Beyond Moulin Mechanisms*

Aranyak Mehta† Tim Roughgarden‡ Mukund Sundararajan§

January 26, 2008

Abstract

The only known general technique for designing truthful and approximately budget-balanced cost-sharing mechanisms with good efficiency or computational complexity properties is due to Moulin (1999). This framework has been successfully applied to a wide range of problems. For many fundamental cost-sharing applications, however, Moulin mechanisms provably suffer from poor budget-balance, poor economic efficiency, or both.

We propose *acyclic mechanisms*, a new framework for designing truthful and approximately budget-balanced cost-sharing mechanisms. Acyclic mechanisms strictly generalize Moulin mechanisms and offer three important advantages. First, it is easier to design acyclic mechanisms than Moulin mechanisms: many classical primal-dual algorithms naturally induce a non-Moulin acyclic mechanism with good performance guarantees. Second, for important classes of cost-sharing problems, acyclic mechanisms have exponentially better budget-balance and economic efficiency than Moulin mechanisms. Finally, while Moulin mechanisms have found application primarily in binary demand games, we extend acyclic mechanisms to general demand games, a multi-parameter setting in which each bidder can be allocated one of several levels of service.

*A preliminary version of this paper appeared in the Proceedings of the 8th ACM Conference on Electronic Commerce, June 2007.

†Google, Inc., Mountain View, CA. This work was done while the author was at IBM Almaden Research Center, San Jose, CA. Email: aranyak@google.com.

‡Department of Computer Science, Stanford University, 462 Gates Building, 353 Serra Mall, Stanford, CA 94305. Supported in part by NSF CAREER Award CCF-0448664, an ONR Young Investigator Award, and an Alfred P. Sloan Fellowship. Email: tim@cs.stanford.edu.

§Department of Computer Science, Stanford University, 470 Gates Building, 353 Serra Mall, Stanford, CA 94305. Supported by NSF Award CCF-0448664 and a Stanford Graduate Fellowship. Email: mukunds@cs.stanford.edu.

1 Introduction

Designing truthful mechanisms with non-trivial worst-case revenue guarantees is an important but challenging problem. Such guarantees are especially crucial in cost-sharing problems, where the mechanism incurs outcome-dependent costs, such as production costs. *(Approximate) budget-balance* is the natural constraint that the revenue collected by the mechanism (approximately) equals the cost incurred.

General mechanism design techniques are highly coveted but equally rare. The most powerful technique is, of course, the VCG mechanism, which is truthful and economically efficient (i.e., welfare-maximizing). However, the VCG mechanism typically offers no non-trivial revenue guarantees (see e.g. [35]). More broadly, many important cost functions are complex, and can even be NP-hard to evaluate; for these, no "reasonable" truthful mechanism can achieve exact budget-balance [15, 21, 23].

The only known general technique for designing truthful, approximately budget-balanced cost-sharing mechanisms that have reasonable economic efficiency or computational complexity is due to Moulin [34]. Roughly, a Moulin mechanism simulates an ascending iterative auction. In each iteration, prices are simultaneously offered to the remaining players. Players that accept remain in contention; the others are removed. The mechanism halts when all remaining players accept the prices offered to them. To achieve approximate budget-balance, the mechanism offers prices at each iteration that approximately cover the cost that would be incurred if the iteration is the last. To obtain truthfulness, a Moulin mechanism offers each player a non-decreasing sequence of prices. Thus, Moulin mechanisms are flexible, intuitive, and provide explicit control over the generated revenue. Moreover, they have been successfully designed for a wide range of applications.

Why aren't Moulin mechanisms enough? There are three reasons. First, recent negative results [21, 40] show that for many fundamental cost-sharing problems, Moulin mechanisms inevitably suffer from poor budget-balance, poor economic efficiency, or both. Second, designing the ascending prices that are offered in each iteration of a Moulin mechanism can be a highly non-trivial problem. For example, for metric uncapacitated facility location (UFL) cost-sharing problems, many classical approximation algorithms (e.g. [7, 17, 22, 24, 42]) naturally induce prices, but such prices need not be ascending and thus do not lead to truthful Moulin mechanisms; instead, a new metric UFL algorithm was devised for this purpose [36]. Third, Moulin mechanisms have found application primarily in "binary demand games", in which each player is either served by the mechanism or not (see also [34]), and not in richer multi-parameter problems. These drawbacks motivate the search for mechanism design techniques that go beyond Moulin mechanisms, while retaining their desirable features.

1.1 Our Results

We propose *acyclic mechanisms*, a new framework for designing truthful, approximately budget-balanced cost-sharing mechanisms. Acyclic mechanisms strictly generalize Moulin mechanisms, retain nearly all of their laudable properties, and address the three drawbacks discussed above.

To describe the difference between Moulin and acyclic mechanisms, recall that a Moulin mechanism simulates an ascending auction in which prices are simultaneously offered to the remaining players in each iteration. In an iteration of an acyclic mechanism, these prices are offered to the players *one-by-one* in some designer-prescribed order. If a player refuses the price offered to it, the iteration terminates immediately, this player is removed for the rest of the auction, and the next iteration begins anew with all of the remaining players. See Section 3 for a formal definition. Perhaps surprisingly, this minor modification greatly enriches the set of possible truthful mechanisms.

Problem	Moulin lower bounds	Acyclic upper bounds
Vertex Cover	$\alpha,\ \beta = \Omega(k^{1/3})$	$\alpha = O(\log k),\ \beta = 2$
Set Cover	$\alpha,\ \beta = \Omega(\sqrt{k})$	$\alpha,\ \beta = O(\log k)$
Metric UFL	$\alpha = \Omega(\log k),\ \beta = 3$	$\alpha = O(\log k),\ \beta = 1.61$
Steiner Tree	$\alpha = \Omega(\log^2 k),\ \beta = 2$	$\alpha = O(\log^2 k),\ \beta = 2$
Fault-Tolerant UFL	N/A	$\alpha = O(R_{max}^2 + \log k),\ \beta = O(R_{max}^2)$

Table 1: Summary of approximation results. "Moulin lower bounds" are provable lower bounds, established in [21, 40, 41], on the best-possible worst-case approximate efficiency and budget-balance achievable by Moulin mechanisms for the given problem class, where k denotes the number of players. "Acyclic upper bounds" are the performance guarantees for the acyclic mechanisms designed and analyzed in this paper. For fault-tolerant UFL, R_{max} denotes the maximum number of facilities to which a demand might be connected (Section 7).

