.

Each vertex $i \neq t_0$ of T will be associated with a *radius* r_i that serves distinct purposes in the tree T and the original graph G. First, the edge from i to its parent in T will have cost r_i. Second, r_i will denote the radius of a ball B_i in the graph G centered at the player i. These balls will be used to determine ancestor-descendant relationships in T.

We initialize the tree T to contain only the root vertex t_0. We give t_0 a radius of $+\infty$, and the ball B_{t_0} of t_0 is defined as the entire player set S. We then add players of S to the tree T one-by-one, in the order prescribed by σ. When adding a new player i, we consider all of the balls of previously added players that contain i. If nothing else, the ball B_{t_0} contains i. Among all such balls, let B_j be one of minimum radius r_j. First, we add the node i to the tree T by making i a child of j. Second, we define the radius r_i as follows. If $j = t_0$, then r_i is half the shortest-path distance between the root t and the player i in the graph G. If $j \neq t_0$, then we define $r_i = r_j/2$. Third, we set the cost of the edge (i, j) in T to be this radius r_i. Finally, we define the ball B_i of player i to be the players of S that lie within distance r_i of i in the graph G. See Figure 5.1 for an instance of this construction.

To begin, we record some simple relations between shortest-path distances in T and in G.

Lemma 5.3.3 *Let i, j be a pair of vertices in T and P_{ij} the unique i-j path in T.*

(a) The cost of P_{ij} is at most four times the cost of its most expensive edge.

(b) The cost of P_{ij} is at least $d_G(i, j)/2$.

Proof Sketch: Edge costs in T decrease by factors of 2 along every root-leaf path. If P_{ij} contains at most one edge incident to t_0, then the sum of the edge costs of P_{ij} is at most twice the cost of its most expensive edge. Otherwise P_{ij} comprises two paths of this type and its cost is at most four times that of its most expensive edge.

Part (b) holds for players i, j that are adjacent in T by the definition of the tree construction. To extend the inequality to a longer path P_{ij}, sum over its constituent edges and use the Triangle Inequality of shortest-path distances in G. ■

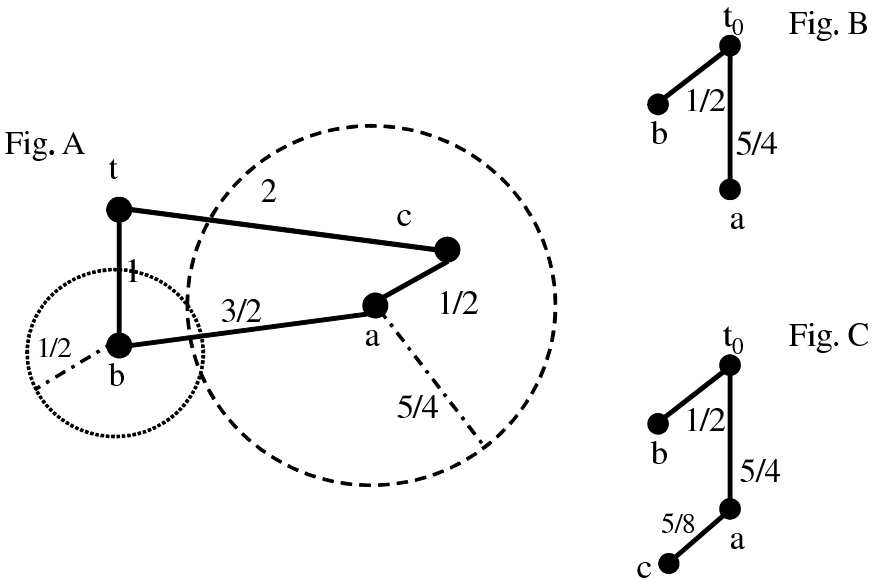


Figure 5.1: Proof of Theorem 5.3.2: the construction of the tree T (Figs. B and C) from the graph G (Fig. A) and ordering $\sigma = a, b, c$ of the players. Fig. B depicts T after players a and b have been considered, and Fig. A shows the balls corresponding to these players.

Now let OPT denote the cost of a minimum-cost Steiner tree in G that spans $S \cup \{t\}$. We next give a series of three lemmas, culminating in a proof that the sum of the costs of the edges of T exceeds OPT by an $O(\log |S|)$ factor. The first lemma states that two edges of the tree T that have roughly equal cost correspond to well-separated players in the graph G; it follows easily from the way we construct T.

Lemma 5.3.4 *Suppose (i_1, j_1) and (i_2, j_2) are edges of T, directed toward the root t_0, with costs c_1 and c_2, respectively. If $c_1 \leq c_2 < 2c_1$, then $d_G(i_1, i_2) \geq c_1$.*

We next show how to use Lemma 5.3.4 to upper bound the number of edges of T with cost in a given range.

Lemma 5.3.5 *For every $\nu \geq 1$, the number of edges of T that have cost in the interval $[OPT/\nu, 2OPT/\nu)$ is at most 2ν.*

Proof: Fix $\nu \geq 1$ and suppose that q edges of T have cost at least OPT/ν and less than $2OPT/\nu$. Lemma 5.3.4 implies that there is a set $A \subseteq S$ of q players that are mutually far apart in G: $d_G(i, i') \geq OPT/\nu$ for every pair i, i' of distinct players of A.

Consider an optimal Steiner tree T^* in G that spans $S \cup \{t\}$ (with cost OPT). Order the players of $A = \{i_1, \ldots, i_q\}$ according to a pre-order traversal of T^* (starting from the root, say). As is well known, we can double every edge of T^* and decompose the resulting multigraph into a collection of paths that connect pairs of adjacent players (including i_1 and i_q). This proves that $\sum_{j=1}^{q} d_G(i_j, i_{j+1}) \leq 2OPT$, where i_{q+1} refers to player i_1. Thus $d_G(i_j, i_{j+1}) \leq 2OPT/q$ for some $j \in \{1, 2, \ldots, q\}$. Since $d_G(i, i') \geq OPT/\nu$ for every $i, i' \in A$, $q \leq 2\nu$. ■

We now combine Lemma 5.3.5 with a grouping argument to upper bound the sum of the edge costs in the tree T.

Lemma 5.3.6 *The sum of the costs of the edges in T is at most $(4 \log_2 |S| + 5) \cdot OPT$.*

Proof: First, note that every edge cost in T is bounded above by the distance $d_G(i, t)$ in G between the root t and some player i of S. Since every such distance is a lower bound on OPT, every edge of T has cost at most OPT.

Next, let $k = |S|$ and consider the edges with cost in the interval $[2^i OPT/k, 2^{i+1}OPT/k)$ for some $i \in \{0, 1, \ldots, \lfloor \log_2 k \rfloor\}$. By Lemma 5.3.5, there are at most $k/2^{i-1}$ edges in this group. The sum of the edge costs in each of the $\lceil \log_2 k \rceil$ groups is therefore at most $4OPT$. Since T has $k+1$ vertices, it has k edges, and thus the total cost of the edges not in any of these groups — each of which has cost less than OPT/k — is at most OPT. Summing over all of the edges proves the lemma. ■

Next, let $\chi^T_{sh}(i_\ell, S_\ell)$ denote the Shapley cost share of the ℓth player (in the given ordering σ) in the fixed-tree multicast instance corresponding to the tree T and the set S_ℓ of the first ℓ players according to σ. Since fixed-tree multicast cost-sharing problems are submodular (Example 2.2.8), inequality (5.2) and Lemma 5.3.6 immediately give the following upper bound on the sum of these Shapley cost shares.

Lemma 5.3.7 *Let i_ℓ denote the ℓth player and S_ℓ the first ℓ players of S according to σ, respectively. Then*

$$\sum_{\ell=1}^{|S|} \chi^T_{sh}(i_\ell, S_\ell) \leq (\ln |S| + 1) \cdot (4 \log_2 |S| + 5) \cdot OPT.$$

Finally, we show that the JV cost share of a player in G is at most a constant factor times its Shapley cost share in T. This is the step of the proof of Theorem 5.3.2 where we use specific properties of the JV cost-sharing method (Proposition 5.3.1).

Lemma 5.3.8 *Let i_ℓ denote the ℓth player and S_ℓ the first ℓ players of S according to σ, respectively. For every $\ell \in \{1, 2, \ldots, |S|\}$,*

$$\chi_{JV}(i_\ell, S_\ell) \leq 8 \cdot \chi^T_{sh}(i_\ell, S_\ell).$$

Proof: Fix $\ell \in \{1, 2, \ldots, |S|\}$ and let $e_1, e_2, \ldots, e_p$ denote the sequence of edges in the i_ℓ-t_0 path in T. Let c_j denote the cost of edge e_j. Let $A_j \subseteq S_\ell$ denote the players of S_ℓ whose path to t_0 in T contains the edge e_j. Let m_j denote the number $|A_j|$ of such players.

Our tree construction ensures that children of i_ℓ correspond only to players subsequent to i_ℓ in the ordering σ, and no such players are in S_ℓ. Thus $A_1 = \{i_\ell\}$, and

of course $A_1 \subseteq \cdots \subseteq A_p \subseteq S_\ell$. First, observe that

$$\chi^T_{sh}(i_\ell, S_\ell) = \sum_{j=1}^{p} \frac{c_j}{m_j}. \tag{5.11}$$

Next, fix $j \in \{2, 3, \ldots, p\}$ and consider a player $i \in A_j$ distinct from i_ℓ. Since the edge e_j separates players i and i_ℓ from t_0 in T, the most expensive edge on the i_ℓ-i path P in T has cost at most c_{j-1}. By Lemma 5.3.3(a), the path P has cost at most $4c_{j-1}$. By Lemma 5.3.3(b), the distance $d_G(i_\ell, i)$ between the players in G is at most $8c_{j-1}$. By Proposition 5.3.1(c), the players i_ℓ and i are in a common connected component by the time $8c_{j-1}$ in the execution of Edmonds's algorithm that defines the JV cost share $\chi_{JV}(i_\ell, S_\ell)$. Crucially, it follows that if player i_ℓ is active at a time subsequent to $8c_{j-1}$ in this execution, then its weakly connected component at this time does not contain the root t and contains at least the m_j (active) players of A_j. Similarly, Lemma 5.3.3 and Proposition 5.3.1(b) imply that player i_ℓ is inactive by the time $8c_p$.

Combining these observations with Proposition 5.3.1(a), we obtain

$$\chi_{JV}(i_\ell, S_\ell) \leq \sum_{j=1}^{p} \int_{8c_{j-1}}^{8c_j} \frac{dt}{m_j} \leq 8 \sum_{j=1}^{p} \frac{c_j}{m_j}, \tag{5.12}$$

where we are interpreting c_0 as 0. Comparing equality (5.11) and inequality (5.12) proves the lemma. ■

Theorem 5.3.2 now follows immediately from Lemma 5.3.7, Lemma 5.3.8, and Theorem 4.2.4.

5.3.3 Every Moulin Mechanism is $\Omega(\log^2 k)$-Approximate

This section proves that the JV mechanism is an *optimal* Moulin mechanism for Steiner tree cost-sharing problems, in the sense that every no-deficit mechanism for such problems is $\Omega(\log^2 k)$-approximate, where k is the size of an efficient outcome. To motivate our proof of this result, we begin with an example showing that our analysis of the JV mechanism is tight up to constant factors.

Example 5.3.9 We construct a Steiner tree instance (Figure 5.2) in rounds by iteratively bisecting an edge of cost 1 as follows. Initially we place the root t at one end of the edge and $\sqrt{n}$ players at the other end of the edge. (Think of n as a large power of 4.) In the second round, we bisect the edge with a new vertex in the middle and add $\sqrt{n}$ further players co-located at this vertex. In round j, we bisect the existing 2^{j-1} edge segments and, for each new node, we add $\sqrt{n}$ new co-located players. The construction concludes when there are n players, after $\Theta(\log n)$ rounds.

Order the players in the same order in which they were added during the construction; break ties among players added in the same round arbitrarily. This defines n successive Steiner tree instances. Consider the cost share of the most recently added player of one of these instances. The JV cost-sharing method satisfies the following property: if a player is co-located with $i-1$ other players (all added earlier) and is distance c away from the nearest non-co-located player that was added in an earlier round, then its cost share in this instance is $\Omega(c/i)$. Because of this, the sum of the cost shares of players added in the jth round of the above construction is $\Omega(\log n)$. Since there are $\Omega(\log n)$ rounds, the sum of all of these successive cost shares is $\Omega(\log^2 n)$. Since the minimum-cost Steiner tree of the full instance has cost 1 and the JV cost-sharing method is positive in this instance, Proposition 4.2.10 implies that the induced Moulin mechanism is $\Omega(\log^2 n)$-approximate.

The main result of this section is a comparable lower bound for *every* $O(1)$-budget-balanced Moulin mechanism.

Theorem 5.3.10 *There is a constant $c > 0$ such that, for every constant $\beta \geq 1$, every β-budget-balanced Moulin mechanism for Steiner tree cost-sharing problems is no better than strongly $(c \log^2 k)$-approximate, where k is the number of players served in an optimal outcome.*

Theorem 5.3.10 implies that Steiner tree cost-sharing problems are fundamentally more difficult for Moulin mechanisms than submodular cost-sharing problems (cf., Section 5.1).

We now outline the proof of Theorem 5.3.10. At the highest level, our goal is to exhibit a (large) network G such that every $O(1)$-budget-balanced Steiner tree Moulin

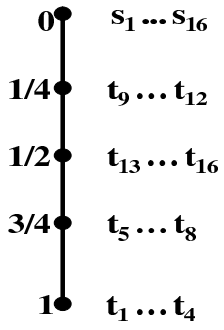


Figure 5.2: The bad example for $n = 16$. One terminal from every terminal pair, the s_i's, are located at one end of the path. Every other node is labeled with the distance from the end with the s_i's and the group of terminals located at that node. Terminals are numbered consistent with the order induced by the construction sequence.

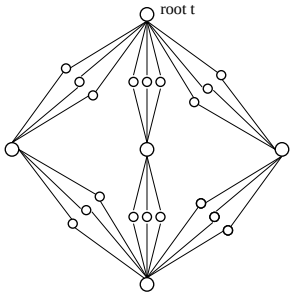


Figure 5.3: Network G_2 in the proof of Theorem 5.3.10, with $m = 3$. All edges have length 1/4.

mechanism behaves like the JV mechanism in Example 5.3.9 on some subnetwork of G.

Fix values for the parameters k and β, where k is a power of 4. Let m be an integer with $m \geq 8\beta\sqrt{k} \cdot (2\beta)^{\sqrt{k}}$. We construct a sequence of networks, culminating in G. The network G_0 consists of a set V_0 of two nodes connected by an edge of cost 1. One of these is the root t. The player set U_0 is $\sqrt{k}$ players that are co-located at the non-root node. For $j > 0$, we obtain the network G_j from G_{j-1} by replacing each edge (u, w) of G_{j-1} with m internally disjoint two-hop paths between u and w. See Figure 5.3. The cost of each of these $2m$ edges is half of the cost of the edge (u, w). Thus every edge in G_j has cost 2^{-j}.

Let V_j denote the vertices of G_j that are not also present in G_{j-1}. We augment the universe by placing $\sqrt{k}$ new co-located players at each vertex of V_j; call each of these groups a *j-group* and denote the union of them by U_j. The final network G is then G_p, where $p = (\log_2 k)/2$. Let $V = V_0 \cup \cdots \cup V_p$ and $U = U_0 \cup \cdots \cup U_p$ denote the corresponding vertex and player sets. Let C denote the corresponding Steiner tree cost function.

A *line* in G_j is a subgraph defined inductively as follows. The only line in G_0 is all of G_0. Each line L_{j-1} of G_{j-1} gives rise to a set of m^{2^j} lines in G_j, each obtained by replacing each edge of L_{j-1} by one of the m two-hop paths to which it corresponds in G_j. Every line in the network G_j has 2^j vertices other than the root, 2^j edges, and unit total cost. In G_p, $\sqrt{k}$ players inhabit each of the $2^p = \sqrt{k}$ non-root vertices on a line.

Now fix an arbitrary cross-monotonic, β-budget-balanced Steiner tree cost-sharing method χ. Our plan is to identify a line of G_p and an ordering of the players on this line such that χ behaves like the JV cost-sharing method in Example 5.3.9. We construct this line iteratively via the following key technical lemma.

Lemma 5.3.11 *Let $S \subseteq U$ be a subset of players that lies on a line in G_p, includes at least one player of U_0, and includes at least one player each from a pair u, w of vertices that are adjacent in G_{j-1}. Let $A_1, \ldots, A_m$ denote the j-groups that correspond to the edge (u, w). Then for some group A_q, its players can be ordered $i_1, \ldots, i_{\sqrt{k}}$ so that*

$$\chi(i_\ell, S \cup \{i_1, \ldots, i_\ell\}) \geq \frac{2^{-j}}{4\ell} \tag{5.13}$$

for every $\ell \in \{1, 2, \ldots, \sqrt{k}\}$.

Before proving Lemma 5.3.11, we use it to prove Theorem 5.3.10 by inductively constructing player sets $S_0, \ldots, S_p$ and orderings $\sigma_0, \ldots, \sigma_p$ with the following properties.

(1) For every $j \in \{0, 1, 2, \ldots, p\}$, S_j corresponds to the $\sqrt{k} \cdot 2^j$ players occupying some line L_j of G_j.