The reason is that many natural methods of charging prices do not give rise to ascending auctions when prices are offered simultaneously, but *do* yield ascending auctions when some of the offers are suppressed by the early termination of an iteration.

Acyclic mechanisms offer three important advantages over Moulin mechanisms. First, we show in Section 4 that several truthful acyclic mechanisms follow in a generic way from "off-the-shelf" primal-dual algorithms. For example, all known primal-dual and dual fitting algorithms for metric UFL [22, 24] and Steiner tree [1, 16] naturally induce truthful acyclic mechanisms.

Second, for several important classes of cost-sharing problems, acyclic mechanisms have far better budget-balance and economic efficiency than Moulin mechanisms. We make this comparison precise using the standard notions of *α-approximate efficiency* and *β-budget-balance*, defined formally in Section 2. The approximation ratios α, β are always at least one, with $\alpha = 1$ and $\beta = 1$ denoting full efficiency and exact budget-balance, respectively. For some cost-sharing problems, Moulin mechanisms cannot obtain good approximate efficiency or budget-balance. For example, for Vertex Cover cost-sharing problems, every Moulin mechanism possesses $\Omega(k^{1/3})$-approximate budget-balance and economic efficiency, where k is the number of players [21, 40]. In sharp contrast, we show that the well-known primal-dual Vertex Cover approximation algorithm induces a truthful acyclic mechanism that is 2-budget-balanced and has $O(\log k)$-approximate efficiency. We also give quantitative improvements over the best-possible Moulin mechanisms for several other types of cost-sharing problems. Table 1 summarizes our results. (We use $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, and $f(n) = o(g(n))$ to mean 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.)

Finally, in Section 6 we extend acyclic mechanisms to general demand cost-sharing problems, a multi-parameter setting in which each bidder can be allocated one of multiple levels of service. As a paradigmatic example, in Section 7 we focus on a fault-tolerant version of UFL cost-sharing problems.

What do we sacrifice for the increased generality of acyclic mechanisms? Only a modicum of collusion-resistance: while Moulin mechanisms are *groupstrategyproof*—without side payments, every non-trivial deviation by a coalition harms one of its members—acyclic mechanisms are *weakly groupstrategyproof*, meaning that every non-trivial deviation by a coalition fails to help one of its members.

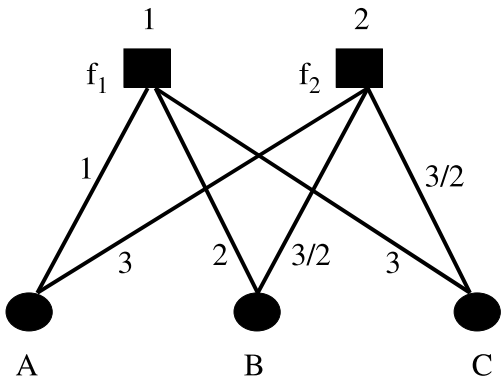


Figure 1: Example 2.1. An instance of uncapacitated facility location (UFL).

1.2 Related Work

The theory of Moulin mechanisms was developed by Moulin [34] and Moulin and Shenker [35]. Subsequently, researchers designed approximately budget-balanced Moulin mechanisms for a wide range of combinatorial cost-sharing problems [4, 18, 19, 23, 25, 30, 31, 36, 41]. Immorlica, Mahdian, and Mirrokni [21] were the first to prove that, for several basic classes of cost-sharing problems, Moulin mechanisms inevitably suffer from poor budget-balance. Roughgarden and Sundararajan [40] developed a framework for quantifying efficiency loss in Moulin mechanisms, and noted that Moulin mechanisms with poor budget-balance have equally poor economic efficiency. Finally, Devanur, Mihail, and Vazirani [10] designed several approximately budget-balanced cost-sharing mechanisms that are not Moulin mechanisms. All of the mechanisms in [10] are equivalent to instantiations of the acyclic mechanism framework developed in this paper.

All of our mechanisms are derived from primal-dual algorithms. There is, of course, a long history of connections between cost sharing, mechanism design, and the primal-dual method (see e.g. [10, 23, 25, 36, 37] and the references therein). Our work further strengthens these connections.

2 Preliminaries

2.1 Cost-Sharing Problems and Mechanisms

A *cost-sharing problem* is specified by a cost function C defined on a universe U of players. Every player $i \in U$ has a private, nonnegative *valuation* v_i for service. We assume that the cost function C is non-decreasing (i.e., $S \subseteq T$ implies $C(S) \leq C(T)$) and that $C(\emptyset) = 0$. The cost functions that we consider are implicitly defined by combinatorial optimization problems; here, $C(S)$ denotes the value of a minimum-cost solution to the subproblem induced by the subset S of players.

Example 2.1 (UFL) An important type of implicitly defined cost function is an *uncapacitated facility location (UFL)* cost function. Such a function is specified by a set U of demands (the players), 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 1, for example, the universe contains three players and two facilities, the facility opening costs are $f_1 = 1$ 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 1, the cost $C(\{A,B,C\})$ of servicing all of the players is 7.

We focus on direct-revelation mechanisms. Such a mechanism collects a 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. We only allow mechanisms that satisfy the following standard assumptions: *individual rationality*, meaning that $p_i = 0$ if $i \notin S$ and $p_i \leq b_i$ if $i \in S$; and *no positive transfers*, meaning that prices are always nonnegative. We also assume that players have quasilinear utilities, meaning that each player i aims to maximize $u_i(S, p_i) = v_i x_i - p_i$, where $x_i = 1$ if $i \in S$ and $x_i = 0$ if $i \notin S$. (Strictly speaking, we should use the notation $u_i(v_i, S, p_i)$; but the shorter form is convenient and should cause no confusion.)

A mechanism is *strategyproof (SP)*, or *truthful*, if no player can ever strictly increase its utility by misreporting its valuation. Formally, SP means that 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. (The player has true valuation v_i in both cases.) It is *groupstrategyproof (GSP)* [35] 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. It is *weakly groupstrategyproof (WGSP)* [10] 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. We note that while WGSP implies SP, truthful mechanisms are not generally WGSP; for example, VCG mechanisms are typically not WGSP.