(2) σ_j is an ordering of S_j that orders the $\sqrt{k}$ players of each of its j-groups A

consecutively and in a way that (5.13) holds with S equal to the predecessors of A in σ_j.

For the base case, set $S_0 = U_0$. Since χ is no-deficit, the players of S_0 can be ordered $i_1, \ldots, i_{\sqrt{k}}$ so that $\chi(i_\ell, \{i_1, \ldots, i_\ell\}) \geq C(\{i_1, \ldots, i_\ell\})/\ell = 1/\ell$ for every ℓ. Let σ_0 denote this ordering of S_0.

For the inductive step, let L_{j-1} be the line of G_{j-1} corresponding to S_{j-1}, and consider the edges of L_{j-1} in an arbitrary order. Each such edge gives rise to m j-groups; applying Lemma 5.3.11 with S equal to the players already chosen (in this and previous steps), one of these j-groups can be ordered so that (5.13) holds. Add an arbitrary such group to the player set, ordered after all previously chosen players and so that (5.13) holds. After all of the edges of L_{j-1} have been processed, we obtain a player set S_j and ordering σ_j of them that satisfy the inductive invariants (1) and (2).

Now consider the sum $\sum_{\ell=1}^{k} \chi(i_\ell, S_\ell)$, where i_ℓ and S_ℓ denote the ℓth player and the first ℓ players of S_p with respect to σ_p, respectively. For $j > 0$, the 2^{j-1} j-groups of S_p each contribute at least

$$\sum_{\ell=1}^{\sqrt{k}} \frac{2^{-j}}{4\ell} = \frac{2^{-j}\mathcal{H}_{\sqrt{k}}}{4}$$

to this sum; the 0-group S_0 also contributes at least $\frac{\mathcal{H}_{\sqrt{k}}}{4}$. Thus the sum $\sum_{\ell=1}^{k} \chi(i_\ell, S_\ell)$ is at least

$$\frac{\mathcal{H}_{\sqrt{k}}}{4}\left(1 + \sum_{j=1}^{(\log k)/2} 2^{j-1} \cdot 2^{-j}\right) \geq c \log^2 k = \left(c \log^2 k\right) \cdot C(S)$$

for a constant $c > 0$ that is independent of k. This, combined with Proposition 4.2.12, completes the proof of Theorem 5.3.10.

To conclude, we provide a proof of Lemma 5.3.11.

Proof of Lemma 5.3.11: Let $A_1^1, \ldots, A_m^1$ denote the j-groups corresponding to the edge (u, w) of G_{j-1} and set $X^1 = \cup_{r=1}^{m} A_r^1$. The proof plan is to inductively identify subcollections of these j-groups such that inequality (5.13) holds for an increasing

number of the players in the remaining j-groups. Toward this end, call a set A_r^1 *1-eligible* if

$$\sum_{i \in A_r^1} \chi(i, S \cup X^1) \geq \frac{2^{-j}}{4}. \tag{5.14}$$

Every 1-eligible group contains a player i for which $\chi(i, S \cup X^1) \geq 2^{-j}/4\sqrt{k}$.

Our key claim is that at least $m/2\beta$ groups are 1-eligible. We prove this claim via an averaging argument that relies on the β-budget-balance and cross-monotonicity of χ. Precisely, reindex the 1-eligible groups $A_1^1, \ldots, A_q^1$ and let Y^1 denote their union. An optimal Steiner tree spanning $S \cup Y^1$ consists of a line through S and one group of Y^1, plus $q-1$ "spokes" attaching the rest of the groups to either u or w. Thus $C(S \cup Y^1) = 1 + (q-1)2^{-j}$. Since χ is cross-monotonic and β-budget-balanced, we have

$$\sum_{i \in S \cup Y^1} \chi(i, S \cup X^1) \leq \sum_{i \in S \cup Y^1} \chi(i, S \cup Y^1) \leq \beta(1 + (q-1)2^{-j}).$$

Since (5.14) fails for ineligible groups, and there at most m such groups,

$$\sum_{i \in X^1 \setminus Y^1} \chi(i, S \cup X^1) \leq \frac{m2^{-j}}{4}.$$

On the other hand, since $C(S \cup X^1) = 1 + (m-1)2^{-j}$ and χ is no-deficit,

$$\sum_{i \in S \cup X^1} \chi(i, S \cup X^1) \geq 1 + (m-1)2^{-j}.$$

Combining these three inequalities and rearranging gives the constraint

$$q \geq \frac{3m}{4\beta} - 2^j - \frac{1}{\beta} \geq \frac{m}{2\beta},$$

where the second inequality holds because m is sufficiently large.

Now we iterate the process. In more detail, obtain A_r^2 from each 1-eligible group A_r^1 by removing a player i for which $\chi(i, S \cup X^1) \geq 2^{-j}/4\sqrt{k}$. (Such a player must exist by 1-eligibility.) Let X^2 denote the union of these sets. Such a set A_r^2 is *2-eligible*

if

$$\sum_{i \in A_r^2} \chi(i, S \cup X^2) \geq \frac{2^{-j}}{4}.$$

Every 2-eligible j-group contains a player i for which $\chi(i, S \cup X^2) \geq 2^{-j}/4(\sqrt{k}-1)$. Arguing as above, at least a $1/2\beta$ fraction of the sets A_r^2 are 2-eligible.

Iterating this procedure and reindexing the eligible groups after each iteration, we inductively obtain a collection of disjoint sets $A_1^h, \ldots, A_{q_h}^h$ for each $h \in \{1, 2, \ldots, \sqrt{k}\}$ with the following properties:

(1) $q_h \geq m/(2\beta)^h$;

(2) for each $r \in \{1, \ldots, q_h\}$, A_r^h contains a player i_r^h such that $\chi(i_r^h, S \cup X_h) \geq 2^{-j}/4(\sqrt{k}-h+1)$, where $X_h = \cup_r A_r^h$;

(3) for each $r \in \{1, \ldots, q_h\}$ and $h > 1$, $A_r^h = A_r^{h-1} \setminus \{i_r^{h-1}\}$.

Since m is sufficiently large, $q_{\sqrt{k}} \geq 1$. By properties (2) and (3) and cross-monotonicity of χ, the group A_1^1 that corresponds to $A_1^{\sqrt{k}}$ satisfies the requirements of the lemma. ■

5.4 Steiner Forest Cost-sharing Problems

In this section we identify an optimal Moulin mechanism for the Steiner forest cost-sharing problem defined in Example 2.2.9. We seek an no-deficit Moulin mechanism for this problem with the best-possible approximate efficiency. Section 5.4.1 describes the previously proposed Könemann, Leonardi and Schäfer mechanism [55]. Section 5.4.2 bounds the efficiency of this mechanism and proves that it is optimal.

5.4.1 The KLS Cost-Sharing Method

We describe the KLS cost-sharing method, due to Könemann, Leonardi and Schäfer [55]; it is a cross-monotonic, 2-budget-balanced cost-sharing method for every Steiner forest cost function, and no cross-monotonic method can have better budget-balance [45, 55]. We call the induced Moulin mechanism the KLS mechanism.

The KLS cost-sharing method modifies the primal-dual Steiner forest algorithms of Agrawal, Klein and Ravi [3] and Goemans and Williamson [35] in a novel way to ensure cross-monotonicity. The method takes as input a graph $G = (V, E)$ with edge costs and a set of players S, where each player $i \in S$ is identified with a source-sink pair (s_i, t_i). It outputs a feasible Steiner forest (a subgraph containing an s_i-t_i path for all $i \in S$) and a cost share for each player.

The Steiner forest and cost shares are defined in tandem via the following primal-dual algorithm, which we describe as a process over time. Primal variables correspond to edges and dual variables correspond to subsets of nodes in the graph. The algorithm maintains an acyclic set F of edges (initially empty) and a set of feasible dual variables $\{y_A\}$ (initially zero). By *feasible*, we mean that for every edge $e \in E$, the sum of the dual variables $\sum_{A \subseteq S\,:\, e \in \delta(A)} y_A$ corresponding to sets A that contain exactly one endpoint of e is at most the edge cost c_e; this is the dual packing constraint corresponding to the edge e.

The algorithm proceeds by uniformly raising the values of *active* dual variables, which correspond to a subset of the connected components of the current subgraph (V, F). We define the activity criterion soon. These variables are increased until an edge e becomes *tight*, in the sense that the dual packing inequality corresponding to e holds with equality. At this point, the edge e is added to the set F and the set of active dual variables is updated. The algorithm continues in this way until no active dual variables remain.

A key definition in [55], which determines the active dual variables, is that of the *death time* of a terminal. The death time of both s_i and t_i is defined as half of the length of a shortest s_i-t_i path. The active dual variables at time τ with respect to the subgraph (V, F) are then defined as the variables y_A such that A is a connected component of (V, F) containing at least one terminal with a death time larger than τ. Intuitively, the death times of s_i and t_i are defined as the smallest value that ensures s_i-t_i connectivity in the final output of the above primal-dual algorithm, while conservatively ignoring the contributions of other source-sink pairs.

Cost shares for the players of S are defined using the dual variables as follows. At each instant, the increase in an active dual variable is split equally among the active

terminals in the corresponding connected component. The cost share of a terminal is then defined as the total cost that it accumulates over time. The KLS cost shares of a player, as described by [55], is simply the sum of the cost-shares of the corresponding terminals. We muliply these cost-shares by a factor of 4 to ensure that the cost of the constructed solution is recovered.

Results from [55] immediately show that these cost shares are cross-monotonic; with the scaling, their sum is at least the cost of the constructed Steiner forest and is at most twice the cost of an optimal Steiner forest. Further, by weak duality, the cost of the constructed Steiner forest is at most twice that of an optimal solution.

5.4.2 The KLS Mechanism Is $O(\log^2 k)$-Approximate

We now analyze the efficiency of the KLS mechanism. The main result of this section is the following.

Theorem 5.4.1 *The KLS mechanism is $O(\log^2 k)$-approximate for every Steiner forest cost-sharing problem, where k is the size of an efficient outcome.*

Recall from Section 5.3.3 that every $O(1)$-budget-balanced Moulin mechanism is $\Omega(\log^2 k)$-approximate, and that Steiner tree cost functions are instances of Steiner forest cost functions (one terminal from every terminal pair is located at a fixed node called the *root*). The KLS mechanism is thus an optimal $O(1)$-budget-balanced Moulin mechanism, up to constant factors.

Overview of the Proof of Theorem 5.4.1

The proof of the theorem is technical and so we start by providing an overview. By Theorem 4.2.4 it suffices to show that the KLS cost-sharing method is is $O(\log^2 k)$-summable. By the definition of summability (Definition 4.2.2), we need to analyze the following procedure. Given an arbitrary Steiner forest instance and an arbitrary ordering of the players (source-sink pairs), we add the players to the instance one-by-one, according to the given ordering. Each time we add a new player, we compute its KLS cost share in the instance induced by the set of players added thus far. The

key question is: by how much can the sum of these successive cost shares exceed the cost of servicing all of the players?

The problem has an online flavor as we add players one-by-one. However, rather than bounding the cost of the online solution as is typical in online analysis, we are interested in the sum of cost shares.

Our analysis proceeds in two steps. The first step is motivated by the difficulty in directly bounding the above successive cost shares in a general network. The idea of this step is to replace the given network by a forest with cost at most an $O(\log k)$ times that of an optimal Steiner forest. The construction resembles that of an online Steiner tree using the standard greedy strategy [44]. However, to facilitate our charging argument in the second step, we require that each tree of this forest be an ultrametric—i.e., all root-leaf paths have equal length.

The goal of the first step is also reminiscent of probabilistic tree embeddings (see e.g. [9, 30]). However we cannot apply such an embedding as a black box. The reason is that our charging argument in the second step requires structure beyond the low distortion guarantee—it also needs the distances in the ultrametric to be tightly coupled with the dual growth process used to define the KLS cost-shares.

In the second step, we demonstrate how to charge the k successive KLS cost shares to the ultrametrics constructed in the first step. Loosely speaking, we show how subtrees in each ultrametric correspond to active components during the execution of the primal-dual algorithm that defines the KLS cost shares. Our charging scheme charges each point of each ultrametric $O(\log k)$ times, proving an $O(\log^2 k)$ bound on the summability of the KLS cost-sharing method.

While portions of this argument are similar to that used in Section 5.3.2 to upper bound the summability of the Jain-Vazirani (JV) Steiner tree cost-sharing method, the refined ultrametric structure and the charging argument in this section are new. One reason we require the ultrametric structure is that the primal-dual algorithm underlying the KLS mechanism determines cost shares using fixed "death times", rather than via the component structure in the dual growth process as in JV, where a player becomes inactive once it belongs to the root's component. While crucial for cross-monotonicity of the KLS cost-sharing method, this property can cause a terminal to

accumulate a cost share beyond the point at which it is connected to its mate, and it is not obvious how to bound this additional accumulation. The example below exhibits a Steiner tree problem instance for which the summability of the KLS method is an $\Omega(\log n)$ factor times larger than that of the Jain-Vazirani method. Nonetheless, we prove in this section that the KLS method is always $O(\log^2 k)$-summable, matching the (tight) worst-case bound for the Jain-Vazirani method *Steiner tree cost functions.*

Example 5.4.2 Our instance consists of $n+1$ points on the unit line. The root lies at point 0. The ith point in the ordering lies at distance i/n from 0. Let σ be the identity permutation. Recall that in the JV primal-dual algorithm, a player becomes inactive once it belongs to the root's component. For every successive instance, this happens at time $O(1/n)$ for every player in the instance. Then, the ith successive JV cost share in this instance is $O(1/n)$; The JV cost-sharing method has summability $O(1)$ in this example.

On the other hand, consider the KLS cost share of the ith terminal with death time $i/2n$. Until time $1/2n$, the cost share increases at a rate of 1. At time $1/2n$, all the i nodes preceding and including the ith one, form a single component. Thereafter, for every $j < i$, from time $j/2n$ to $(j+1)/2n$, there are $i-j$ active nodes in i's component (all these nodes have death times of at least $(j+1)/2n$, and therefore, its cost share goes up at a rate of $1/(i-j)$. The net cost share of the ith terminal is $\Theta((\log i)/n)$. The KLS cost-sharing method therefore has summability $\Theta(\log n)$ in this example.

Building the Forest

We now describe the first step of our analysis of Theorem 5.4.1. We define a procedure that takes as input a Steiner forest instance $G = (V, E)$ with edge costs, an (adversarial) ordering σ of the source-sink pairs $(s_1, t_1), \ldots, (s_k, t_k)$ and constructs a forest F, defined on the terminals, that has cost $O(\log k)$ times that of a minimum-cost Steiner forest, as well as other structure useful in the second step of the analysis. While the following description will be algorithmic, we emphasize that this construction is purely for the purposes of analyzing the summability of the KLS cost-sharing

method.

Consider an optimal solution to the given Steiner forest instance. Our forest F will have one tree for each connected component of this optimal solution. We will construct these trees independently of each other, so we can restrict our description to a single component T^* of the optimal Steiner forest. Let A^* denote the terminals spanned by T^*. The vertex set of the tree T that we construct will contain all the terminals in A^* as well as some auxiliary vertices.

We now describe the construction of T. Figure 5.4 depicts the construction for a simple graph with three terminal pairs. The ordering $\sigma = (s_1, t_1), \ldots, (s_k, t_k)$ on source-sink pairs induces an ordering $s_1, t_1, s_2, t_2, \ldots, s_k, t_k$ on the terminals and also an ordering of A^*. We construct T by adding terminals in A^* to it in this order. When a terminal is considered, we attach it to the existing tree and endow it with a *radius*. The *ball* of a terminal x with radius r is defined as the terminals of A^* at distance at most r from x in the given graph G. To start the construction, we introduce an auxiliary root x_0 and create an edge e_0 between x_0 and x_1, the first terminal, of length D_{max}, where D_{max} is half the largest distance (according to the graph G) between any two terminals of A^*. We call this edge e_0 the *backbone edge*. We endow the terminal x_1 with a ball of infinite radius.

Now consider some subsequent terminal x. Among all of the previously added terminals whose ball contains x, we define the terminal y with the minimum radius to be the *parent* of x and write $p(x) = y$. If y has finite radius—i.e., is not the first terminal of A^* with respect to σ—then we define x's radius r_x to be half of its parent's radius. Otherwise, we define the radius r_x to be half of the shortest-path distance between x and y in G. To attach x to the tree T, consider the path from y to x_0 in T. We connect x to the point along this path at a distance r_x from y, possibly creating a new internal node. The backbone edge and the definition of D_{max} ensure that this is always possible. Call this point $v(x)$. The length of the edge between $v(x)$ and x is defined to be r_x. Note that the leaves of T are in bijective correspondence with the terminals A^* allowing us to refer to a leaf of the tree by its corresponding terminal. See Figure 5.4 for an instance of the tree construction.

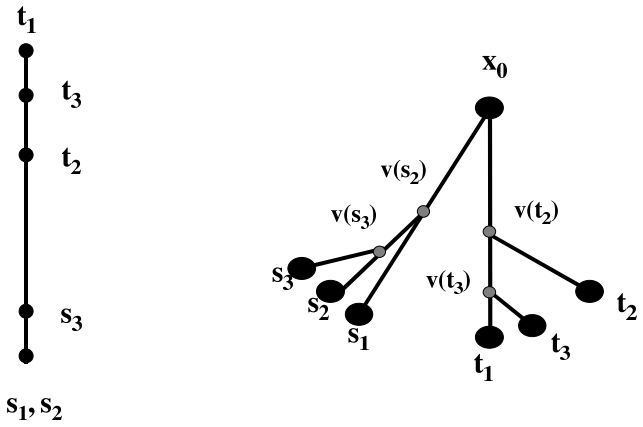


Figure 5.4: An instance of the tree construction with three terminal-pairs $(s_1, t_1), (s_2, t_2), (s_3, t_3)$. The construction ordering σ is $(s_1, t_1, s_2, t_2, s_3, t_3)$. Figure (a) shows the graph G which is also the tree T^*. Figure (b) shows the ultrametric tree T; the edge (x_0, t_1) is the backbone edge.