Traditionally, the role of a cost-sharing mechanism is to simply select an allocation (such as a subset of players to serve in a UFL cost-sharing problem). The recent computer science literature has, for the most part, additionally demanded cost-sharing mechanisms that produce a feasible way of supplying the chosen allocation—in a UFL problem, a proposal of facilities to open and connections between open facilities and the served players. This paper follows the latter approach: all of the mechanisms we consider also produce a feasible solution to the optimization problem induced by the served set S. The cost $C_M(S)$ of this feasible solution is permitted to exceed the optimal cost $C(S)$. For instance, in Example 2.1, a mechanism M might choose to service the set $\{A,B,C\}$ by opening the second facility, connecting all three players to it, and incurring (suboptimal) cost $C_M(\{A,B,C\}) = 8$. Allowing suboptimal solutions is necessary for feasibly implementable (i.e., polynomial-time) mechanisms: all of the optimization problems we consider are NP-hard, and thus computing the optimal cost $C(S)$ given a set S cannot be accomplished in polynomial time, unless $P = NP$.

2.2 Approximate Budget-Balance and Efficiency

We study two types of approximation guarantees for cost-sharing mechanisms, one for revenue and one for economic efficiency. First, for a parameter $\beta \geq 1$, a mechanism M is *β-budget balanced* if

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

for every outcome (set S, prices p, and feasible solution with cost $C_M(S)$) of the mechanism. We emphasize that the revenue of the mechanism must (approximately) cover the cost of the solution it proposes, which can exceed the optimal cost $C(S)$, and should also be no more than the optimal cost. This requirement can only be met if the feasible solution produced by the mechanism has cost at most β times that of optimal. (Alternatively, we could require that $C_M(S) \leq \sum_{i \in S} p_i \leq \beta \cdot C(S)$. A mechanism satisfying one of these definitions is easily modified to satisfy the other simply by scaling its prices by a β factor. All of our results have natural analogues for this alternative definition of approximate budget-balance.)

Second, following [40], we quantify efficiency loss in cost-sharing mechanisms via the *social cost* objective. The social cost incurred by a mechanism is defined as the cost $C_M(S)$ of the feasible solution it produces for the instance corresponding to S, plus the sum $\sum_{i \notin S} v_i$ of the *excluded* valuations. The optimal social cost is

$$\min_{S \subseteq U} \left[C(S) + \sum_{i \notin S} v_i \right]. \tag{1}$$

A cost-sharing mechanism has two sources of inefficiency: first, it might choose a suboptimal set S of players to serve; second, it might produce a suboptimal solution to the optimization problem induced by S.

Minimizing social cost is ordinally equivalent to maximizing social surplus $\sum_{i \in S} v_i - C(S)$. No meaningful approximation results are possible for the latter objective [14]. The weaker goal of approximating the optimal social cost permits the rigorous differentiation between cost-sharing mechanisms on efficiency grounds, and the social cost objective can be interpreted as the "minimal perturbation" of surplus necessary for non-trivial approximation results (see [40] for a formal argument). We call a cost-sharing mechanism *α-approximate* if, assuming truthful bids, it always produces a solution with social cost at most α times that of an optimal solution.

2.3 Cost-Sharing Methods and Moulin Mechanisms

A *Moulin mechanism* is a type of cost-sharing mechanism that is driven by a *cost-sharing method*—a function χ that assigns a non-negative *cost share* $\chi(i, S)$ for every subset $S \subseteq U$ of players and every player $i \in S$. We consider cost-sharing methods that, given a set S, produce both the cost shares $\chi(i, S)$ for all $i \in S$ and also a feasible solution for the optimization problem induced by S. A cost-sharing method is *β-budget balanced* for a cost function C and a parameter $\beta \geq 1$ if

$$\frac{C_\chi(S)}{\beta} \leq \sum_{i \in S} \chi(i, S) \leq C(S), \tag{2}$$

where $C_\chi(S)$ is the cost of the feasible solution produced by the method χ. This cost can exceed the optimal cost $C(S)$, and depends on the cost-sharing method χ. 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 $i \in S$, $\chi(i, S) \geq \chi(i, T)$.

Given a cost-sharing method χ for 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 2.2 Let U be a universe of players and χ a 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 Moulin mechanism $M(\chi)$ clearly inherits the budget-balance factor of the cost-sharing method χ. Moulin [34] proved that for every cross-monotonic cost-sharing method χ, the corresponding mechanism $M(\chi)$ is GSP.

3 Acyclic Mechanisms

3.1 Overview

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.

3.2 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. As suggested in Section 3.1, a cost-sharing method and an offer function induce a mechanism that simulates an iterative auction in a natural, generic way.

Definition 3.1 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 3.2 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 3.3 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 Section 3.1, 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 3.3.

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 3.4 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 3.4, 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 3.5 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 3.6 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 3.7 Acyclic mechanisms are strictly more general than Moulin mechanisms. For example, all sequential mechanisms (see [34]), 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 3.4 is easy to satisfy in several applications. Looking ahead, Section 4 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 2.1), there is a one-to-one correspondence between players and dual variables. We employ cost-sharing methods that define cost shares as the dual variable values computed by a primal-dual UFL algorithm that runs over time. The offer time of a player is defined as the time at which the player's dual variable first assumes its final value.

Remark 3.8 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 3.4 implies that for every subset S of players, the following graph is directed 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 3.4 is reminiscent of but different from the notion of "semi-cross-monotonicity" introduced in [21].

3.3 Properties of Acyclic Mechanisms

The following basic properties of acyclic mechanisms are immediate.

Proposition 3.9 *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 $M(\chi, \tau)$ has no positive transfers and is individually rational.

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 3.10 *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 3.4(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, $\chi(\ell, S \backslash T) > b_\ell$. 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 3.4(a) then gives $\chi(\ell, S) = \chi(\ell, S \setminus T)$, a contradiction. ■

Corollary 3.11 *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 3.10 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 3.12 *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 3.11 implies that no player of $L(i, S)$ will be deleted in this or subsequent iterations. The lemma now follows from Definition 3.4(b). ■

Lemma 3.12 implies that acyclic mechanisms are strategyproof.