Some useful properties of the tree T:

We establish properties of the tree T useful in the charging argument of the second step of the proof. We first show that the tree T is an ultrametric.

Lemma 5.4.3 *The tree T is an ultrametric, with all root-leaf paths having length D_{max}.*

Proof: The lemma statement is an invariant of the construction process. It is trivially true after the first step of the tree construction as the tree consists of a single edge, the backbone edge with length D_{max}. When a new terminal x is added to the tree T, the distance from the interior node $v(x)$ to x equals the distance from $v(x)$ to the parent $p(x)$ of x, maintaining the invariant. ■

Next, we relate distances between pairs of leaves of the tree T to distances between the corresponsing terminals in the graph. For every two terminals x, y in A^*, let $d_T(x, y)$ and $d_G(x, y)$ denote the distances between x and y in the tree T and in the graph G, respectively. The next lemma follows immediately from the construction.

Lemma 5.4.4 *For every terminal $x \in A^*$ with parent $p(x)$, $d_T(x, p(x)) \geq d_G(x, p(x))$.*

Proof: By construction, $d_T(x, p(x)) = 2r_x = r_{p(x)}$, which is at least $d_G(x, p(x))$ as x is in $p(x)$'s ball. ■

We now approximately extend Lemma 5.4.4 to every pair of terminals $x, y \in A^*$.

Lemma 5.4.5 *For every pair $x, y \in A^*$ of terminals in T, $d_T(x, y) \geq d_G(x, y)/5$.*

The idea of the proof is to consider a walk W_{xy} between x and y in T and relate the length of this walk, ℓ_{xy} to both $d_T(x, y)$ and $d_G(x, y)$. Precisely, fix $x, y \in A^*$ and consider the (unique) path P_{xy} between x and y in the tree T. Label each internal node $v(x)$ by the parent of x. The length of this path is $d_T(x, y)$. To construct the walk W_{xy}, consider the sequence S_{xy} of vertices that the path P_{xy} visits; apart from x and y, all of these are internal nodes of T. Obtain a sequence S'_{xy} of terminals from S_{xy} by replacing the internal nodes of S_{xy} by their label values (terminals) and then removing duplicates. Obtain the walk W_{xy} in T by visiting the terminal nodes in S'_{xy} in order, along the unique paths in T that connect consecutive nodes. The walk W_{xy} contains P_{xy} as a subgraph, and can be decomposed into P_{xy} and a set of circuits, each of which starts and ends at an internal node of P_{xy}, visiting the terminal node corresponding to the label of the internal node along the way. Figure 5.5 shows the walk W_{xy} and the path P_{xy} for $x = s_2$ and $y = t_2$ for the tree T from Figure 5.4. The proof of Lemma 5.4.5 is an easy consequence of following two lemmas.

Lemma 5.4.6 *For every pair $x, y \in A^*$ of terminals in T, $\ell_{x,y} \geq d_G(x, y)$.*

Proof: Every pair of consecutive nodes in the sequence S'_{xy} share a parent-child relationship. Applying Lemma 5.4.4 and the triangle inequality completes the proof. ■

Lemma 5.4.7 *For every pair $x, y \in A^*$ of terminals in T, $d_T(x, y) \geq \ell_{x,y}/5$.*

Proof: The proof is by a charging argument. Recall that the walk W_{xy} contains P_{xy} as a subgraph, and can be decomposed into P_{xy} and a set of circuits, each of which starts and ends at an internal node of P_{xy}, visiting the terminal node which labels the internal node along the way. (See Figure 5.5.)

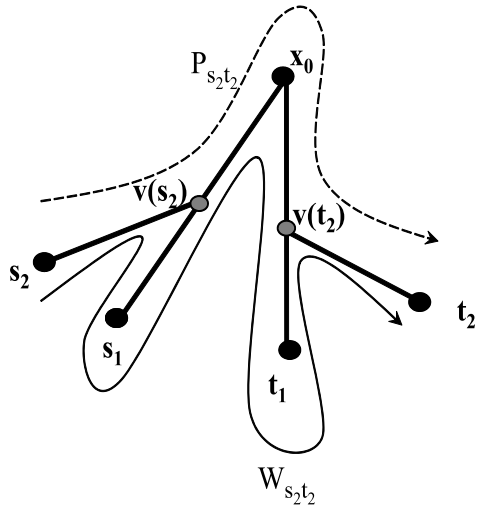


Figure 5.5: The walk W_{xy} and the path P_{xy} for $x = s_2$ and $y = t_2$ for the tree T from Figure 5.4. The terminals s_3, t_3 from Figure 5.4 are not shown.

We will charge the length of the circuits to the segments of P_{xy} that precede them in the walk. More precisely we will show that the circuit has a length at most four times that of the segment of P_{xy} we charge it to, and hence the length of the walk W_{xy} is at most five times the length of path P_{xy}, proving the lemma.

Consider a circuit of W_{xy} rooted at the internal point $v(z)$ and visiting the terminal $p(z)$. The length of the circuit is at most $2r_z$, while the length of the segment of P_{xy} immediately preceding $v(z)$ is at least $r_z/2$ (r_z minus the radius of any of $z's$ children). Thus, the circuit is at most four times the length of the segment of P_{xy} we charge it to. ■

Bounding the size of the tree T:

In this section we establish the following bound the size of the tree T in terms of the optimal tree T^*.

Lemma 5.4.8 *The sum of the costs of the edges in T is $O(\log k) \cdot c(T^*)$.*

We mention two lemmas that culminate in the proof. Both lemmas have direct analogous to ones used to bound the summability of the JV cost-sharing method. The first lemma states that terminals that have edges of a comparable length in the tree T must be well separated in the graph G. The proof follows that of Lemma 5.3.4.

Lemma 5.4.9 *Consider two terminals x_1 and x_2 with edges x_1, y_1 and x_2, y_2 of length c_1 and c_2 in the tree T such that $c_1 \leq c_2$ and $c_2/c_1 < 2$. Then $d_G(x_1, x_2) \geq c_1$.*

The second lemma uses the above lemma to prove an upper bound on the number of edges of the tree T that can have a comparable length. The proof is identical to that of Lemma 5.3.5.

Lemma 5.4.10 *For any $\gamma \geq 1$, the number of edges of T that have cost in the range $(c(T^*)/\gamma,\ 2c(T^*)/\gamma]$ is at most 2γ.*

We now complete the proof of Lemma 5.4.8.

Proof of Lemma 5.4.8: The proof follows by sorting the edges of T into bins and bounding the sum of the edge lengths in every bin. Suppose that A^* has m players of S. First, we claim that every edge cost in T is at most $c(T^*)$. Our construction ensures that every edge cost is at most D_{max}. On the other hand, as T^* spans all of the terminals of A^*, its cost is at least D_{max}.

Now consider edges with cost in the interval $(2^i c(T^*)/m, 2^{i+1}c(T^*)/m]$ for some $i \in \{0, 1, \ldots, \lfloor \log_2 m \rfloor\}$. By Lemma 5.4.10, there are at most $m/2^{i-1}$ edges in this group. The sum of the edge costs in each of the $O(\log m)$ possible groups is therefore at most $4c(T^*)$. Finally, since T has at most $2m$ edges, the total contribution of edges with cost at most $c(T^*)/m$ is at most $2c(T^*)$. Summing over all groups of edges and noting that $m \leq k$ proves the lemma. ■

The Charging Argument

We are now ready to bound the summability of the KLS cost-sharing method. Our charging argument will proceed independently for each ultrametric constructed in Section 5.4.2; we will fix one such ultrametric T, spanning a set A^* of terminals.

For technical reasons, we henceforth use a version of the tree T scaled up by a factor 10. Lemmas 5.4.3 and 5.4.8 continue to hold while Lemma 5.4.5 can be restated as follows:

Lemma 5.4.11 *For every pair $x, y \in A^*$ in T, $d_G(x, y) \leq \frac{1}{2} d_T(x, y)$.*

We now commence the charging argument. Let x_ℓ and A_ℓ denote the ℓth terminal and the first ℓ terminals of A^*, respectively, with respect to the ordering induced by σ. We aim to charge the KLS cost share $\chi^{KLS}(x_\ell, A_\ell)$ of a terminal $x_\ell \in A^*$ to points of the tree T. (A technical detail: since matched pairs of terminals appear consecutively in the ordering induced by σ, the set A_ℓ contains only matched pairs of terminals, plus possibly an orphaned source s_i. In either case, $\chi^{KLS}(x_\ell, A_\ell)$ denotes the KLS cost share assigned to the terminal x_ℓ in the Steiner forest instance induced by all of the players with at least one terminal in the set A_ℓ.)

The charging proceeds as follows. Let P_ℓ be the unique path in T from x_ℓ to x_0, and consider the primal-dual algorithm that determines the KLS cost share $\chi^{KLS}(x_\ell, A_\ell)$. At each moment in time τ up to the death time of x_ℓ, the terminal's cost share increases at a positive rate, equal to the inverse of the number of active terminals in x_ℓ's component at time τ. For each such time τ, we charge this (marginal) increment in x_ℓ's cost-share to the point $g_\ell(\tau)$ which is at distance τ from x_ℓ along the path P_ℓ.

Since every leaf-root path of T has length at least D_{max} (Lemma 5.4.11)—half of the largest distance between two terminals of A^*—and since D_{max} is at least the death time of every terminal of A^*, this procedure fully charges the sum $\sum_\ell \chi^{KLS}(x_\ell, A_\ell)$ of the KLS cost shares to T.

Fix a point g of T. Only terminals in the subtree of T rooted at g charge part of their cost share to g. By the ultrametric property (Lemma 5.4.3), all of these terminals are equidistant from the point g in T; let this common distance be τ_g. Such a terminal charges part of its cost share to g if and only if its death time is at least τ_g; let B denote these terminals. We first show that at time τ_g the terminals of $B \cap A_\ell$ are in the same component and hence must share the increase in the active dual variable corresponding to this compoenent; we then use this to show that for every point g of the tree T, the sum of the (marginal) charges to g by the terminals of A^* is only

$O(\log k)$.

Lemma 5.4.12 *Suppose $x_\ell \in B$. Then at time τ_g in the run of the primal-dual algorithm that defines the KLS cost share $\chi^{KLS}(x_\ell, A_\ell)$, all the terminals of $A_\ell \cap B$ are in the same component as x_ℓ.*

Proof: Fix $x \in (A_l \cap B) \setminus \{x_\ell\}$. As x and x_ℓ lie at distance τ_g from the point g in T, $d_T(x, x_\ell) \leq 2\tau_g$. Lemma 5.4.5 then implies that $d_G(x, x_\ell) \leq \tau_g$. Now, as $x, x_\ell \in B$, the death times of x_ℓ and x are both at least τ_g, and in particular the terminal x is active at time τ_g. As $d_G(x, x_\ell) \leq \tau_g$, even ignoring the contributions of other terminals, the components containing x_ℓ and x must merge by time τ_g in the run of the primal-dual algorithm that defines the KLS cost share $\chi^{KLS}(x_\ell, A_\ell)$. ■

Lemma 5.4.13 *For every point g of T, the total marginal charge to g is at most $2 \cdot \mathcal{H}_{|B|}$, where $\mathcal{H}_j = \sum_{i \leq j} 1/i$ denotes the jth Harmonic number.*

Proof: Since the KLS cost-sharing method splits the increase in value of an active dual variable equally among the active terminals contained in the corresponding component, Lemma 5.4.12 implies that the marginal charge to the point g by the terminal $x_\ell \in B$ is at most $2/|B \cap A_\ell|$, which is at most $2/|A_\ell|$. Summing over the contributions of the terminals in B, the total amount charged to this point is at most $2 \cdot \sum_{1 \leq l \leq |A^*|} 1/l$, which is $2 \cdot \mathcal{H}_{A*}$, proving the lemma. ■

Theorem 5.4.1 now follows easily from Lemmas 5.4.8 and 5.4.13 and summing over all of the components of the optimal solution.

5.5 SSRoB Cost-Sharing Problems

In this section we identify an optimal Moulin mechanism for the SSRoB cost-sharing problem defined in Example 2.2.10. We seek an $O(1)$-budget-balanced Moulin mechanism for this problem with the best-possible approximate efficiency. Section 5.5.1 describes a mechanism, independently proposed by Gupta Srinivasan and Tardos [39] and Leonardi and Schäfer [56]. Section 5.5.2 bounds the efficiency of this mechanism and argues that the mechanism is optimal.

5.5.1 The GST cost-sharing method

Gupta, Srinivasan, and Tardos [39] and Leonardi and Schäfer [56] independently designed the following $O(1)$-budget-balanced cross-monotonic cost-sharing method for SSRoB, which we call the *GST method*. Given an SSRoB cost function and a set $S \subseteq U$ of players, we use the randomized algorithm of [40] to produce a feasible solution. This algorithm first chooses a random subset $D \subseteq S$ by adding each player $i \in S$ to D independently with probability $1/M$. Second, it computes an approximate Steiner tree spanning $D \cup \{t\}$ using, for example, the 2-approximate MST heuristic [73], and buys infinite capacity on all of the edges of this tree. Third, for each player $i \notin D$, it rents one unit of capacity for exclusive use by i on a shortest path from its vertex to the closest vertex in $D \cup \{t\}$. This defines a feasible solution with probability 1, and the expected cost of this solution is at most 4 times that of an optimal solution to the SSRoB instance induced by S [40].

We now discuss the cost shares. The GST cost-sharing method uses JV cost shares (Section 5.3) for the Steiner tree problem as a building block. These cost shares are cross-monotonic and 2-budget-balanced for any instance of the Steiner tree cost-sharing problem.

The GST cost share $\chi_{GST}(i, S)$ is defined as the sum of two terms $\chi_{buy}(i, S)$ and $\chi_{rent}(i, S)$ which represent the cost shares of a player $i \in S$ with respect to the bought and rented edges respectively. $\chi_{buy}(i, S)$ and $\chi_{rent}(i, S)$ are the expected values of two random variables $B(i, S)$ and $R(i, S)$ defined over the the random choice of the set D in the above algorithm. $R(i, S)$ equals the length of the shortest path used to connect i to a vertex in $D \cup \{t\}$ if $i \notin D$ and is 0 otherwise. $B(i, S)$ equals M times the Jain-Vazirani cost share $\chi_{JV}(i, D)$ of i with respect to the Steiner tree instance defined by G, c, t, and the players D if $i \in D$, and is 0 otherwise. For fixed coin tosses of the above randomized algorithm, for any player i, note that at most one of $B(i, S)$ or $R(i, S)$ is non-zero.

For fixed coin tosses, the cost-sharing method induced by the set of random variables $B(i, S)$ for all $S \subseteq U$ and $i \in S$ is the JV cost-sharing method on an instance of the Steiner tree problem and is therefore cross-monotonic, while the cost-sharing method induced by the set of random variables $R(i, S)$ for all $S \subseteq U$ and $i \in S$ are

cross-monotonic because renting costs fall as the set S and hence the set D expands. GST cost-shares are thus cross-monotone as they are the weighted (by probabilities induced by the random choice of set D) sum of cross-monotone cost-sharing methods. Results from [39, 56] imply that the GST cost-shares are 4-budget-balanced. We call the Moulin mechanism induced by the GST cost-sharing method the GST mechanism.

5.5.2 The GST Mechanism is $O(\log^2 k)$-approximate

The main result of ths section bounds the efficiency loss of the GST mechanism.

Theorem 5.5.1 *The GST mechanism is $O(\log^2 k)$-approximate for every SSRoB cost-sharing problem, where k is the number of players in an efficient solution.*

The SSRoB cost-sharing problem reduces to the Steiner tree cost-sharing problem when $M = 1$, and Section 5.3.3 shows that every $O(1)$-budget-balanced Moulin mechanism for Steiner tree cost-sharing problem is $\Omega(\log^2 k)$-approximate. This matches the upper bound from Theorem 5.5.1 and argues this GST mechanism is an optimal $O(1)$-budget-balanced Moulin mechanism. We now discuss the proof of Theorem 5.5.1.

By Theorem 4.2.4, it suffices to show that GST cost shares are $O(\log^2 k)$-summable. Mirroring several recent analyses of sampling algorithms for rent-or-buy problems [40, 38], our analysis proceeds in two steps. First we show that the summability of χ_{rent} is at most a constant factor times that of χ_{buy}. Second, we directly upper bound the summability of χ_{buy}. The first step will follow from a property of JV cost shares called *strictness*.

Lemma 5.5.2 (strictness) *For every Steiner tree instance, for every set $S \subseteq U$ and player $i \in S$, the JV cost share of a player is at least half the distance between i and the nearest node in the set $S \setminus \{i\} \cup \{t\}$.*

We now use strictness to bound χ_{rent} in terms of χ_{buy}.

Lemma 5.5.3 *Let C be a SSRoB cost function. For every $S \subseteq U$, $i \in S$, $\chi_{rent}(i,S) \leq 2 \cdot \chi_{buy}(i,S)$.*

Proof: Fix a set $S \subseteq U$ and $i \in S$ arbitrarily. Recall the description of GST cost shares from Section 5.5.1. Condition on the membership of all the players of $S \setminus \{i\}$ in the set D. Let $D \cup (S \setminus \{i\}) = D'$. Define $R(i,S)$ and $B(i,S)$ as in the definition of GST cost shares.

Player i will be included in the set D with probability $1/M$, in which case its conditional cost share will be $2M \cdot \chi_{JV}(i, D' \cup \{i\})$, where $\chi_{JV}(i, D' \cup \{i\})$ is player i's JV cost share in the Steiner tree cost-sharing problem induced by the players of $D' \cup \{i\}$. Player i is excluded from the random sample D with probability $(1-1/M)$, in which case its conditional cost share equals the distance between i and the nearest player in the set $D' \cup \{t\}$.

By Lemma 5.5.2, $E[R(i,S)|D'] \leq 2 \cdot (1-1/M) E[B(i,S)|D']$. Taking expectations to remove the conditioning on D' proves the lemma. ■

We now show that for every SSRoB cost function, the corresponding cost-sharing method χ_{buy} is $O(\log^2 k)$-summable. This will complete the proof of Theorem 5.5.1.

Lemma 5.5.4 *Let C be a SSRoB cost function and χ_{buy} the first term of the corresponding GST cost-sharing method. Then χ_{buy} is $O(\log^2 k)$-summable for C.*

Proof: Condition on the random choice of the set $D \subseteq S$. Let OPT_D denote the cost of the optimal steiner tree for the terminals $D \cup \{t\}$. A result of Gupta, Kumar, and Roughgarden [40], based on earlier work by Karger and Minkoff [50], implies that M times the expectation (over D) of OPT_D is at most the cost $C(S)$ of an optimal SSRoB solution.

As JV cost shares are $O(\log^2 k)$-summable for all Steiner tree cost functions (Section 5.3)

$$\sum_{\ell=1}^{|S|} \chi_{JV}(i_\ell, S_\ell) \leq O(\log^2 k) \cdot OPT_D \leq O(\log^2 k) \cdot C(S) \tag{5.15}$$

for every ordering σ of D, where S_ℓ and i_ℓ denote the set of the first ℓ players of D and the ℓth player of S (with respect to σ), respectively. Taking expectations to remove

the conditioning on D and recalling the definition of χ_{buy} completes the proof. ■

Remark 5.5.5 MRoB cost-sharing problems (Example 2.2.11) also admit $O(1)$-budget-balanced, $O(\log^2 k)$-approximate Moulin mechanisms. See Roughgarden and Sundararajan [71] for details.

Chapter 6

Acyclic Mechanisms

Having identified optimal Moulin mechanisms for various cost-sharing problems, we now propose an alternative framework called *acyclic* mechanisms. Like Moulin mechanisms, acyclic mechanisms are ascending auctions based on cost-sharing methods. However, acyclic mechanisms can employ a wider class of cost-sharing methods—the cost-sharing methods need not be cross-monotonic. Section 6.1 discusses how acyclic mechanisms get around the lack of cross-monotonicity. Section 6.2 defines acyclic mechanisms formally. Section 6.3 establishes the truthfulness of acyclic mechanisms.

The next chapter identifies the two benefits of acyclic mechanisms. First, it is easier to design acyclic mechanisms than Moulin mechanisms: many classical combinatorial algorithms (based on the primal-dual method) naturally induce a non-Moulin acyclic mechanism with good performance guarantees. Second, for important classes of cost-sharing problems, there exist acyclic mechanisms that have exponentially better economic efficiency compared to the best Moulin mechanisms. Section 6.4 summarizes these efficiency improvements.

6.1 Circumventing Non-crossmonotonicity

Recall from Chapter 4 that a Moulin mechanism can be viewed as a simulation of an iterative ascending auction, with the prices that are simultaneously offered to the remaining players at each iteration governed by the underlying cost-sharing method.

Cross-monotonicity of the cost-sharing method ensures that the sequence of prices offered to a player is nondecreasing, which in turn implies that the mechanism is truthful. Conversely, non-cross-monotonic cost-sharing methods result in iterative auctions that need not be ascending, and the corresponding mechanisms are generally not truthful.

In an acyclic mechanism, in each iteration of the simulated iterative auction, prices are offered to the remaining players according to a designer-specified *order*. If each remaining player accepts the price offered to it, then the mechanism halts, and the remaining players are served at the prices offered in the final iteration. If some player refuses to pay the price it is offered, then the iteration terminates immediately, this player is removed for the rest of the auction, and the next iteration begins with the remaining players. Thus, a player need not be offered a price in every iteration.

Ordering the offers to the remaining players permits the construction of truthful mechanisms from non-cross-monotonic cost-sharing methods. Intuitively, the early termination of an iteration conceals subsequent prices from the players. If aborted iterations correlate appropriately with failures of cross-monotonicity, then the simulated iterative auction is ascending in the following sense: whenever an offer is made to a player, it is at least as large as every offer made in previous iterations. This property is sufficient for truthfulness. As we will see, many primal-dual algorithms naturally induce a cost-sharing method that is not cross-monotonic but possesses precisely this type of correlation.

6.2 Definitions

To define an acyclic mechanism for a cost function C and a universe U, we require both a cost-sharing method χ and an *offer function* τ. An offer function specifies a nonnegative *offer time* $\tau(i, S)$ for every subset $S \subseteq U$ and every player $i \in S$. These times specify the ordering in which the players of S should be offered a price, with lower times corresponding to earlier offers, and equal times indicating simultaneous offers. A cost-sharing method and an offer function induce a mechanism that simulates an iterative auction in a natural, generic way:

Definition 6.2.1 Let U be a universe of players, χ a cost-sharing method defined on U, and τ an offer function defined on U. The *mechanism* $M(\chi, \tau)$ *induced by* χ *and* τ is the following.

1. Collect a bid b_i from each player $i \in U$.
2. Initialize $S := U$.
3. If $b_i \geq \chi(i, S)$ for every $i \in S$, then halt. Output the set S, the feasible solution constructed by χ, and charge each player $i \in S$ the price $p_i = \chi(i, S)$.
4. Among all players $i \in S$ with $b_i < \chi(i, S)$, let i^* be one with minimum $\tau(i, S)$. (Break ties arbitrarily.)
5. Set $S := S \setminus \{i^*\}$ and return to Step 3.

Remark 6.2.2 The definition of the mechanism $M(\chi, \tau)$ depends only on the ordering of the offer times, and not on their numerical values. We work with real-valued offer times rather than abstract orderings because such times arise naturally in primal-dual algorithms.

Remark 6.2.3 For every universe U and cost-sharing method χ, the Moulin mechanism induced by χ is equivalent to the mechanism induced by χ and the identically zero offer function.

As foreshadowed in the previous section, the mechanism induced by a cost-sharing method and an offer function will be truthful only if all failures of cross-monotonicity are suppressed by the offer function. We formalize the required property next; we prove that it is sufficient for truthfulness in Section 6.3.

Let τ be an offer function defined on a universe U. For a subset $S \subseteq U$ and a player $i \in S$, let $L(i, S)$, $E(i, S)$, and $G(i, S)$ denote the players of S with offer time $\tau(\cdot, S)$ strictly less than, equal to, and strictly greater than that of i, respectively.

Definition 6.2.4 Let χ and τ be a cost-sharing method and an offer function, respectively, defined on a universe U. The function τ is *valid for* χ if the following two properties hold for every subset $S \subseteq U$ and player $i \in S$:

(a) $\chi(i, S \setminus T) = \chi(i, S)$ for every subset $T \subseteq G(i, S)$;

(b) $\chi(i, S \setminus T) \geq \chi(i, S)$ for every subset $T \subseteq G(i, S) \cup (E(i, S) \setminus \{i\})$.

In Definition 6.2.4, a player's cost share must remain fixed as players with subsequent offer times are removed, and it can only increase with the deletion of players with equal offer times. The deletion of a player with an earlier offer time imposes no constraints, as such a deletion terminates the iteration and suppresses the values of subsequent cost shares. Also, we impose no explicit constraints on how the offer function τ changes between consecutive iterations.

Example 6.2.5 Consider the universe $U = \{x, y\}$ and the non-cross-monotonic cost-sharing method χ defined by $\chi(y, \{x, y\}) = 1$ and $\chi(x, \{x, y\}) = \chi(y, \{y\}) = \chi(x, \{x\}) = 1/2$. Let τ_x and τ_y denote offer functions satisfying $\tau_x(x, \{x, y\}) < \tau_x(y, \{x, y\})$ and $\tau_y(y, \{x, y\}) < \tau_y(x, \{x, y\})$, respectively. Then τ_x is valid for χ while τ_y is not.

Definition 6.2.6 An *acyclic mechanism* is a mechanism $M(\chi, \tau)$ induced by a cost-sharing method χ and an offer function τ that is valid for χ.

Remark 6.2.7 Acyclic mechanisms are strictly more general than Moulin mechanisms. For example, all sequential mechanisms (see [62]), in which players are exogenously ordered and successively offered service at the current marginal cost, are easily implementable as acyclic mechanisms. These mechanisms are fully budget-balanced and are not generally Moulin mechanisms. Sequential mechanisms are not immediately useful for our purposes, however, as they have poor efficiency and computational complexity properties.

Definition 6.2.4 is easy to satisfy in several applications. Looking ahead, Chapter 7 shows that several well-known algorithms naturally induce a cost-sharing method and an offer function that is valid for it. In all of our applications, the cost share $\chi(i, S)$ of a player corresponds to part of a dual solution to the optimization problem induced by S, and the offer time $\tau(i, S)$ is the time at which player i is "deactivated" by a

primal-dual algorithm. For example, in UFL (Example 2.2.4), there is a one-to-one correspondence between players and dual variables.

Remark 6.2.8 We use the term "acyclic" to reflect the fact that the offer function of an acyclic mechanism orders the remaining players in a way that conceals the non-cross-monotonicity of the underlying cost-sharing method. In particular, Definition 6.2.4 implies that for every subset S of players, the following directed graph acyclic: the vertices are the players of S, and the arc (i, j) is included if and only if $\chi(j, S \setminus \{i\}) < \chi(j, S)$. This consequence of Definition 6.2.4 is reminiscent of but different from the notion of "semi-cross-monotonicity" introduced in [45].

6.3 Properties of Acyclic Mechanisms

The following basic properties of acyclic mechanisms are immediate.

Proposition 6.3.1 *Let χ and τ be a cost-sharing method and an offer function defined on the universe U, and $M(\chi, \tau)$ the induced mechanism.*

(a) For every bid vector b, the mechanism $M(\chi, \tau)$ halts within $|U|$ iterations.

(b) If χ and τ run in polynomial time, then so does $M(\chi, \tau)$.

(c) If χ is β-budget-balanced with respect to a cost function C, then so is $M(\chi, \tau)$.

(d) The mechanism $M(\chi, \tau)$ satisfies no positive transfers and individual rationality.

The rest of this section studies the incentive-compatibility properties of acyclic mechanisms. Our key lemma states that the prices offered to a player can only increase during the execution of an acyclic mechanism. To make this precise, we say that player i is *offered the price p in iteration j* of an acyclic mechanism $M(\chi, \tau)$ if the following conditions hold: first, if S is the set of players remaining at the beginning of the jth iteration, then $i \in S$; second, if a player i^* is chosen for deletion in this iteration, then $\tau(i, S) \leq \tau(i^*, S)$; third, the price p is the cost share $\chi(i, S)$.

We first prove a preliminary result, stating that the price offered to a player by an acyclic mechanism is fixed once a player with a subsequent offer time is offered a price.

Lemma 6.3.2 *Suppose an acyclic mechanism $M(\chi, \tau)$ offers prices to players j and i in an iteration with remaining players S, and $\tau(j, S) < \tau(i, S)$. Then $\chi(j, S)$ is the only price offered to j in subsequent iterations.*

Proof: Let b denote the bid vector and m the iteration with remaining players S. We show that no player of $L(i, S)$ will ever be deleted; thus all removed players lie in $G(j, S)$, and the lemma follows from Definition 6.2.4(a).

We proceed by contradiction, and let ℓ denote the first player of $L(i, S)$ removed at or after iteration m. Let $T \subseteq S$ denote the players of S removed prior to ℓ. Since ℓ was removed, $\chi(\ell, S \setminus T) > b_\ell$. Since i was offered a price in iteration m and $\ell \in L(i, S)$, $\chi(\ell, S) \leq b_\ell < \chi(\ell, S \setminus T)$. By our choice of ℓ, T contains no players of $L(i, S)$, and hence $T \subseteq G(\ell, S)$. But Definition 6.2.4(a) then gives $\chi(\ell, S) = \chi(\ell, S \setminus T)$, a contradiction. ■

Corollary 6.3.3 *If an acyclic mechanism $M(\chi, \tau)$ offers a price to player i when the remaining set of players is S, then M never deletes a player of $L(i, S)$.*

Proof: Let b denote the bid vector. Since i is offered a price when the remaining set of players is S, $\chi(j, S) \leq b_j$ for every $j \in L(i, S)$. Lemma 6.3.2 implies that every player $j \in L(i, S)$ will be offered the same price $\chi(j, S)$ in subsequent iterations, and hence no such player will ever be deleted. ■

We now show that acyclic mechanisms only offer ascending sequences of prices.

Lemma 6.3.4 *If an acyclic mechanism $M(\chi, \tau)$ offers a player i the price p_i^1 in some iteration and the price p_i^2 in a subsequent iteration, then $p_i^1 \leq p_i^2$.*

Proof: Let S denote the remaining players in the earlier iteration, so $p_i^1 = \chi(i, S)$. Since i was offered a price in this iteration, Corollary 6.3.3 implies that no player of $L(i, S)$ will be deleted in this or subsequent iterations. The lemma now follows from Definition 6.2.4(b). ■

Lemma 6.3.4 implies that acyclic mechanisms are strategyproof.

Theorem 6.3.5 *Every acyclic mechanism is strategyproof.*

Since we generalize Theorem 6.3.5 in Theorem 6.3.8 below, we omit its short proof.

The next example shows that mechanisms induced by invalid offer functions are not generally truthful.

Example 6.3.6 Define U, χ, and τ_y as in Example 6.2.5. The mechanism $M(\chi, \tau_y)$ induced by χ and τ_y is not strategyproof. To see this, suppose that $v_y = 3/4$ and $b_x = 1/4$. If player y bids truthfully, it is not served and receives zero utility. If it bids at least 1, however, it is served at the price $1/2$ and receives positive utility.

Recall from Remark 4.1.3 that Moulin mechanisms are groupstrategyproof (GSP). The next example shows that acyclic mechanisms need not be GSP.

Example 6.3.7 Define U, χ, and τ_x as in Example 6.2.5. Since τ_x is valid for χ, the acyclic mechanism $M(\chi, \tau_x)$ is strategyproof. It is not GSP, however. To see this, set $v_x = 1/2$ and $v_y = 1$. In every possible execution of $M(\chi, \tau_x)$, player x receives zero utility. The coalition $\{x, y\}$ can manipulate the mechanism by bidding $b_x = 0$ and $b_y = 1$; player x obtains the same utility as with truthful bidding, and player y obtains strictly more.

We conclude this section by proving that acyclic mechanisms are weakly groupstrategyproof (recall Section 2.4), and thus nearly match the incentive-compatibility guarantee of Moulin mechanisms.

Theorem 6.3.8 *Every acyclic mechanism is weakly groupstrategyproof (WGSP).*

Proof: Let $M(\chi, \tau)$ be an acyclic mechanism defined on the universe U. Recall from Chapter 2 that a mechanism is WGSP if no coordinated false bid by a coalition of players can strictly increase the utility of every player in the coalition. Fix a coalition $T \subseteq U$, a valuation v_i and a bid b_i for every player $i \in T$, and bids b_{-T} for the players not in T. Let $\mathcal{E}_v$ and $\mathcal{E}_b$ denote the executions of M for the bid vectors (v_T, b_{-T}) and

(b_T, b_{-T}), respectively. Let (S, p) and (S', p') denote the outcomes of these executions. We claim that $u_i(S, p) \geq u_i(S', p')$ for some $i \in T$.

There are three cases. First, if no player of T is deleted in $\mathcal{E}_v$ or $\mathcal{E}_b$, then these executions terminate with identical outcomes (S, p) and (S', p'), and the claim holds. Second, if some player $i \in T$ is deleted in $\mathcal{E}_b$, then $u_i(S', p') = 0$. Since $u_i(S, p) \geq 0$ by the individual rationality of $M(\chi, \tau)$ (Proposition 6.3.1(d)), the claim holds. For the final case, assume that $T \subseteq S'$ and $T \not\subseteq S$, and let i be the first player of T deleted in $\mathcal{E}_v$, say in the jth iteration; obviously, $u_i(S, p) = 0$. The executions $\mathcal{E}_v$ and $\mathcal{E}_b$ are identical up to their jth iterations, and i is offered the same price p_i^* in both executions. Since i is deleted in $\mathcal{E}_v$, $p_i^* > v_i$. By Lemma 6.3.4, $p_i' \geq p_i^* > v_i$. Thus $u_i(S', p') < 0 = u_i(S, p)$, completing the proof. ■

Remark 6.3.9 The proof of Theorem 6.3.8 immediately implies an incentive-compatibility guarantee somewhat stronger than WGSP: for every acyclic mechanism, every deviation by a coalition that strictly increases the utility of one of its members either decreases the utility of or prevents service to another member (cf., Example 6.3.7).