Theorem 3.13 *Every acyclic mechanism is strategyproof.*

Since we generalize Theorem 3.13 in Theorem 3.16 below, we omit its short proof.

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

Example 3.14 Define U, χ, and τ_y as in Example 3.5. 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 Section 2.3 that Moulin mechanisms are groupstrategyproof (GSP). The next example shows that acyclic mechanisms need not be GSP.

Example 3.15 Define U, χ, and τ_x as in Example 3.5. 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 2.1), and thus nearly match the incentive-compatibility guarantee of Moulin mechanisms.

Theorem 3.16 *Every acyclic mechanism is WGSP.*

Proof: Let $M(\chi, \tau)$ be an acyclic mechanism defined on the universe U. Recall from Section 2.1 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 3.9(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 3.12, $p_i' \geq p_i^* > v_i$. Thus $u_i(S', p') < 0 = u_i(S, p)$, completing the proof. ■

Remark 3.17 The proof of Theorem 3.16 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 3.15).

Remark 3.18 Not all WGSP mechanisms are acyclic; see Juarez [27]. 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.

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

4 Acyclic Mechanisms via Primal-Dual Algorithms

This section demonstrates how several well-known primal-dual algorithms naturally induce acyclic cost-sharing 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 5 proves that these mechanisms match or, in most cases, improve upon the best approximation guarantees possible for Moulin mechanisms.

Section 4.1 formally defines the five types of combinatorial cost-sharing problems that we study. Section 4.2 gives a self-contained account of three primal-dual algorithms, shows how each induces a cost-sharing method and an offer function in a natural way, and proves that these cost-sharing methods are not cross-monotonic. Section 4.3 proves the acyclicity of these mechanisms, and also shows that not all natural primal-dual algorithms induce acyclic mechanisms.

4.1 Five Combinatorial Cost-Sharing Problems

This section and the next focus on the following five classes of cost-sharing problems.

Non-Metric Uncapacitated Facility Location (NMUFL). As introduced in Example 2.1, a *non-metric uncapacitated facility location (NMUFL)* cost function C is described by a universe U of demands, a set F of facilities with nonnegative opening costs, and connection cost function c defined on $F \times U$. Infinite connection costs are also allowed. For a subset $S \subseteq U$, $C(S)$ 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.

Set Cover. A *set cover* cost function C is described by a universe U of elements and 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 $\mathcal{C}$.

There is a close connection between NMUFL and Set Cover problems, and the latter can be viewed as special cases of the former: 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).

Vertex Cover. A *vertex cover* cost function C is described 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 of a vertex cover—a subset of vertices that includes at least one endpoint of each edge—of (V, S).

Vertex cover cost functions are clearly special cases of set cover cost functions: edges correspond to elements, and sets of edges incident on a common vertex form the subsets.

Metric Uncapacitated Facility Location. A *metric uncapacitated facility location* cost function is a NMUFL cost function 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,

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

Steiner Tree (ST). A *Steiner tree (ST)* cost function is specified via an undirected graph $G = (V, E)$ with nonnegative edge costs, a root vertex $r \in V$, and a subset $U \subseteq V$ of source vertices. For a subset $S \subseteq U$, $C(S)$ is defined as the minimum cost of a subgraph of G that contains at least one path between the root r and each source of S.

4.2 Primal-Dual Algorithms and Cost-Sharing Methods

Good 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 (e.g. [18, 23, 25, 30, 36]), and we demonstrate that this connection is equally powerful in the design of acyclic mechanisms.

This section describes three non-cross-monotonic cost-sharing methods induced by well-known primal-dual algorithms, including two incomparable methods for NMUFL problems and a method for ST problems. Sections 4.3–5.2 leverage these methods to design acyclic mechanisms with good performance guarantees, and in particular establish most of the upper bounds listed in Table 1.

4.2.1 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 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 $\mathcal{C}(S)$ 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:

$$
\begin{array}{lll}
\text{Min} & \displaystyle\sum_{(q,T) \in \mathcal{C}(S)} c(q,T) x_{qT} & \\
\text{subject to:} & & \\
(IP(S)) & \displaystyle\sum_{(q,T) \in \mathcal{C}(S) \,:\, i \in T} x_{qT} \geq 1 & \text{for all } i \in S \\
& x_{qT} \in \{0,1\} & \text{for all } (q,T) \in \mathcal{C}(S).
\end{array}
$$

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

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

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. [9]), 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 $IP(S)$:

$$\sum_{i \in S} \alpha_i \leq \sum_{(q,T) \in \mathcal{C}(S)} c(q,T) x_{qT}. \tag{3}$$

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

$$\sum_{(q,T) \in \mathcal{C}(S)} c(q,T) x^*_{qT} \leq \beta \cdot \sum_{i \in S} \alpha^*_i. \tag{4}$$

Interpret x^* as a feasible solution to the NMUFL instance induced by S, and each dual variable α^*_i as a cost share $\chi(i, S)$. By inequalities (3) and (4), 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 (3) and (4).

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 2: The PD algorithm for NMUFL.

There are several broadly applicable algorithmic paradigms for designing approximation algorithms of this type (see e.g. [43]).

Cost-sharing methods do not automatically yield truthful cost-sharing mechanisms unless they satisfy additional constraints (cf., Definition 3.4). 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 (3) and (4) 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.

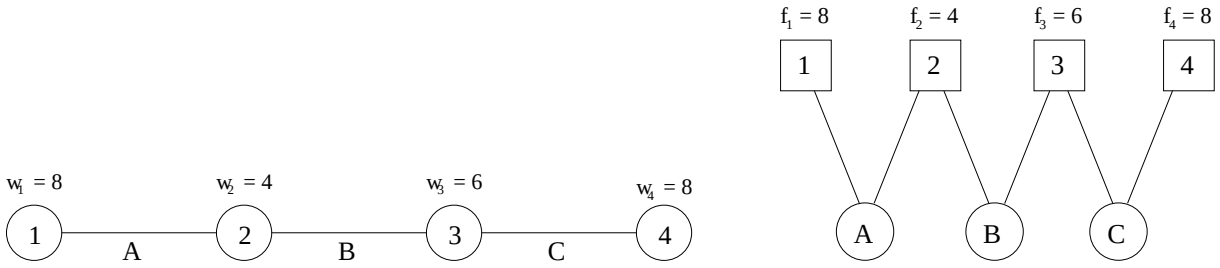
Figure 2 displays a primal-dual algorithm for the NMUFL problem, which we call the *PD algorithm*. (This algorithm is well known; see [20] and [43, Chapter 15].) At the beginning of the algorithm, all dual variables are zero and all stars are unchosen. The algorithm also maintains a notion of time, initially 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 [22]). 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 4.1 (PD Invariant) *For every NMUFL instance, the PD algorithm terminates with a dual feasible solution.*

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.