Remark 6.3.10 Not all WGSP mechanisms are acyclic; see Juarez [49]. For example, the following mechanism for two players is WGSP but not acyclic: offer service to the first player at a fixed price, and to the second at a price that is a strictly increasing function of the first player's bid. The key point is that acyclic mechanisms are based on cost-sharing methods—mechanisms based on cost-sharing methods satisfy the property that for every bid vector, the allocation determines the prices. This condition is violated by the mechanism in this example.

Characterizing the class of WGSP mechanisms and its relationship to acyclic mechanisms is an interesting direction for future research.

6.4 Acyclic mechanisms: Summary of Applications

As the following table summarizes, acyclic mechanisms yield exponentially superior budget-balance and efficiency compared to the best Moulin mechanisms for set cover

and vertex cover cost-sharing problems. For UFL cost-sharing problems, acyclic mechanisms achieve better budget-balance than that achievable by Moulin mechanisms. All of these mechanisms have polynomial time implementations. See Chapter 7 for details. The Moulin lower bounds are discussed in Section 4.5.2. Recall that k is the size of the optimal solution, n is the size of the universe of players, and all of the results are worst case bounds with respect to valuation profiles and problem instances of the problem family.

Problem	Moulin lower bounds	Acyclic upper bounds
Vertex Cover	α, $\beta = \Omega(k^{1/3})$	$\alpha = O(\log k)$, $\beta = 2$
Set Cover	α, $\beta = \Omega(\sqrt{k})$	α, $\beta = O(\log n \log k)$
Metric UFL	$\alpha = \Omega(\log k)$, $\beta = 3$	$\alpha = O(\log k)$, $\beta = 1.61$

If we drop the requirement of a computationally efficient implementation, Bleischwitz et al. [12] show that there is an acyclic mechanism that is perfectly budget-balanced and $O(\log k)$-approximate for all subadditive cost-sharing problems (recall from Figure 2.1 that this all the cost-sharing problems we study in this thesis are subadditive). This mechanism is based on the egalitarian cost-sharing method of Datta and Ray [28], and is similar in spirit to the DMV mechanism for NMUFL cost-sharing problems from Chapter 7. Section 8.1 identifies a non-acyclic , no-deficit mechanism that is $O(\log k)$-approximate for an even wider class of cost-sharing problems—those that have monotone cost-functions—however, though this mechanism is no-deficit, it is not budget-balanced, it is SP but not WGSP.

6.5 Notes

6.5.1 Acycicity

Moulin [62] defines a class of cost-sharing mechanisms called incremental cost-sharing mechanisms, which are acyclic. The idea is to arrange players in a sequence and check if a player is willing to pay its marginal cost-share. (This paper also defines *Generalized incremental cost-sharing methods* that vary the offer sequence based on which

previous offers were accepted or rejected, just like some of our acylic mechanisms.) These mechanisms apply to cost-sharing problems with multiple levels of service, are strategyproof and budget-balanced. However, they have poor efficiency for the cost-sharing problems we study.

Bleischwitz et al. [13] propose non-Moulin mechanisms for symmetric subadditive costs that are groupstrategyproof, but improve on the best budget-balance achievable by Moulin mechanisms. The mechanisms they propose use deletion priorities just as acyclic mechanisms do, but use only two priorities and two prices, and are applicable only to symmetric cost functions.

Chapter 7

Acyclic Mechanisms via Primal-Dual

This chapter demonstrates how several well-known primal-dual algorithms naturally induce acyclic mechanisms. All of these algorithms were designed prior to the development of Moulin mechanisms, but since the cost-sharing methods induced by these algorithms are not cross-monotonic, they could not be used to construct such mechanisms.

Section 7.1 gives a self-contained account of two primal-dual algorithms, shows how each induces a cost-sharing method and an offer function in a natural way, and notes that these cost-sharing methods are not cross-monotonic. Section 7.2 proves that these cost-sharing methods are valid (Definition 6.2.4). Section 7.3 proves that the mechanisms induced by these cost-sharing methods match or, in most cases, improve upon the best approximation guarantees possible for Moulin mechanisms.

7.1 Primal-Dual Algorithms and Cost-Sharing Methods

Good Moulin and acyclic mechanisms depend on good cost-sharing methods—functions that take as input a subset S of players, and output both a feasible solution for the

optimization problem induced by S and cost shares for the players that approximately cover the cost of this solution. This goal is strongly reminiscent of that achieved by *primal-dual algorithms*—algorithms that output a feasible solution to an optimization problem, as well as a "dual solution" that certifies the near-optimality of the solution. This parallel has already been exploited in the design of Moulin mechanisms (Chapter 5), and we demonstrate that this connection is equally powerful in the design of acyclic mechanisms.

This section describes two non-cross-monotonic cost-sharing methods for NMUFL cost-sharing problems, induced by well-known primal-dual algorithms. Sections 7.2–7.3.2 leverage these methods to design acyclic mechanisms with good performance guarantees, and in particular establish the upper bounds listed in Section 6.4.

7.1.1 The PD Mechanism for NMUFL Problems

Primal-dual algorithms lead to cost-sharing methods in a generic way. Our first illustration is a NMUFL algorithm that forms the basis of our 2-budget-balanced acyclic mechanism for Vertex Cover problems. Consider a NMUFL problem (Example 2.2.3) defined by a universe U, facilities F, and facility and connection costs f and c, respectively. A *star* is a pair (q, T), where $q \in F$ is a facility and T is a subset of demands. The cost $c(q, T)$ of the star (q, T) is defined as $f_q + \sum_{i \in T} c(q, i)$. Let $\mathrm{S}(S)$ denote the set of all stars involving only players of S. The following integer program is an exact formulation of the NMUFL problem induced by a subset $S \subseteq U$ of players:

$$
\begin{aligned}
& \text{Min} && \sum_{(q,T) \in (S)(S)} c(q,T) x_{qT} && \\
& \text{subject to:} && && \\
(IP(S)) \qquad & && \sum_{(q,T) \in \mathrm{S}(S) \,:\, i \in T} x_{qT} \geq 1 && \text{for all } i \in S \\
& && x_{qT} \in \{0,1\} && \text{for all } (q,T) \in \mathrm{S}(S).
\end{aligned}
$$

There is one decision variable per star (q, T), and setting a variable $x_{qT} = 1$ should be interpreted as opening the facility q and assigning all of the demands of T to q.

There is one constraint per player i of S, stating that at least one star containing i must be selected. Every feasible solution of the NMUFL instance induced by S can be mapped easily to a feasible solution of $IP(S)$ of no greater cost, and conversely.

Replacing the last constraint of $IP(S)$ by $x_{qT} \geq 0$ for every star $(q, T) \in \mathcal{C}(S)$ yields a linear programming relaxation. The dual linear program of this relaxation is

$$
\begin{aligned}
& \text{Max} && \sum_{i \in S} \alpha_i && \\
& \text{subject to:} && && \\
(D(S)) \qquad & && \sum_{i \in T} \alpha_i \leq c(q,T) && \text{for all } (q,T) \in \mathcal{C}(S) \\
& && \alpha_i \geq 0 && \text{for all } i \in S.
\end{aligned}
$$

There is a one-to-one correspondence between the dual decision variables α_i and the players of S. By weak linear programming duality (see e.g. [21]), the objective function value of every feasible solution α of $D(S)$ provides a lower bound on the objective function value of every feasible solution x of $IP(S)$:

$$\sum_{i \in S} \alpha_i \leq \sum_{(q,T) \in \mathcal{C}(S)} c(q,T) x_{qT}. \tag{7.1}$$

Why are these mathematical programs useful for designing cost-sharing methods? Suppose an algorithm is guaranteed to return feasible solutions x^* and α^* to $IP(S)$ and $D(S)$, respectively, such that

$$\sum_{(q,T) \in \mathcal{C}(S)} c(q,T) x^*_{qT} \leq \beta \cdot \sum_{i \in S} \alpha^*_i. \tag{7.2}$$

Interpret x^* as a feasible solution to the NMUFL instance induced by S, and each dual variable α^*_i scaled by a factor β as a cost share $\chi(i, S)$. By inequalities (7.1) and (7.2), this cost-sharing method χ is β-budget-balanced. Thus, designing a β-budget-balanced cost-sharing method reduces to designing a β-approximation algorithm with performance guarantee established via the primal-dual inequalities (7.1) and (7.2). There are several broadly applicable algorithmic paradigms for designing

1. Initialize $\alpha_i = 0$ for all $i \in S$, $x_{qT} = 0$ for all $(q, T) \in \mathcal{C}(S)$, and the time t to 0. All players of S are active and unconnected.

2. While active players remain:

   (a) Uniformly increase α_i for every active player $i \in S$, until $\sum_{i \in T} \alpha_i = c(q, T)$ for some star (q, T) containing at least one active player. Increase t by the same amount.

   (b) Choose such a star (q, T) and let W denote the players already connected to q. Set $x_{qW} = 0$ and $x_{qT \cup W} = 1$. Deactivate and connect to q all of the players of T.

Figure 7.1: The PD algorithm for NMUFL.

approximation algorithms of this type (see e.g. [73]).

Cost-sharing methods do not automatically yield truthful cost-sharing mechanisms unless they satisfy additional constraints (cf., Definition 6.2.4). This motivates concentrating on a particularly simple class of algorithms: *primal-dual algorithms*. Roughly, a primal-dual algorithm constructs feasible solutions to a (primal) optimization problem and the dual of its linear relaxation in tandem, maintaining inequalities (7.1) and (7.2) as invariants during its execution. Typically, the algorithm begins with the all-zero primal and dual solutions, and primal feasibility is attained only at termination.

Figure 7.1 displays a primal-dual algorithm for the NMUFL problem, which we call the *PD algorithm*. (This algorithm is well known; see [43] and [73, Chapter 15].) At the beginning of the algorithm, all dual variables are zero and all stars are unchosen. The algorithm also maintains a notion of time, initially zero. A player is *active* if it is not contained in a chosen star, and *inactive* otherwise. In each iteration, the dual variables α_i of all active players are increased simultaneously at unit rate until the dual constraint for some unchosen star (q, T) becomes tight: $\sum_{i \in T} \alpha_i = c(q, T)$. When such a star becomes tight, it is chosen and the active players of T are deactivated; ties are broken in an arbitrary but consistent way. Such a star can be found in polynomial time, even though there are an exponential number of stars (see [46]). As long as

there is a feasible solution with finite cost, the algorithm will terminate with such a solution. By Step 2a, it maintains dual feasibility as an invariant.

Lemma 7.1.1 (PD Invariant) *For every NMUFL instance, the PD algorithm terminates with feasible solutions to both the primal (the linear relaxation of $IP(S)$) and the dual $D(S)$.*

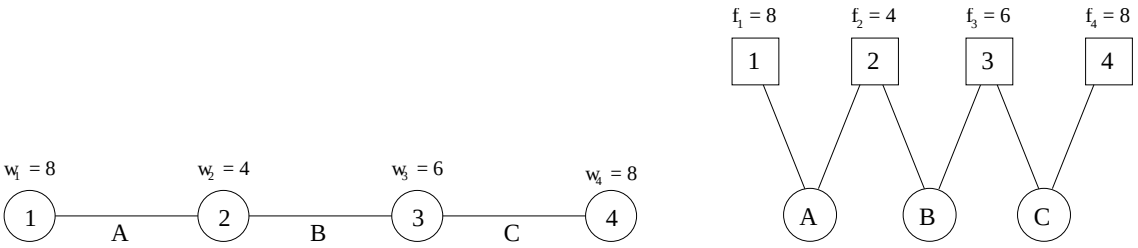
This primal-dual algorithm induces a cost-sharing method χ_{PD} for the given NMUFL problem: given a subset $S \subseteq U$, return the feasible NMUFL solution computed by this algorithm, and set each cost share $\chi_{PD}(i, S)$ to the final value of the dual variable α_i scaled by d_{max}, where d_{max} denotes the maximum number of facilities to which a player can be assigned at finite cost. The important application of this mechanism is to Vertex Cover problems, for which $d_{max} = 2$ (cf., Figure 7.2(b)).

The cost-sharing method χ_{PD} is not cross-monotonic, even in the special case of Vertex Cover cost-sharing problems, and thus does not yield a truthful Moulin mechanism.

Example 7.1.2 Consider the Vertex Cover cost-sharing problem shown in Figure 7.2(a), with vertex weights as shown. This problem corresponds to the NMUFL instance shown in Figure 7.2(b). Edges in the figure represent zero connection costs; non-edges represent infinite connection costs.

We claim that $\chi_{PD}(C, \{B, C\}) < \chi_{PD}(C, \{A, B, C\})$, which is a violation of cross-monotonicity. To compute the cost share $\chi_{PD}(C, \{A, B, C\})$, we execute the primal-dual algorithm of Figure 7.1 with all three players present. At time 2, the star $(2, \{A, B\})$ becomes tight and players A and B are deactivated. At time 4, the star $(3, \{B, C\})$ becomes tight, C is deactivated, and algorithm terminates with $\chi_{PD}(C, \{A, B, C\})$ equal to 8 (recall the scaling by $d_{max} = 2$). If we remove player A and execute the algorithm, the star $(3, \{B, C\})$ is the first to become tight, at time $t = 3$. The algorithm halts at this point with $\chi_{PD}(C, \{B, C\}) = 6$ which is strictly less than $\chi_{PD}(C, \{A, B, C\})$.

The primal-dual algorithm in Figure 7.1 also induces an offer function: set $\tau_{PD}(i, S)$ equal to the time at which player i is deactivated in Step 2b of the algorithm. We



(a) Vertex Cover problem (b) Equivalent NMUFL problem

Figure 7.2: Example 7.1.2. The cost-sharing method χ_{PD} is not cross-monotonic.

call the mechanism $M(\chi_{PD}, \tau_{PD})$ induced by χ_{PD} and τ_{PD} the *PD mechanism*. Sections 7.2–7.3.2 establish that the PD mechanism is acyclic and, for Vertex Cover problems, is 2-budget-balanced (Definition 2.3.1) and $O(\log k)$-approximate (Equation (3.1)).

Example 7.1.3 In Example 7.1.2, χ_{PD} fails to be cross-monotonic because

$$\chi_{PD}(C, \{B, C\}) = 6 < 8 = \chi_{PD}(C, \{A, B, C\}).$$

On the other hand, $\tau_{PD}(A, \{A, B, C\}) = 2 < 4 = \tau_{PD}(C, \{A, B, C\})$; in words, the PD mechanism offers player A its first-round price of 4 before it offers player C its first-round price of 8. Cross-monotonicity fails only when player A refuses this price; in this case, the PD mechanism makes no first-round offer to player C, thereby suppressing the non-cross-monotonicity.

7.1.2 The DMV Mechanism for NMUFL Problems

Next we give a second NMUFL cost-sharing method that leads to a mechanism that outperforms the PD mechanism for general NMUFL and metric UFL problems (but not for Vertex Cover problems). The method is again defined via a primal-dual algorithm for the programs $IP(S)$ and $D(S)$ (Figure 7.3). See also Remark 7.1.7 below for a greedy interpretation of this algorithm.

This algorithm differs from the PD algorithm primarily in its choice of the star (q, T) in the main loop. First, only stars (q, T) entirely composed of active players T are

1. Initialize $\alpha_i = 0$ for all $i \in S$, $x_{qT} = 0$ for all $(q, T) \in \mathcal{C}(S)$, and the time t to 0. All players of S are active and unconnected, all facilities are closed.

2. While active players remain:

   (a) Uniformly increase α_i for every active player $i \in S$, until for some star (q, T) of active players T: (i) q is closed and $\sum_{i \in T} \alpha_i = c(q, T)$; or (ii) q is open and $\sum_{i \in T} \alpha_i = \sum_{i \in T} c(q, i)$. Increase t by the same amount.

   (b) Choose such a star (q, T). Deactivate and connect to q all of the players of T. In case (i), open q and set $x_{qT} = 1$. In case (ii), let W denote the players already connected to q, and set $x_{qW} = 0$ and $x_{qT \cup W} = 1$.

Figure 7.3: The DF algorithm for NMUFL.

eligible for selection. Second, the selection criterion depends on whether or not the facility q appears in a previously chosen star. These rules are designed to maintain the invariant that the current primal and dual solutions have equal objective function value. Primal-dual algorithms of this type are sometimes called *dual-fitting algorithms* [46], so we call this algorithm the *DF algorithm*.

Lemma 7.1.4 (DF Invariant) *After each iteration of Step 2 of the DF algorithm,*

$$\sum_{(q,T) \in \mathcal{C}(S)} c(q, T) x_{qT} = \sum_{i \in I} \alpha_i,$$

where I denotes the current set of inactive players.

We omit the straightforward inductive proof. See also [43, 46] for alternative descriptions of the DF algorithm, including polynomial-time implementations.

The DF algorithm only constrains dual variable growth in Step 2 via a strict subset of the dual constraints—stars comprising only active players—and the algorithm need not maintain dual feasibility. However the dual variables suitably scaled do satisfy dual feasibility.

Lemma 7.1.5 *For every NMUFL instance, the DF algorithm terminates with a solution (α_is) such that the solution scaled down by a factor $\mathcal{H}_{|U|}$ is dual feasible.*

Lemma 7.1.5 follows from the well-known dual-fitting analysis of the greedy Set Cover algorithm (see [20, 43] and [73, Chapter 13]).

Like the PD algorithm, the DF algorithm induces a cost-sharing method χ_{DF} and an offer function τ_{DF}. Given a subset $S \subseteq U$, the method χ_{DF} returns the feasible solution computed by the DF algorithm for the NMUFL instance induced by S, and cost shares equal to the dual variables. The offer time $\tau_{DF}(i, S)$ is defined as the time at which player i is deactivated in Step 2b of the DF algorithm. We call the induced mechanism $M(\chi_{DF}, \tau_{DF})$ the *DMV mechanism*, as special cases of this mechanism were studied in [26]. Sections 7.2–7.3.2 prove acyclicity of and good performance guarantees for the DMV mechanism.

Remark 7.1.6 In Example 7.1.2, $\chi_{DF}(C, \{A, B, C\}) = 6$ while $\chi_{DF}(C, \{B, C\}) = 3$. Thus χ_{DF} is not cross-monotonic. Minor modifications to this example show that χ_{DF} also fails to be cross-monotonic in the special case of metric UFL problems.

Remark 7.1.7 The DF algorithm can also be interpreted as a greedy algorithm [46]. Given a partial solution to a NMUFL instance, define the *cost effectiveness* of a star (q, T) as $c(q, T)/|T|$ if q is closed and as $\sum_{i \in T} c(q, i)/|T|$ if q is already open. The main loop of the DF algorithm (Step 2) is equivalent to repeatedly choosing the star of active players with smallest cost effectiveness. The dual variable of each participating player is set to the cost effectiveness of the star.

Remark 7.1.8 The DMV mechanism has an alternative description in which all of the successive invocations of the underlying DF algorithm are combined into a single one. In particular, the mechanisms in [26] are described in this way.

7.2 Acyclicity

We now prove that both mechanisms defined in Section 7.1 are acyclic.

The proofs of acyclicity for the PD and DMV NMUFL mechanisms are essentially the same. We begin by noting that cost shares and offer times are equal in the DF method, and differ only by a fixed scaling factor in the PD method.

Lemma 7.2.1 *For every NMUFL problem with universe U, subset $S \subseteq U$, and player $i \in S$:*

(a) $\chi_{PD}(i, S) = d_{max} \cdot \tau_{PD}(i, S)$;

(b) $\chi_{DF}(i, S) = \tau_{DF}(i, S)$.

Proof: In the PD algorithm, every dual variable α_i is increased at unit rate from time 0 to the the time at which the corresponding player is deactivated, which by definition is $\tau_{PD}(i, S)$. Since $\chi_{PD}(i, S)$ is the final value of α_i scaled by a factor d_{max}, (a) follows.

By the same argument, after Step 2 of the DF algorithm, $\alpha_i = \tau_{DF}(i, S)$ for every player $i \in S$. Since $\chi_{DF}(i, S)$ is is the final value of $\tau_{DF}(i, S)$, (b) follows. ■

We can now prove that the PD mechanism is acyclic and hence, by Theorem 6.3.8, weakly groupstrategyproof (WGSP).

Theorem 7.2.2 *The PD mechanism is acyclic.*

Proof: Fix a NMUFL cost-sharing problem and let $\mathcal{E}(S)$ denote the execution of the PD algorithm on the NMUFL instance induced by a subset $S \subseteq U$ of players. Fix $S \subseteq U$ and a player $i \in S$. Let (q, A) denote the star chosen at time $\tau_{PD}(i, S)$ in $\mathcal{E}(S)$ that contains player i.

To establish Definition 6.2.4(a), choose $T \subseteq G(i, S)$. Since the offer time of a player equals the earliest time at which a star containing it is chosen by the PD algorithm, no star chosen in $\mathcal{E}(S)$ at or before time $\tau_{PD}(i, S)$ includes a player of T. By induction on the main loop, the executions $\mathcal{E}(S)$ and $\mathcal{E}(S \setminus T)$ are identical up to and at the time $\tau_{PD}(i, S)$. As a result, $\tau_{PD}(i, S \setminus T) = \tau_{PD}(i, S)$. By Lemma 7.2.1(a), $\chi_{PD}(i, S \setminus T) = \chi_{PD}(i, S)$.

The proof of Definition 6.2.4(b) is similar. Fix a subset $T \subseteq G(i, S) \cup (E(i, S) \setminus \{i\})$ of players. The executions $\mathcal{E}(S)$ and $\mathcal{E}(S \setminus T)$ are identical prior to the time $\tau_{PD}(i, S)$.

Thus $\tau_{PD}(i, S \setminus T) \geq \tau_{PD}(i, S)$ and, by Lemma 7.2.1(a), $\chi_{PD}(i, S \setminus T) \geq \chi_{PD}(i, S)$. ■

An identical argument proves the acyclicity of the DMV mechanism.

Theorem 7.2.3 *The DMV mechanism is acyclic.*

7.3 Improved Approximation Guarantees

This section proves tight upper and lower bounds on the approximate budget-balance (Section 7.3.1) and efficiency (Section 7.3.2) of the acyclic mechanisms defined in the previous section.

7.3.1 Budget-Balance Guarantees

This section shows how budget-balance guarantees for both of the mechanisms defined in Section 7.1 follow easily from existing work in the approximation algorithms literature.

The PD Mechanism for NMUFL and Vertex Cover Problems

We next show that the PD mechanism for NMUFL problems is d_{max}-budget-balanced, where d_{max} denotes the maximum number of facilities to which a player can be assigned at finite cost. Extending the well-known analysis of primal-dual Set Cover algorithms implies the following guarantee for the PD algorithm.

Lemma 7.3.1 *For every NMUFL instance, the PD algorithm computes a primal solution* $\{x^*_{qT}\}_{(q,T)\in\mathcal{C}(S)}$ *and a dual solution* $\{\alpha^*_i\}_{i\in S}$ *satisfying*

$$\sum_{(q,T)\in\mathcal{C}(S)} c(q,T)x^*_{qT} \leq d_{max} \cdot \sum_{i\in S} \alpha^*_i.$$

The intuition behind Lemma 7.3.1 is that every increase of a dual variable in the PD algorithm only contributes to dual constraints of d_{max} different facilities, and thus

the primal cost will only exceed the sum of the dual variables by a d_{max} factor. The details are essentially the same as those for Set Cover algorithms, which appear in Hochbaum [43] and Vazirani [73, Chapter 15].

In addition, the dual solution computed by the PD algorithm is feasible (Lemma 7.1.1), and hence the computed primal and dual solutions satisfy weak duality (7.1). As discussed in Section 7.1.1, since the cost shares of the PD method are the dual variables computed by the PD algorithm, scaled by a factor d_{max}, budget-balance of the PD method and mechanism follow.

Theorem 7.3.2 *For every NMUFL cost-sharing problem, the PD mechanism is* (d_{max})*-budget-balanced.*

For vertex cover, d_{max} is 2. Recall that every Moulin mechanism for Vertex Cover problems is $\Omega(|U|^{1/3})$-budget-balanced [45]. Assuming the Unique Games Conjecture [52], the budget-balance guarantee in Theorem 7.3.2 is the best possible for a polynomial-time mechanism for small values of d_{max} [53].

The PD mechanism has poor budget-balance in NMUFL problems in which d_{max} is large. In these cases, the DMV mechanism achieves a superior performance guarantee. In particular, the following lemma is obvious from Lemma 7.1.4.

Lemma 7.3.3 *For every NMUFL instance, the DF algorithm computes a feasible primal solution* $\{x^*_{qT}\}_{(q,T)\in\mathcal{C}(S)}$ *and an (infeasible) dual solution* $\{\alpha^*_i\}_{i\in S}$ *satisfying*

$$\sum_{(q,T)\in\mathcal{C}(S)} c(q,T)x^*_{qT} = \sum_{i\in S}\alpha^*_i.$$

Also, by Lemma 7.1.5, the dual solution computed by the DF algorithm is feasible if all the dual variables are scaled by $\mathcal{H}_{|U|}$. As with Theorem 7.3.2, budget-balance follows.

Theorem 7.3.4 ([26]) *For every NMUFL cost-sharing problem with universe* U*, the DMV mechanism is* $\mathcal{H}_{|U|}$*)-budget-balanced.*

Every Moulin mechanism for NMUFL problems is $\Omega(\sqrt{|U|})$-budget-balanced [45]. Under standard complexity assumptions, the budget-balance guarantee in Theorem 7.3.4 is the best possible for polynomial-time NMUFL mechanisms [31].

The DMV mechanism can achieve radically better budget-balance for the special case of metric UFL problems. Jain et al. [46] proved the following.

Lemma 7.3.5 ([46]) *For every metric UFL instance, the metric DF algorithm terminates with a dual feasible solution if all the dual variables are scaled down by a factor* 1.861*.*

Budget-balance of the DMV mechanism follows.

Theorem 7.3.6 ([26]) *For every metric UFL cost-sharing problem, the metric DMV mechanism is* (1.861)*-budget-balanced.*

No metric UFL Moulin mechanism is better than 3-budget-balanced [45].

Remark 7.3.7 The budget-balance guarantee in Theorem 7.3.6 can be improved using a slightly different mechanism. Jain et al. [46] suggested a modification of the DF algorithm for metric UFL and proved that scaling its dual variables by a factor of 1.61 is enough to recover dual feasibility. The proofs of Theorems 7.2.2 and 7.2.3 carry over to show that the mechanism induced by this refined algorithm is acyclic. As in Theorem 7.3.6, this mechanism is 1.61-budget-balanced.

7.3.2 Efficiency Guarantees

This section proves matching upper and lower bounds on the approximate efficiency achieved by the mechanisms defined in Section 7.1. We obtain efficiency guarantees for the PD and DMV mechanisms for NMUFL problems as a consequence of the following more general result.

Theorem 7.3.8 *Let $M(\chi, \tau)$ be a (β)-budget-balanced acyclic mechanism for a cost-sharing problem C with universe U such that:*

(P1) for some constant $\gamma > 0$, $\chi(i, S) = \gamma \cdot \tau(i, S)$ for all $S \subseteq U$ and $i \in S$;

(P2) for every $S \subseteq U$ and $T \subseteq S$,

$$\sum_{i \in T} \chi(i, S) \leq \beta \cdot C(T). \tag{7.3}$$

Then, $M(\chi, \tau)$ is $(\beta \cdot \mathcal{H}_k + \beta - 1)$-approximate, where k is the size of an optimally efficient solution.

Property (P1) states that offer times are proportional to cost shares. Property (P2) can be interpreted as a "stability" property in the spirit of the core (see e.g. [69]), demanding that each coalition T has no incentive to secede from the mechanism and seek service elsewhere at cost $C(T)$.

Theorem 7.3.8 has immediate implications for the PD and DMV mechanisms.

Corollary 7.3.9 *For every NMUFL cost-sharing problem, the PD mechanism is $O(d_{max} \cdot \log k)$-approximate, where d_{max} is the largest number of facilities to which a demand can be assigned at finite cost. Here k is the size of an optimally efficient solution.*

Proof: To check condition (7.3), fix a NMUFL problem with universe U and subsets $T \subseteq S \subseteq U$. Let χ_{PD} denote the PD cost-sharing method. The cost shares $\{\chi_{PD}(i, S)\}_{i \in S}$ scaled down by a factor d_{max} form a feasible solution to the dual program $D(S)$ of Section 7.1.1 (Lemma 7.1.1). The subset of cost shares $\{\chi_{PD}(i, S)\}_{i \in T}$ scaled down by a factor d_{max} form a feasible solution to the dual program $D(T)$. Condition (7.3) follows from weak duality.

The corollary is now immediate from Lemma 7.2.1(a), Theorem 7.3.2, and Theorem 7.3.8. ■

For example, for Vertex Cover problems, the PD mechanism is $O(\log k)$-approximate. Every Moulin mechanism for such problems is $\Omega(k^{1/3})$-approximate [45, 70].

Corollary 7.3.10 *For every NMUFL cost-sharing problem, the DMV mechanism is $O(\log n \cdot \log k)$-approximate. Here k is the size of an optimally efficient solution, and n is the size of the universe U.*

Proof: Immediate from Lemma 7.1.5, Lemma 7.2.1(b), Theorem 7.3.4, and Theorem 7.3.8. ■

Recall that every Moulin mechanism for NMUFL problems is $\Omega(\sqrt{n})$-approximate [45, 70].

Remark 7.3.11 Analogous to the above theorem, the 1.861-budget-balanced metric UFL mechanism (Theorem 7.3.6) and its variant discussed in Remark 7.3.7 are $O(\log k)$-approximate.

Our proof of Theorem 7.3.8 depends on two lemmas. The first bounds the service cost incurred by the mechanism in terms of the cost of the optimal solution and part of the excluded valuations of an optimal solution.

Lemma 7.3.12 *Let $M = M(\chi, \tau)$ be a no-deficit acyclic mechanism for a cost-sharing problem C with universe U that satisfies property (P2) of Theorem 7.3.8. Let v be a valuation profile for U, S the outcome of M on input v, and S^* the outcome with optimal social cost. Then,*

$$C_M(S) \leq \left(\beta \cdot C(S^*) + \sum_{i \in S \setminus S^*} v_i \right).$$

Proof: Since M satisfies the no-deficit condition,

$$C_M(S) \leq \left(\sum_{i \in S \cap S^*} \chi(i, S) + \sum_{i \in S \setminus S^*} \chi(i, S) \right). \tag{7.4}$$

By property (P2) and since C is nondecreasing,

$$\sum_{i \in S \cap S^*} \chi(i, S) \leq \beta \cdot C(S \cap S^*) \leq \beta \cdot C(S^*). \tag{7.5}$$

By individual rationality (Proposition 6.3.1(d)), $\chi(i, S) \leq v_i$ for every $i \in S \setminus S^*$; combining this with inequalities (7.4) and (7.5) proves the lemma. ■

The second lemma upper bounds the excluded valuation of the mechanism in terms of the service cost of an optimal solution.

Lemma 7.3.13 *Let $M = M(\chi, \tau)$ be an acyclic mechanism for a cost-sharing problem C with universe U of k players that satisfies properties (P1) and (P2) (for some β) of Theorem 7.3.8. Let v be a valuation profile for U, S the outcome of M on input v, and S^* the outcome with optimal social cost. Then,*

$$\sum_{i \in S^* \setminus S} v_i \leq \beta \cdot \mathcal{H}_k \cdot C(S^*).$$

Here $k = |S^*|$.

Proof: Let $\ell = |S^* \setminus S|$ and rename the players so that player i is the ith player of $S^* \setminus S$ to be deleted by M on input v. Let S_i denote the set of players from which i is deleted by M. We prove that

$$\chi(i, S_i) \leq \frac{\beta \cdot C(S^*)}{\ell - i + 1} \tag{7.6}$$

for every $i \in \{1, 2, \ldots, \ell\}$; here β is defined as in the statement of Theorem 7.3.8. Player i's deletion from S_i implies that $v_i < \chi(i, S_i)$; summing (7.6) over all players of $S^* \setminus S$ then yields the lemma.

Fix a player i of $S^* \setminus S$. We first claim that, when player i is deleted, its offer time $\tau(i, S_i)$ is minimum among the remaining players $\{i, i+1, \ldots, \ell\}$ of $S^* \setminus S$. If not, there is a player $j > i$ of $S^* \setminus S$ with $\tau(j, S_i) < \tau(i, S_i)$. Since i is offered a price in the iteration it is deleted, Corollary 6.3.3 implies that $L(i, S_i) \subseteq S$. But $j \in L(i, S_i) \cap (S^* \setminus S)$, a contradiction.

This claim and property (P1) imply that, when player i is deleted, its cost share $\chi(i, S_i)$ is minimum among the remaining players of $S^* \setminus S$. Property (P2) and the fact that C is nondecreasing give a bound on the sum of the cost shares of these players:

$$\sum_{j=i}^{\ell} \chi(j, S_i) \leq \beta \cdot C(\{i, i+1, \ldots, \ell\}) \leq \beta \cdot C(S^*);$$

since player i's cost share is the smallest of the $(\ell - i + 1)$ remaining players of $S^* \setminus S$, it is at most $\beta \cdot C(S^*)/(\ell - i + 1)$. This establishes (7.6) and completes the proof. ■

Theorem 7.3.8 now follows easily.

Proof of Theorem 7.3.8: Fix a cost-sharing problem with universe U and a valuation profile v for U. Applying Lemmas 7.3.12 and 7.3.13, we have:

$$\begin{aligned} C_M(S) + \sum_{i \notin S} v_i &\leq \beta \cdot C(S^*) + \sum_{i \in S \setminus S^*} v_i + \beta \cdot \mathcal{H}_k \cdot C(S^*) + \sum_{i \in U \setminus (S \cup S^*)} v_i \\ &\leq (\beta \cdot \mathcal{H}_k + \beta) \cdot C(S^*) + \sum_{i \notin S^*} v_i \end{aligned}$$

Rearranging terms then proves the theorem.

■

Remark 7.3.14 Lemma 7.3.13 and Theorem 7.3.8 continue to hold if property (P1) is replaced by the weaker assumption that, for every subset $S \subseteq U$ of players and $i, j \in S$, $\chi(i, S) < \chi(j, S)$ if and only if $\tau(i, S) < \tau(j, S)$.

Our final result in this section shows that the logarithmic factor in Theorem 7.3.8 cannot be removed: even for extremely simple cost-sharing problems, *every* $O(1)$-budget-balanced acyclic mechanism is $\Omega(\log k)$-approximate. Recall the excludable public good cost-sharing problem (Example 2.2.12).