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 4.2 Consider the Vertex Cover cost-sharing problem shown in Figure 3(a), with vertex weights as shown. This problem corresponds to the NMUFL instance shown in Figure 3(b). Edges in the figure represent zero connection costs; non-edges represent infinite connection costs.



(a) Vertex Cover problem

(b) Equivalent NMUFL problem

Figure 3: Example 4.2. The cost-sharing method χ_{PD} is not cross-monotonic.

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 2 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 $\chi_{PD}(C, \{A, B, C\}) = 4$. 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 $\chi_{PD}(C, \{B, C\}) = 3 < \chi_{PD}(C, \{A, B, C\})$.

The primal-dual algorithm in Figure 2 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 call the mechanism $M(\chi_{PD}, \tau_{PD})$ induced by χ_{PD} and τ_{PD} the *PD mechanism*. Sections 4.3–5.2 establish that the PD mechanism is acyclic and, for Vertex Cover problems, is 2-budget-balanced and $O(\log k)$-approximate.

Example 4.3 In Example 4.2, χ_{PD} fails to be cross-monotonic because

$$\chi_{PD}(C, \{B, C\}) = 3 < 4 = \chi_{PD}(C, \{A, B, C\}).$$

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 2 before it offers player C its first-round price of 4. 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.

4.2.2 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 4). See also Remark 4.8 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 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, prior to the scaling in Step 3, the current primal and dual solutions have equal objective function value. Primal-dual algorithms

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 $x_{qT \cup W} = 1$.

3. Divide every dual variable α_i by $\mathcal{H}_k$, where $\mathcal{H}_k = \sum_{i=1}^{k} 1/i$ and $k = |U|$.

Figure 4: The DF algorithm for NMUFL.

of this type are sometimes called *dual-fitting algorithms* [22], so we call this algorithm the *DF algorithm*.

Lemma 4.4 (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 [20, 22] 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. This motivates Step 3, which scales the dual variables to recover dual feasibility.

Lemma 4.5 *For every NMUFL instance, the DF algorithm terminates with a dual feasible solution.*

Lemma 4.5 follows from the well-known dual-fitting analysis of the greedy Set Cover algorithm (see [8, 20] and [43, Chapter 13]).

Remark 4.6 Lemma 4.5 holds with a scaling factor of $\mathcal{H}_{|S|}$. For incentive-compatibility reasons (Section 4.3.1), we scale by the larger factor of $\mathcal{H}_{|U|}$ in Step 3.

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 final (scaled) 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 [10]. Sections 4.3–5.2 prove acyclicity of and good performance guarantees for the DMV mechanism.

Remark 4.7 In Example 4.2, $\chi_{DF}(C, \{A, B, C\}) = 6/\mathcal{H}_3 = 36/11$ while $\chi_{DF}(C, \{B, C\}) = 3/\mathcal{H}_3 = 18/11$. 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 4.8 The DF algorithm can also be interpreted as a greedy algorithm [22]. 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, divided by $\mathcal{H}_k$.

Remark 4.9 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 [10] are described in this way.

4.2.3 The AKR-GW Mechanism for ST Problems

Our final cost-sharing method is an analogue of the PD method for ST cost-sharing problems. To define this ST cost-sharing method, we introduce a well-known primal-dual formulation for the ST problem. Consider a graph $G = (V, E)$ with nonnegative edge costs c, a root vertex r, and source vertices $U = \{s_1, \ldots, s_k\}$ (the players). For a subset $A \subseteq V$ of vertices, let $\delta(A)$ denote the edges of E with precisely one endpoint in A. For a subset $S \subseteq U$ of players, a subset $A \subseteq V \setminus \{r\}$ is *S-separating* if it contains at least one vertex of S. Let $\mathcal{C}(S)$ denote the S-separating subsets. The following integer program is an exact formulation of the ST problem induced by a subset $S \subseteq U$ of players:

$$
\begin{aligned}
& & \text{Min} \quad & \sum_{e \in E} c_e x_e & & \\
& & \text{subject to:} & & & \\
& (IP - ST(S)) & & \sum_{e \in \delta(A)} x_e \geq 1 & & \text{for all } A \in \mathcal{C}(S) \\
& & & x_e \in \{0, 1\} & & \text{for all } e \in E.
\end{aligned}
$$

The decision variables of $IP(S)$ indicate which edges are chosen. The constraints require that for every subset A that includes at least one source of S and excludes the root, at least one chosen edge protrudes from A. Every subgraph of G that spans $S \cup \{r\}$ satisfies these constraints, and conversely.

We can obtain a linear program from $IP - ST(S)$ by replacing the final set of constraints by the nonnegativity constraints $x_e \geq 0$ for all $e \in E$. The dual linear program is

$$
\begin{aligned}
& & \text{Max} \quad & \sum_{A \in \mathcal{C}(S)} y_A & & \\
& & \text{subject to:} & & & \\
& (D - ST(S)) & & \sum_{A \in \mathcal{C}(S) \,:\, e \in \delta(A)} y_A \leq c_e & & \text{for all } e \in E \\
& & & y_A \geq 0 & & \text{for all } A \in \mathcal{C}(S).
\end{aligned}
$$

In contrast to the previous dual program $D(S)$, dual variables can correspond to more than one player of S. Despite this more complex structure, we show that the standard primal-dual algorithm for this problem leads easily to an acyclic mechanism.

1. Initialize $y_A = 0$ for all $A \in \mathcal{C}(S)$, $x_e = 0$ for all $e \in E$, $F = \emptyset$, and the time t to 0. All players of S are *active*.

2. While active players remain:

   (a) Let $A_1, \ldots, A_\ell$ denote the connected components of (V, F) that include at least one active player. Uniformly increase the y-value of each such component until $\sum_{A \in \mathcal{C}(S) \,:\, e \in \delta(A)} y_S = c_e$ for some edge e with $x_e = 0$. Increase t by the same amount.

   (b) Choose such an edge e. Set $x_e = 1$ and add e to F. Deactivate every player contained in r's connected component in (V, F).

3. Output $F^* = \{e \in F \,:\, F \setminus \{e\} \text{ is infeasible}\}$.

Figure 5: The AKR-GW algorithm for ST.

Next we review the primal-dual Steiner tree algorithm designed in [1]; the primal-dual interpretation was made explicit in [16]. We call this algorithm the *AKR-GW algorithm*, and it is shown in Figure 5. The algorithm starts with the all-zero primal and dual solutions, and iteratively augments its (infeasible) primal solution one edge at a time. A player i is defined to be active while there is no path from its source s_i to the root. At each iteration, the algorithm considers the connected components of (V, F), where F is the edges selected so far. The algorithm uniformly increases the dual variables corresponding to the components that contain at least one active player. These dual variables are increased at a uniform rate until some dual constraint becomes tight. The corresponding edge is then added to the primal solution. The main loop halts when all players are inactive, at which point the current primal solution is feasible. Step 2a ensures that every dual constraint is respected, so the AKR-GW algorithm terminates with a dual feasible solution. The algorithm concludes with a pruning step (Step 3) that is relevant only for budget-balance (Theorem 5.8). Let F be the solution computed by the main loop and call an edge $e \in F$ *essential* if $F \setminus \{e\}$ is not a feasible Steiner tree. The essential edges of F form a feasible solution (see [16]), and these are the final output of the algorithm. This algorithm can be implemented in polynomial time [1, 16].

We obtain a cost-sharing method χ_{ST} and offer function τ_{ST} from the AKR-GW algorithm as follows. The offer function is defined as in our earlier applications: given S, $\tau_{ST}(i, S)$ is the time at which the AKR-GW algorithm deactivates player i in Step 2b. Defining the cost-sharing method χ_{ST} is complicated by the many-to-many correspondence between dual variables and players (cf., the dual program $D(S)$ for NMUFL). A natural solution is to divide the value of a dual variable equally among participating players [23, 30]. Formally, let $\kappa(A) \geq 1$ denote the number of source vertices of S inhabiting the S-separating set A, and define

$$\chi_{ST}(i, S) = \sum_{A \in \mathcal{C}(S) \,:\, i \in A} \frac{y_A}{\kappa(A)}, \tag{5}$$

where $\{y_A\}_{A \in \mathcal{C}(S)}$ is the dual feasible solution computed by the AKR-GW algorithm.

The cost-sharing method χ_{ST}, given $S \subseteq U$, returns the cost shares given in (5) together with the solution computed by the AKR-GW algorithm for the Steiner tree instance induced by S. Simple examples show that χ_{ST} is not cross-monotonic. We call the mechanism $M(\chi_{ST}, \tau_{ST})$ the *AKR-GW mechanism*.

4.3 Acyclicity

We now prove that all three of the mechanisms defined in Subsection 4.2 are acyclic.

4.3.1 The PD and DMV Mechanisms

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 PD method, and differ only by a fixed scaling factor in the DF method.

Lemma 4.10 *For every NMUFL problem with universe U, subset $S \subseteq U$, and player $i \in S$:*

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

(b) $\chi_{DF}(i, S) = \tau_{DF}(i, S)/\mathcal{H}_{|U|}$.

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, (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 this value divided by $\mathcal{H}_{|U|}$, (b) follows. ■

We can now prove that the PD mechanism is acyclic and hence, by Theorem 3.16, WGSP.

Theorem 4.11 *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 3.4(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 4.10(a), $\chi_{PD}(i, S \setminus T) = \chi_{PD}(i, S)$.

The proof of Definition 3.4(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 4.10(a), $\chi_{PD}(i, S \setminus T) \geq \chi_{PD}(i, S)$. ■

An identical argument proves the acyclicity of the DMV mechanism.

Theorem 4.12 *The DMV mechanism is acyclic.*

4.3.2 The AKR-GW Mechanism

The AKR-GW mechanism is also acyclic, although the argument is more delicate than for the PD and DMV mechanisms. Indeed, Example 4.15 below shows that the corresponding mechanism for a more general class of problems need not be acyclic.

First, we require a technical monotonicity lemma about the AKR-GW algorithm. A set A and the corresponding dual variable y_A are *active at time τ* in the AKR-GW algorithm if y_A is increased in Step 2a when the time t equals τ.

Lemma 4.13 *Fix a Steiner tree cost-sharing problem with universe U. For $S \subseteq U$, let $\mathcal{E}(S)$ denote the execution of the AKR-GW algorithm on the instance induced by S. Choose $T \subset S \subseteq U$. Let τ^* denote the earliest time in $\mathcal{E}(S)$ at which some source of T is deactivated.*

(a) If A contains no players of T, then A is a connected component in $\mathcal{E}(S)$ at time $\tau \leq \tau^*$ if and only if it is a connected component of $\mathcal{E}(S \setminus T)$ at time τ.

(b) If A is a connected component at time $\tau \leq \tau^*$ in $\mathcal{E}(S \setminus T)$, then there is a connected component A' in $\mathcal{E}(S)$ at time τ with $A \subseteq A'$.

Roughly, Lemma 4.13 states that removing a subset T of source vertices can only shatter active dual variables into smaller ones, up until the time at which the first source of T becomes inactive. We omit the proof and move on to establish the acyclicity of the AKR-GW mechanism.

Theorem 4.14 *The AKR-GW mechanism is acyclic.*

Proof: Fix a Steiner tree cost-sharing problem with universe U, a subset $S \subseteq U$, and a player $i \in S$. By the definition of the AKR-GW algorithm and the cost-sharing method χ_{ST} (5), we can interpret the cost share $\chi_{ST}(i, S)$ of a player i as accruing over the time interval $[0, \tau_{ST}(i, S)]$ in the AKR-GW algorithm. The marginal increase at time τ is equal to $1/\kappa(A)$, where A is player i's connected component at time τ and $\kappa(A)$ is the number of players of S contained in A.