Theorem 7.3.15 *Every no-deficit acyclic mechanism for the public excludable good problem with n players is at least $\mathcal{H}_n - 1$-approximate.*

Proof: Fix a universe U of k players and a no-deficit acyclic mechanism $M(\chi, \tau)$. We first claim the following: for every nonempty set $S \subseteq U$, there is a player with minimum offer time $\tau(i, S)$ and cost share $\chi(i, S) \geq 1/|S|$. In proof, let $T \subset S$ denote the players with offer time strictly larger than the minimum. Since χ is no-deficit, $\sum_{i \in S \setminus T} \chi(i, S \setminus T) \geq C(S \setminus T) = 1$ and hence $\chi(i, S \setminus T) \geq 1/|S \setminus T| \geq 1/|S|$ for some player $i \in S \setminus T$. Invoking Definition 6.2.4(a) shows that $\chi(i, S) = \chi(i, S \setminus T)$ and completes the claim.

Using this claim, we can inductively rename the players of U as follows. For $i = 1, 2, \ldots, n$, player i is a player of $S_i \equiv U \setminus \{1, 2, \ldots, i-1\}$ that has minimum offer time $\tau(\cdot, S_i)$ and cost share $\chi(\cdot, S_i)$ at least $1/(n-i+1)$. Now set the valuation v_i of player i to $1/(n-i+1) - \epsilon$ for small $\epsilon > 0$. The optimal solution has efficiency $\approx \mathcal{H}_n - 1$ Since player i has minimum offer time in S_i and $v_i < \chi(i, S_i)$ for every i, the mechanism M outputs the empty allocation and has an efficiency of zero. ■

7.4 Notes

7.4.1 Is Acyclicity Automatic?

Do all primal-dual algorithms yield acyclic cost-sharing methods in the sense of Section 6.2? Goemans and Williamson [35] and Agarwal et al. [3] propose a primal-dual algorithm for a generalized version of the Steiner tree cost-sharing problem that includes Steiner forest problems. This algorithm does yield non-cross-monotonic cost shares that induce an acyclic mechanism for the Steiner tree problem. (The budget-balance and efficiency approximations achieved by the induced acyclic mechanism match, but don't improve, those achieved by the optimal Moulin mechanism for the Steiner tree problem.) However, there is no offer-function that together with these cost-shares induces an acyclic mechanism for the Steiner forest problem (See Mehta et al. [59] for details).

7.4.2 Acyclic Mechanisms and Summability

The generic methods known for deriving efficiency guarantees for Moulin mechanisms do not seem to carry over to acyclic mechanisms. In more detail, recall that a cost-sharing method χ is α-*summable* (Definition 4.2.2) for a cost function C if, for every ordering σ of the players of U and every subset $S \subseteq U$,

$$\sum_{\ell=1}^{|S|} \chi(i_\ell, S_\ell) \leq \alpha \cdot C(S) \tag{7.7}$$

where S_ℓ and i_ℓ denote the set of the first ℓ players of S and the ℓth player of S (with respect to σ), respectively. Intuitively, the ordering σ represents the reversal of the order in which players are deleted, and $\chi(i_\ell, S_\ell)$ is the worst-case valuation that player i_ℓ could have possessed, given that it was deleted from the set S_ℓ. For Moulin mechanisms, summability characterizes approximate efficiency in the following sense: if M is a Moulin mechanism based on an α-summable, no-deficit cost-sharing method, then it is $\Theta(\alpha)$-approximate (Chapter 4, Theorem 4.4.1).

Unfortunately, the summability of a cost-sharing method χ does not imply upper or lower bounds on the approximate efficiency of an acyclic mechanism constructed from χ. Summability does not automatically lead to a valid lower bound on approximate efficiency because, depending on the associated offer function, not all orderings of the players correspond to possible deletion sequences. It does not automatically give a valid upper bound because it only treats deletion sequences that result in the empty set. For cross-monotonic cost-sharing methods, worst-case deletion sequences are, essentially without loss of generality, of this form. For a non-cross-monotonic method, this need not be the case; intuitively, the presence of additional undeleted players can increase the left-hand side of (7.7).

The definition of summability can be refined to handle both of these issues, resulting in a characterization of the approximate efficiency of an acyclic mechanism. However, the resulting expression is too unwieldy to be evaluated easily for non-trivial mechanisms.

Chapter 8

Lower Bounds On Truthful Mechanisms

In previous chapters, we identify no-deficit, truthful mechanisms that have polylog approximate efficiency for a wide variety of cost-sharing problems—For submodular cost-sharing problems, metric UFL and vertex cover, we achieve a worst-case efficiency approximation of $\Theta(\log k)$. For Steiner tree cost-sharing problems, their variants and NMUFL cost-sharing problems such as set cover, we achieve a worst-case efficiency approximation of $\Theta(\log^2 k)$. The mechanisms that we identify are computationally efficient, i.e., they have polynomial time implementations. In this chapter we drop this requirement to see if we can identify no-deficit, truthful mechanisms with improved efficiency.

Section 8.1 starts by identifying a no-deficit, truthful mechanism that is $O(\log k)$-approximate for *all* cost-sharing problems with monotone cost functions. This is a very general result—Recall that all the cost-sharing problems studied in this thesis are not only monotone, but also subadditive. In particular, this mechanism improves on the best truthful, no-deficit mechanism we could identify for the Steiner tree and NMUFL cost-sharing problems, which are $\Omega(\log^2 k)$ approximate.

Given this result, the million dollar question is: *Are there truthful, no deficit mechanisms that achieve constant factor approximations of efficiency for any non-trivial cost-sharing problem family?* (Recall from Section 4.3 that maginal cost problems

admit optimally efficient mechanisms, but are trivial.)

This chapter shows that the answer to this question is an emphatic *no*, even for the combinatorially simple excludable public good cost-sharing problem (Example 2.2.12). As Figure 2.1 shows, this problem is a special case of nearly all of the cost-sharing problems that have been studied in the theoretical computer science literature, so our lower bound extends to these problem families as well.

We first investigate deterministic mechanisms that are budget-balanced and symmetric. Specifically, we show that the Shapley value mechanism (Example 4.1.4) is optimal among all deterministic, symmetric, and budget-balanced cost-sharing mechanisms for excludable public good problems. (Moulin and Shenker [63] proves only that the Shapley value mechanism is an optimal Moulin mechanism.) Here, "symmetric" means that players that submit equal bids are given the same allocations and prices. This proof is based on a new characterization of the Shapley value mechanism that improves upon a previous characterization of Deb and Razzolini [25]. See Section 8.2.

Next, we forgo a characterization based approach, and prove a far more general result. We show that *every* (γ, β)*-budget-balanced truthful mechanism is* $\Omega(\log k/\beta)$*-approximate*, where k is the number of participants. Our lower bound applies even to randomized mechanisms that are only truthful in expectation, and only (γ, β)-budget-balanced in expectation. Our lower bound is optimal up to constant factors for all $\beta = O(\sqrt{\log k})$, with the nearly matching upper bound provided by a scaled version of the Shapley value mechanism (recall Corollary 4.4.3). Our lower bounds also apply to the social cost approximation measure (Equation 3.2). See Section 8.3.

8.1 The Composed VCG-Shapley Mechanism

We now identify a no-deficit, truthful mechanism that is $\mathcal{H}_k$-approximate (recall that $\mathcal{H}_k \in \Theta(\log k)$) for every cost-sharing problem with an underlying cost function that is monotone. This mechanism satisfies no positive transfers and voluntary participation. This is a very general result; all the cost-sharing problems we study in this thesis are not only monotone, but also subadditive. The idea is to compose the VCG mechanism

with Clarke tax (Section 2.4.1) with an additional cost-sharing phase, based on the Shapley value mechanism (Example 4.1.4). The mechanism is a simplification of one proposed by Swamy (personal communication, August 2008).

Definition 8.1.1 (Composed VCG-Shapley mechanism) The allocation rule is defined as follows. Bids are collected and the optimally efficient allocation S^* is computed (If there are multiple such allocations, break ties consistently using a lexicographic ordering of 2^U). This is precisely the allocation of the VCG mechanism. The players in S^* graduate to a cost-sharing phase based on a modification of the Shapley value mechanism (recall Example 4.1.4) where all the cost-shares scaled by a factor $C(S^*)$. The composed VCG-Shapley mechanism allocates service to the players that the modified Shapley value mechanism services when run on the player set S^*. The composed VCG-Shapley mechanism charges every winning player its minimum winning bid as a function of the other players' bids.

We now prove that this mechanism is incentive compatible. The tricky part is that the set S^*, and hence the outcome of the cost-sharing phase, can conceivably by influenced by a player's bid.

Lemma 8.1.2 *The composed VCG-Shapley mechanism is truthful and satisfies voluntary participation.*

Proof: Recall Proposition 2.4.5. The mechanism is a threshold mechanism by definition, so it is sufficient to show that it is monotone.

Fix a player i and bids b_{-i} of the other players. Fix a winning bid b_i and another bid b_i', such that $b_i' > b_i$. Let S^* and $S^{*'}$ be the optimally efficient sets with bid vectors $(b_i; b_{-i})$ and $(b_i'; b_{-i})$, respectively. Because player i receives service with bid b_i, it is in S^*. The optimally efficient allocation is invariant to an increase in the bid of any winning bidder. Thus $S^* = S^{*'}$, which implies that $i \in S^{*'}$ and that $C(S^*) = C(S^{*'})$. Thus player i graduates to the cost-sharing phase and participates in same cost-sharing problem whether it bids b_i or b_i'. The Shapley value mechanism has a monotone allocation rule (this is easy to check), and so if player i wins service with bid b_i, it also wins service with bid b_i'. ■

For a fixed set of bids, suppose that S is the player set serviced by the composed VCG-Shapley mechanism and S^* is the optimally efficient set. If S is non-empty, the composed VCG-Shapley mechanism charges each winner at least $C(S^*)/|S|$ due the cost-sharing phase. So, the mechanism satisfies the no-deficit condition if the cost function is monotone. Recall that the excludable public good cost-sharing problem is also a submodular cost-sharing problem and so the upper bound from Section 5.1 shows that the efficiency loss due to the second phase is upper bounded by $\mathcal{H}_k \cdot C(S^*)$. In summary, we have the following theorem:

Theorem 8.1.3 *The composed VCG-Shapley mechanism is truthful, no-deficit, and $\mathcal{H}_k$-approximate for every cost-sharing problem with a monotone cost function.*

Remark 8.1.4 The composed VCG-Shapley mechanism can be generalized while preserving incentive compatibility: VCG can be replaced with any monotone mechanism that satisfies the following property: Fix a player i and a set of bids b_{-i} of the other players. Fix two bids b_i and b'_i for player i. Let S and S' be the set of players serviced with bids $(b_i; b_{-i})$ and $(b'_i; b_{-i})$, respectively. If $i \in S$ and $i \in S'$, then we must have that $S = S'$. The Shapley value mechanism can be replaced by any mechanism with a monotone allocation rule while preserving incentive compatibility.

In order to achieve a good efficiency guarantee and recover cost, the VCG surrogate should allocate service to a set with near optimal social welfare and the Shapley surrogate should be no-deficit while excluding as little valuation as possible. For instance, if the cost-function is subadditive, the mechanism from Bleischwitz et al. [12] is a good Shapley surrogate.

The composed VCG-Shapley mechanism is not weakly GSP unlike acyclic mechanisms:

Example 8.1.5 A *quadratic* cost-sharing problem is defined by a player set U and a non-decreasing cost function $C(S) = |S|^2$ for all $S \subseteq U$. Consider a three player instance of the quadratic cost-sharing problem. The first two players have valuations of 5 each, and the second has a valuation of 4. For this bid vector, the first two

players win service and each pay 4. However, if the third player drops its valuation to 3, the other two only pay 3 each.

Remark 8.1.6 Bleischwitz et al. [12] (recall Section 6.4) shows that there exist acyclic mechanisms that are budget-balanced and $O(\log k)$-approximate for *all* subadditive cost-sharing problems; As in the case of the composed VCG-Shapley mechanism, we do not know how to implement these mechanisms in a computationally efficient way.

8.2 A Characterization of Symmetric Mechanisms

In this section we prove a lower bound on the efficiency approximation factor of every deterministic, budget-balanced cost-sharing mechanism that satisfies the "equal treatment" property. We derive this lower bound from a new characterization of the Shapley value mechanism, discussed below.

Our characterization heavily uses Proposition 2.4.4, which states that in a truthful cost-sharing mechanism every player is offered a bid independent price. However, Proposition 2.4.4 does not specify the behavior of a truthful mechanism when a player bids exactly its threshold $t_i(b_{-i})$. There are two valid possibilities, each of which yields zero utility to a truthful player: the player is not served (at price 0), or is served and charged its bid. The following technical condition breaks ties in favor of the second outcome.

Definition 8.2.1 A mechanism satisfies *upper semi-continuity* if and only if the following condition holds for every player i and bids b_{-i} of the other players: if player i receives service at every bid larger than b_i, then it also receives service at bid b_i.

A relatively weak requirement for a cost-sharing mechanism is *consumer sovereignty*, i.e every player has a winning bid for every fixed set of bids of the other players. We stress that while our characterization result (Theorem 8.2.3) relies on upper semi-continuity and consumer sovereignty, our lower bound (Corollary 8.2.4) does not depend on them. Our results concern mechanisms satisfying the following symmetry property.

Definition 8.2.2 A mechanism satisfies *equal treatment* if and only if every two players i and j that submit the same bid receive the same allocation and price.

Recall the Shapley value mechanism from Example 4.1.4. This mechanism is truthful, budget-balanced, and $\mathcal{H}_k - 1$-approximate ($\mathcal{H}_l$ is $1 + 1/2 + 1/3 + \ldots 1/l$, the lth harmonic number, $k = |U|$). Moreover, it satisfies equal treatment and upper semi-continuity.

The Shapley value mechanism uses the same threshold function (in the sense of Proposition 2.4.4) for each player, namely:

$$\forall\ b_{-i}: \qquad t(b_{-i}) = \frac{1}{f(b_{-i}) + 1}, \tag{8.1}$$

where, $f(b_{-i})$ is the size of the largest subset S of $U \setminus \{i\}$ such that $b_j \geq 1/(|S| + 1)$ for all $j \in S$. Intuitively, this is precisely the set of other players that the Shapley value mechanism services if player i pays its share and also receives service.

Our characterization theorem is the following.

Theorem 8.2.3 *A deterministic and budget-balanced cost-sharing mechanism satisfies equal treatment, consumer sovereignty, and upper-semicontinuity if and only if it is the Shapley value mechanism.*

Proof: Fix such a mechanism M. We first note that all thresholds $t_i(b_{-i})$ induced by M must lie in $[0, 1]$: every threshold is finite by consumer sovereignty, and is at most 1 by the budget-balance condition. We proceed to show that for all players i and bids b_{-i} by the other players, the threshold function t_i has the same value as that for the Shapley value mechanism. We prove this by downward induction on the number of coordinates of b_{-i} that are equal to 1.

For the base case, fix i and suppose that b_{-i} is the all-ones vector. Suppose that $b_i = 1$. Since all thresholds are in $[0, 1]$ and M is upper semi-continuous, all players are served. By equal treatment and budget-balance, all players pay $1/k$. Thus, $t_i(b_{-i}) = 1/k$ when b_{-i} is the all-ones vector, as in the Shapley value mechanism.

For the inductive step, fix a player i and a bid vector b_{-i} that is not the all-ones vector. Set $b_i = 1$ and consider the bid vector $b = (b_i, b_{-i})$. Let S denote the set of

players j with $b_j = 1$. Let $R \supseteq S$ denote the output of the Shapley value mechanism for the bid vector b — the largest set of players such that $b_j \geq 1/|R|$ for all $j \in R$.

As in the base case, consumer sovereignty, budget-balance, and equal treatment imply that M serves all of the players of S at a common price p. For a player j outside S, b_{-j} has one more bid of 1 than b_{-i} (corresponding to player i), and the inductive hypothesis implies that its threshold is that of the Shapley value mechanism for the same bid vector b. For players of $R \setminus S$, this threshold is $1/|R|$. For a player outside R, this threshold is some value strictly greater than its bid. Since $b_j \geq 1/|R|$ for all $j \in R$ and M is upper semicontinuous, it serves precisely the set R when given the bid vector b. This generates revenue $|S|p + (|R| - |S|)/|R|$. Budget-balance dictates that the common threshold p for all players of S, and in particular the value of $t_i(b_{-i})$, equals $1/|R|$. This agrees with player i's threshold for the bids b_{-i} in the Shapley value mechanism, and the proof is complete. ■

Theorem 8.2.3 implies that the Shapley value mechanism is the optimal deterministic, budget-balanced mechanism that satisfies the equal treatment property.

Corollary 8.2.4 *Every deterministic, budget-balanced cost-sharing mechanism that satisfies equal treatment is at least $\mathcal{H}_k$-approximate.*

We briefly sketch the proof. Let M be such a mechanism. If M fails to satisfy consumer sovereignty, then we can find a player i and bids b_{-i} such that $t_i(b_{-i}) = +\infty$. Letting the valuation of player i tend to infinity shows that the mechanism fails to achieve a finite approximation factor.