To check Definition 3.4(a), fix S, $i \in S$, and a subset $T \subseteq G(i, S)$. Let $\mathcal{E}(S)$ and $\mathcal{E}(S \setminus T)$ denote the execution of the AKR-GW algorithm on the Steiner tree instances induced by S and $S \setminus T$, respectively. By the definition of τ_{ST}, the offer time of a player is the time at which it joins the connected component of the root in the current primal solution. Since $T \subseteq G(i, S)$, no player of T is in the same connected component as player i at or before time $\tau(i, S)$ in $\mathcal{E}(S)$. By Lemma 4.13(a), the contributions to player i's cost share are exactly the same in $\mathcal{E}(S)$ and in $\mathcal{E}(S \setminus T)$ until the time $\tau(i, S)$, at which time player i is deactivated in both executions. Hence $\chi_{ST}(i, S) = \chi_{ST}(i, S \setminus T)$, as desired.

To check Definition 3.4(b), fix $T \subseteq G(i, S) \cup (E(i, S) \setminus \{i\})$. Observe that players of T might have joined player i's connected component long before time $\tau_{ST}(i, S)$ in $\mathcal{E}(S)$. By Lemma 4.13(b), at each time prior to $\tau_{ST}(i, S)$, player i's connected component in $\mathcal{E}(S \setminus T)$ contains at most as many players as that in $\mathcal{E}(S)$, and does not contain the root r. Therefore, by time $\tau_{ST}(i, S)$, player i has accumulated at least as large a cost share in $\mathcal{E}(S \setminus T)$ as in $\mathcal{E}(S)$. Since player i's cost share in the latter execution is fixed by time $\tau_{ST}(i, S)$, the final cost shares satisfy $\chi_{ST}(i, S \setminus T) \geq \chi_{ST}(i, S)$. ■

4.3.3 Is Acyclicity Automatic?

Does every "natural" primal-dual algorithm induce an acyclic mechanism? We next formalize this question and answer it negatively. We restrict attention to primal-dual algorithms that share the following properties with the ones studied in this paper. First, the primal-dual algorithm maintains a notion of time. Second, dual variables correspond to non-empty subsets of players, are initially zero, and are only increased throughout the algorithm. (The DF algorithm for NMUFL can be interpreted as a such an algorithm by moving its final scaling step inside its main loop.) Third, at every point in time, every player is classified as either *active* or *inactive*, and a dual variable is increased only if it corresponds to at least one active player. Finally, the algorithm should terminate with feasible primal and dual solutions.

Every such algorithm induces the following cost-sharing method and offer function. For a subset S of players and a player $i \in S$, the offer time $\tau(i, S)$ is the latest moment in time at

which player i is active in the execution of the primal-dual algorithm on the optimization problem induced by S. The cost-sharing method χ returns the primal solution constructed by the primal-dual algorithm and defines player i's cost share as follows. At the beginning of the primal-dual algorithm, $\chi(i, S)$ is initialized to zero. Whenever some dual variable is increased in the algorithm, this increase is split equally among the active players to which this variable corresponds. Thus, at every moment in time, the sum of the players' cost shares equals the sum of the dual variables. We call $M(\chi, \tau)$ the *canonical mechanism* induced by the primal-dual algorithm. The PD, DMV, and AKR-GW mechanisms are all canonical in this sense.

Not all canonical mechanisms are acyclic. A *Steiner forest cost-sharing problem* is specified by the same data as a Steiner tree problem, except that the sources $U = \{s_1, \ldots, s_k\}$ and root r are replaced by a set $U = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of source-sink pairs (the players). For a subset $S \subseteq U$, $C(S)$ is defined as the minimum cost of a subgraph of G that contains at least one path between each s_i-t_i pair of S. The AKR-GW algorithm of Figure 5 extends to (and was originally designed for) Steiner forest problems [1, 16]; the only difference is that a player is defined to be active if and only if its source and sink lie in different connected components. The next example shows that the canonical mechanism for this primal-dual algorithm is not acyclic.

Example 4.15 Consider a Steiner forest cost-sharing problem with player set $U = \{1, 2, \ldots, k\}$ and graph equal to the path of nodes $s_1, s_3, s_2, t_1, t_3, t_2$. The middle three edges have unit cost and the outer two edges have cost $3/2$. The first player corresponds to (s_1, t_1), the second to (s_2, t_2), and $k-2$ players have sources and sinks co-located at s_3 and t_3, respectively. Let χ_{SF} denote the cost-sharing method induced by the AKR-GW algorithm. When this algorithm is run with all players present, at time $t = 1/2$, all three of the middle edges will be selected. All players other than the first two are deactivated at this juncture. The cost share of each of the first two players is 1 at this time, with a contribution of $1/2$ from both the source and the sink of each player. The final two edges are selected at $t = 3/4$, and the final cost shares of both players are $11/8$, with s_1 and t_2 contributing a further $1/4$ to their respective cost shares, and s_2 and t_1 each contributing $1/8$.

Now suppose the second player is absent. The sink t_1 is in a singleton connected component until $t = 1/2$, at which point edge (t_1, t_3) is selected. Similarly, the source s_1 is isolated until time $t = 3/4$, when edge (s_1, s_3) is selected. These contributions to the first player's cost share equal $5/4$. All other dual variable growth involving s_1 or t_1 is split equally among the first player and the last $k-2$ players; provided k is sufficiently large, the corresponding contributions to player 1's cost share are negligible. Thus, $\chi_{SF}(1, U \setminus \{2\}) \approx 5/4 < 11/8 = \chi_{SF}(1, U)$. By symmetry, $\chi_{SF}(2, U \setminus \{1\}) \approx 5/4 < 11/8 = \chi_{SF}(2, U)$. This mutual failure of cross-monotonicity implies that *no* offer function, canonical or otherwise, can be valid for χ_{SF} (recall Remark 3.8).

Remark 4.16 There is, however, a 2-budget-balanced and $O(\log^2 k)$-approximate Moulin mechanism for Steiner forest problems [30]. This mechanism is canonical for a primal-dual algorithm that is similar to the AKR-GW algorithm but different in an important respect: the duration of activity of each player is independent of the presence or absence of other players.