Suppose that M also satisfies consumer sovereignty. Then the proof of Theorem 8.2.3 shows that the outcome of the mechanism agrees with that of the Shapley value mechanism except on the measure-zero set of bid vectors for which there is at least one bid equal to $1/i$ for some $i \in \{1, \ldots, k\}$. As in Example 4.1.4, bid vectors of the form $1 - \epsilon$, $\frac{1}{2} - \epsilon, \ldots, \frac{1}{k} - \epsilon$ for small $\epsilon > 0$ show that M is no better than $\mathcal{H}_k - 1$-approximate.

An interesting problem is to characterize the class of mechanisms obtained after dropping the (admittedly strong) equal treatment condition, and the perfect budget-balance condition. There are several mechanisms that satisfy the remaining conditions

and appear hard to characterize (e.g. [45, Example 4.1]). In the next section, forgo characterizations in order to prove more general lower bounds.

8.3 A Lower Bound on Cost-Sharing Mechanisms

In this section we prove that every $O(1)$-budget-balanced cost-sharing mechanism for the excludable public good problem is $\Omega(\log k)$-approximate. This lower bound applies even to randomized mechanisms, and even to mechanisms that are only truthful in expectation.

Theorem 8.3.1 *Every cost-sharing mechanism for the excludable public good problem that is truthful in expectation and γ, β-budget-balanced in expectation is $\Omega((\log k)/\beta)$-approximate, where k is the number of players.*

Proof: Fix values for k and $\beta \geq 1$. The plan of the proof is to define a distribution over valuation profiles such that the sum of the valuations is likely to be large but every mechanism is likely to produce the empty allocation. Let $a_1, \ldots, a_k$ be i.i.d. draws from the distribution with density $1/z^2$ on $[1, k]$ and remaining mass $(1/k)$ at zero. Set $v_i = a_i/4k\beta$ for each i and $V = \sum_{i=1}^{k} v_i$. We first note that V is likely to be $\Omega((\log k)/\beta)$. To see why, we have $\mathcal{E}[V] = k\mathcal{E}[v_i] = (\ln k)/4\beta$, $\mathbf{Var}[V] = k\mathbf{Var}[v_i] \leq k\mathcal{E}[v_i^2] = 1/(16\beta^2)$, and $\sigma[V] = 1/4\beta$. By Chebyshev's Inequality, V is at least $(\ln k - 2)/4\beta = \Omega(\log k/\beta)$ with probability at least 3/4.

Let M be a mechanism that is truthful in expectation and β-budget-balanced in expectation, meaning that for every bid vector, the expected revenue of M is at least a β fraction of its expected cost. For the excludable public good problem, the expected cost equals 1 minus the probability that no player is served. We can finish the proof by showing that the expected revenue of M, over both the random choice of valuation profile and the internal coin flips of the mechanism, is at most $1/4\beta$: if true, the expected cost of M is at most 1/4, so no player is served with probability at least 3/4. By the Union Bound, the probability that no player is served and also the sum of the valuations is $\Omega((\log k)/\beta)$ is at least 1/2. Thus, there is a valuation profile for which the optimal efficiency is $\Omega((\log k)/\beta)$ but the mechanism has an efficiency

of zero.

We next apply a transformation of Mehta and Vazirani [60], originally developed for digital goods auctions, to assist in upper bounding the revenue obtained by M. Given a bid vector b, a *randomized threshold mechanism* chooses a random threshold $t_i(b_{-i})$ for each player i (cf., Proposition 2.4.4) from a distribution that is independent of b_i. By Mehta and Vazirani [60], there is a randomized threshold mechanism M' that has the same expected revenue as M on every bid vector.

To upper bound the expected revenue of M', consider a single truthful player i with (random) valuation v_i. Every fixed threshold t extracts expected revenue $t \cdot \mathbf{Pr}[v_i \geq t] \leq 1/4k\beta$ from the player. By the Principle of Deferred Decisions, a randomized threshold that is independent of v_i also obtains expected revenue at most $1/4k\beta$ from player i. Linearity of expectation implies that the expected revenue of M', and hence of M, is at most $1/4\beta$, completing the proof. ■

Scaling the prices of the Shapley value mechanism down by a $\beta \geq 1$ factor gives a β-budget-balanced, $O(\beta+(\log k)/\beta)$-approximate mechanism (Theorem 4.4.1). Thus, the lower bound in Theorem 8.3.1 is optimal up to constant factors for all $\beta = O(\sqrt{\log k})$. Recall that there is also a trivial lower bound of β along this lines of Example 4.4.4.

8.4 Notes

8.4.1 The Power of Randomization

Dobzinski et al [27] show that randomized mechanisms are in fact strictly more powerful than deterministic ones. However the improvement is not significant. Indeed, Theorem 8.3.1 shows that the best-possible approximation guarantee of a randomized cost-sharing mechanism cannot be more than a constant factor smaller than that of the (deterministic) Shapley value mechanism.

8.4.2 Other Characterizations

Other characterizations of the Shapley value mechanism are known. See Moulin and Shenker [63] and Immorlica, Mahdian, and Mirrokni [45] for related characterizations of groupstrategyproof (Definition 2.4.2) mechanisms that satisfy various properties. Our Theorem 8.2.3 is incomparable to these results because we work with the much richer class of truthful, not necessarily groupstrategyproof, mechanisms, but assume equal treatment. Our characterization is more similar to that of Deb and Razzolini [25], who also show that the Shapley value mechanism is the only one that satisfies certain conditions. We weaken their stand-alone condition to consumer sovereignty and do not require their voluntary non-participation condition. Also, our proof is arguably simpler.

Chapter 9

Open Questions

Here are some open questions motivated by this thesis.

9.1 Better Approximation Guarantees

One natural goal is to improve upon the performance guarantees achieved by the mechanisms presented in this thesis. Some concrete suggestions follow.

- Is there a polynomial-time, β-budget-balanced acyclic mechanism for Steiner tree cost-sharing problems with $\beta < 2$ and reasonable (e.g., $O(\log^d k)$ for some constant d) approximate efficiency? Recall that such a result is achievable for $\beta = 2$ (Section 5.3, Section 7.4.1).

- Metric UFL algorithms with approximation ratio less than 1.61 are known [18, 57]. Can these be used to obtain polynomial-time acyclic mechanisms with comparable budget-balance and reasonable approximate efficiency? (Remark 7.3.7 identifies a 1.61-budget-balanced acyclic mechanism that is $O(\log k)$-approximate).

- Is there a polynomial-time, $O(1)$-budget-balanced, $o(\log^2 k)$-approximate acyclic mechanism for Steiner tree cost-sharing problems? Can achieve such bounds even more generally, i.e., for all subadditive cost-sharing problems or monotone cost-sharing problems?

Bleischwitz, Monien, and Schoppmann [12] gives acyclic mechanisms for all subadditive cost-sharing problems that are fully budget-balanced and $O(\log k)$-approximate, but do not run in polynomial time (unless $P = NP$), and do not work for all monotone cost functions.

Since acyclicity is only the means to the end of incentive-compatibility, we can ask the same questions for wider classes of mechanisms. Answer the above questions with "acyclic mechanism" replaced by "weakly groupstrategyproof mechanism" and by "strategyproof mechanism". The composed VCG-Shapley mechanism (Section 8.1) achieves the required efficiency bounds, is no-deficit and strategyproof, but does not have bounded budget-balance, is not acyclic, WGSP or polynomial time implementable.

9.2 General Demand Mechanisms

General demand cost-sharing problems should be studied in much greater depth. Bleischwitz and Schoppmann [14] generalizes Moulin mechanisms to general demand settings and applies it to generalizations of the UFL and Steiner tree cost-sharing problems where players demand redundancy in connectivity. However, the budget-balance achieved by these mechanisms scales with the maximum number of allowable service levels. Are there polynomial-time, $O(1)$-budget-balanced acyclic (alternatively weakly groupstrategyproof or strategyproof) mechanism for such problems that have reasonable economic efficiency?

The frameworks for general demand cost-sharing problems—[59] and [14]—only apply to settings where players have diminishing returns from additional levels of service. Is there a general mechanism design technique when marginal valuations can be increasing?

9.3 Characterizations

Moulin [62] provides characterizations under the assumptions of GSP and full budget-balance. Immorlica, Mahdian, and Mirrokni [45] provide a partial characterization of

GSP mechanisms without any budget-balance assumptions.

Is there a simple characterization of WGSP mechanisms? To what extent do acyclic mechanisms exhaust the class of WGSP mechanisms? (See Juarez [49] for recent progress on these questions.)

Is there a simple characterization of SP, budget-balanced mechanisms? Section 8.2 provides such a characterization, but with additional technical conditions, notably equal treatment.

Bibliography

[1] Foothills park access rights. http://www.cityofpaloalto.org/depts/csd/parks_and_open_space/preserves_and_open_spaces/foothills_park.asp.

[2] Wikipedia: Free rider problem. http://en.wikipedia.org/wiki/Free_rider_problem.

[3] A. Agrawal, P. Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[4] N. Andelman, M. Feldman, and Y. Mansour. Strong price of anarchy. *Games and Economic Behavior*, 2009. To appear.

[5] A. Archer, J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Approximation and collusion in multicast cost sharing. *Games and Economic Behavior*, 47(1):36–71, 2004.

[6] Aaron Archer and Eva Tardos. Truthful mechanisms for one-parameter agents. In *In Proc. of the 42nd IEEE Symposium on Foundations of Computer Science*, pages 482–491, 2001.

[7] S. Arora and C. Lund. Hardness of approximations. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, chapter 10, pages 399–446. PWS Publishing Company, 1997.

[8] Y. Bachrach, E. Markakis, A. D. Procaccia, J. S. Rosenschein, and A. Saberi. Approximating power indices. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 943–950, 2008.

[9] Y. Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 184–193, 1996.

[10] D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. P. Williamson. A note on the prize-collecting traveling salesman problem. *Mathematical Programming*, 59(3):413–420, 1993.

[11] Y. Bleischwitz and B. Monien. Fair cost-sharing methods for scheduling jobs on parallel machines. In *Proceedings of the 6th Italian Conference on Algorithms and Complexity (CIAC)*, volume 3998 of *Lecture Notes in Computer Science*, pages 175–186, 2006.

[12] Y. Bleischwitz, B. Monien, and F. Schoppmann. To be or not to be (served). In *Proceedings of the Third Annual International Workshop on Internet and Network Economics (WINE)*, volume 4858 of *Lecture Notes in Computer Science*, pages 515–528, 2007.

[13] 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. Extended version available at http://wwwcs.upb.de/cs/agmonien/PERSONAL/FSCHOPP/pdf/LexicographicMaximization_extended.pdf.

[14] Yvonne Bleischwitz and Florian Schoppmann. Group-strategyproof cost sharing for metric fault tolerant facility location. In Burkhard Monien and Ulf-Peter

Schroeder, editors, *Proceedings of the 1st International Symposium on Algorithmic Game Theory (SAGT'08)*, volume 4997 of *LNCS*, pages 350–361, 2008.

[15] Yvonne Bleischwitz and Florian Schoppmann. New efficiency results for makespan cost sharing. *Information Processing Letters*, 107(2):64–70, July 2008.

[16] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[17] J. Brenner and G. Schäfer. Cost sharing methods for makespan and completion time scheduling. In *Proceedings of the 24th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 4393 of *Lecture Notes in Computer Science*, pages 670–681, 2007.

[18] J. Byrka. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. In *Proceedings of the 10th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, volume 4627 of *Lecture Notes in Computer Science*, pages 29–43, 2007.

[19] S. Chawla, T. Roughgarden, and M. Sundararajan. Optimal cost-sharing mechanisms for network design. In *Proceedings of the Second Annual International Workshop on Internet and Network Economics (WINE)*, volume 4286 of *Lecture Notes in Computer Science*, pages 112–123, 2006.

[20] V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[21] V. Chvátal. *Linear Programming*. Freeman, 1983.

[22] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971.

[23] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing (STOC)*, pages 151–158, 1971.

[24] C. D'Aspremont and L.-A. Gerard-Varet. Incentives and incomplete information. *Journal of Public Economics*, 11:25–45, 1979.

[25] R. Deb and L. Razzolini. Voluntary cost sharing for an excludable public project. *Mathematical Social Sciences*, 37:123–138, 1999.

[26] N. R. Devanur, M. Mihail, and V. V. Vazirani. Strategyproof cost-sharing mechanisms for set cover and facility location games. *Decision Support Systems*, 39(1):11–22, 2005.

[27] S. Dobzinski, A. Mehta, T. Roughgarden, and M. Sundararajan. Is Shapley cost-sharing optimal? In *Proceedings of the First International Symposium on Algorithmic Game Theory (SAGT)*, volume 4997 of *Lecture Notes in Computer Science*, pages 327–336, 2008.

[28] Bhaskar Dutta and Debraj Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57(3):615–35, May 1989.

[29] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards, Series B*, 71(4):233–240, 1967.

[30] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the 35th Annual ACM Symposium on the Theory of Computing (STOC)*, 2003.

[31] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[32] J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Hardness results for multicast cost sharing. *Theoretical Computer Science*, 304(1-3):215–236, 2003.

[33] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, 2001.

[34] J. Feigenbaum and S. J. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International*

Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, pages 1–13, 2002.

[35] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[36] Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[37] A. Gupta, J. Könemann, S. Leonardi, R. Ravi, and G. Schäfer. An efficient cost-sharing mechanism for the prize-collecting steiner forest problem. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1153–1162, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[38] A. Gupta, M. Pál, R. Ravi, and A. Sinha. Boosted sampling: Approximation algorithms for stochastic optimization. In *Proceedings of the 36th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 417–426, 2004.

[39] A. Gupta, A. Srinivasan, and É. Tardos. Cost-sharing mechanisms for network design. *Algorithmica*, 50(1):98–119, 2008.

[40] Anupam Gupta, Amit Kumar, Martin P´al, and Tim Roughgarden. Approximation via cost sharing: Simpler and better approximation algorithms for network design. *J. ACM*, 54(3):11, 2007.

[41] S. Hart and A. Mas-Colell. Potential, value, and consistency. *Econometrica*, 57(3):589–614, 1989.

[42] J. D. Hartline. *Optimization in the Private Value Model: Competitive Analysis Applied to Auction Design*. PhD thesis, University of Washington, 2003.

[43] D. S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22(2):148–162, 1982.

[44] M. Imase and B. M. Waxman. Dynamic Steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

[45] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. *ACM Transactions on Algorithms*, 4, 2008. Article 24.

[46] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *Journal of the ACM*, 60(6):795–824, 2003.

[47] K. Jain and V. Vazirani. Applications of approximation algorithms to cooperative games. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing (STOC)*, pages 364–372, 2001.

[48] K. Jain and V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. *SIAM Journal on Computing*, 38(1):241–256, 2008.

[49] R. Juarez. Group strategyproof cost sharing: the role of indifferences. Working paper, 2007.

[50] D. R. Karger and M. Minkoff. Building Steiner trees with incomplete global knowledge. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 613–623, 2000.

[51] K. Kent and D. Skorin-Kapov. Population monotonic cost allocation on MST's. In *Operational Research Proceedings KOI*, pages 43–48, 1996.

[52] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 767–775, 2002.

[53] S. Khot and O. Regev. Vertex cover might be hard to approximate to within $(2-\epsilon)$. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity*, pages 379–386, 2003.

[54] J. Kleinberg and É. Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *Journal of the ACM*, 49(5):616–639, 2002.

[55] J. Könemann, S. Leonardi, G. Schäfer, and S. van Zwam. A group-strategyproof mechanism for the Steiner forest game. *SIAM Journal on Computing*, 37(5):1319–1341, 2008.

[56] S. Leonardi and G. Schäfer. Cross-monotonic cost-sharing methods for connected facility location. *Theoretical Computer Science*, 326(1-3):431–442, 2004.

[57] M. Mahdian, Y. Ye, and J. Zhang. Approximation algorithms for metric facility location problems. *SIAM Journal on Computing*, 36(2):411–432, 2006.

[58] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[59] A. Mehta, T. Roughgarden, and M. Sundararajan. Beyond Moulin mechanisms. *Games and Economic Behavior*, 2009. To appear.

[60] A. Mehta and V. V. Vazirani. Randomized truthful auctions of digital goods are randomizations over truthful auctions. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC)*, pages 120–124, 2004.

[61] H. Moulin. *Cooperative microeconomics : a game-theoretic introduction*. Prentice Hall, 1995.

[62] H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999.

[63] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: Budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001.

[64] S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2):117–236, 2005.

[65] R. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[66] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. The MIT Press, July 1994.

[67] Young H. P., N. Okada, and T. Hashimoto. Cost allocation in water resources development. *Water Resources. Research*, 18(3):463–475, 1982.

[68] M. Pál and É. Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proceedings of the 44th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 584–593, 2003.

[69] B. Peleg. Axiomatizations of the core. In R. J. Aumann and S. Hart, editors, *Handbook of Game Theory*, volume 1, chapter 13, pages 397–412. North-Holland, 1992.

[70] T. Roughgarden and M. Sundararajan. New trade-offs in cost-sharing mechanisms. In *Proceedings of the 38th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 79–88, 2006.

[71] T. Roughgarden and M. Sundararajan. Optimal efficiency guarantees for network design mechanisms. In *Proceedings of the 12th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, volume 4513 of *Lecture Notes in Computer Science*, pages 469–483, 2007.

[72] T. Roughgarden and M. Sundararajan. Quantifying inefficiency in cost-sharing mechanisms. *Journal of of the ACM*, 2009. To appear.

[73] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[74] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.