Characterizing the primal-dual algorithms that induce acyclic canonical mechanisms is an interesting open problem.

5 Improved Approximation Guarantees

This section proves tight upper and lower bounds on the approximate budget-balance (Section 5.1) and efficiency (Section 5.2) of the three acyclic mechanisms defined in the previous section.

5.1 Budget-Balance Guarantees

This section shows how budget-balance guarantees for all of the mechanisms defined in Section 4.2 follow easily from existing work in the approximation algorithms literature.

5.1.1 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. The more important application of this mechanism is to Vertex Cover problems, for which $d_{max} = 2$ (cf., Figure 3(b)). Extending the well-known analysis of primal-dual Set Cover algorithms implies the following guarantee for the PD algorithm.

Lemma 5.1 *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 5.1 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 [20] and Vazirani [43, Chapter 15].

In addition, the dual solution computed by the PD algorithm is feasible (Lemma 4.1), and hence the computed primal and dual solutions satisfy weak duality (3). As discussed in Section 4.2.1, since the cost shares of the PD method are the dual variables computed by the PD algorithm, budget-balance of the PD method and mechanism follow.

Theorem 5.2 *For every NMUFL cost-sharing problem, the PD mechanism is d_{max}-budget-balanced.*

Recall that every Moulin mechanism for Vertex Cover problems is $\Omega(k^{1/3})$-budget-balanced [21]. Assuming the Unique Games Conjecture [28], the budget-balance guarantee in Theorem 5.2 is the best possible for a polynomial-time mechanism for small values of d_{max} [29].

5.1.2 The DMV Mechanism for NMUFL and Metric UFL Problems

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 4.4 and Step 3 of the DF algorithm.

Lemma 5.3 *For every NMUFL instance, the DF algorithm computes a primal solution $\{x^*_{qT}\}_{(q,T)\in\mathcal{C}(S)}$ and a dual solution $\{\alpha^*_i\}_{i\in S}$ satisfying*

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

Also, by Lemma 4.5, the dual solution computed by the DF algorithm is feasible. As with Theorem 5.2, budget-balance follows.

Theorem 5.4 ([10]) *For every NMUFL cost-sharing problem with k players, the DMV mechanism is $\mathcal{H}_k$-budget-balanced.*

Every Moulin mechanism for NMUFL problems is $\Omega(\sqrt{k})$-budget-balanced [21]. Under standard complexity assumptions, the budget-balance guarantee in Theorem 5.4 is the best possible for polynomial-time NMUFL mechanisms [13].

After a minor modification, the DMV mechanism can achieve radically better budget-balance for the special case of metric UFL problems. In particular, the *metric DF algorithm* is the same as the DF algorithm, except dual variables are only scaled by a factor of 1.861 in Step 3 of the algorithm. This change clearly has no effect on the acyclicity of the mechanism. Jain et al. [22] proved the following.

Lemma 5.5 ([22]) *For every metric UFL instance, the metric DF algorithm terminates with a dual feasible solution.*

Budget-balance of the canonical mechanism follows.

Theorem 5.6 ([10]) *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 [21].

Remark 5.7 The budget-balance guarantee in Theorem 5.6 can be improved using a slightly different mechanism. Jain et al. [22] 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 proof of Theorems 4.11 and 4.12 carries over to show that the canonical mechanism induced by this refined algorithm is acyclic. As in Theorem 5.6, this mechanism is 1.61-budget-balanced.

5.1.3 The AKR-GW Mechanism for ST Problems

Finally, we argue that the AKR-GW algorithm is 2-budget-balanced. This fact nearly follows from earlier work [1, 16]; the only complication arises from the lack of dual variables for components that contain the root vertex. Modifying the proof in [16] (details omitted) yields the following result.

Theorem 5.8 *For every ST cost-sharing problem, the AKR-GW mechanism is 2-budget-balanced.*

The budget-balance guarantee in Theorem 5.8 matches the lower bound known for Moulin mechanisms [30]. The first 2-budget-balanced Moulin mechanism for ST problems was given in Jain and Vazirani [23]. An interesting open question is whether or not there are polynomial-time β-budget-balanced acyclic mechanisms for ST problems with $\beta < 2$. Such a mechanism cannot be based directly on the linear relaxation proposed in Section 4.2.3, which can have an integrality gap arbitrarily close to 2 [43, Example 22.10].

5.2 Efficiency Guarantees

This section proves matching upper and lower bounds on the approximate efficiency achieved by the three mechanisms defined in Section 4.2. Recall from Section 2.2 that the social cost of a mechanism M for an outcome S with valuation profile v is defined as $C_M(S)+\sum_{i \notin S} v_i$, where C_M is the cost of the feasible solution produced by the mechanism, and that a mechanism is α-approximate if its social cost is always at most α times the minimum-possible (1).

5.2.1 The PD and DMV Mechanisms for NMUFL Problems

We obtain efficiency guarantees for the PD and DMV mechanisms for NMUFL problems as a consequence of the following more general result.

Theorem 5.9 *Let $M(\chi, \tau)$ be a β-budget-balanced acyclic mechanism for a cost-sharing problem C with universe U of k players 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 $T \subseteq S$,

$$\sum_{i \in T} \chi(i, S) \leq C(T). \tag{6}$$

Then, $M(\chi, \tau)$ is $(\mathcal{H}_k + \beta)$-approximate.

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. [38]), demanding that each coalition T has no incentive to secede from the mechanism and seek service elsewhere at cost $C(T)$.

Theorem 5.9 has immediate implications for the PD and DMV mechanisms.

Corollary 5.10 *For every NMUFL cost-sharing problem, the PD mechanism is $O(d_{max} + \log k)$-approximate, where d_{max} is the largest number of facilities to which a demand can be assigned at finite cost.*

Proof: To check condition (6), 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 $\{\chi_{PD}(i, S)\}_{i \in S}$ form a feasible solution to the dual program $D(S)$ of Section 4.2.1 (Lemma 4.1). The subset of cost shares $\{\chi_{PD}(i, S)\}_{i \in T}$ form a feasible solution to the dual program $D(T)$. Condition (6) follows from weak duality.

The corollary is now immediate from Lemma 4.10(a), Theorem 5.2, and Theorem 5.9. ■

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 [21, 40].

Corollary 5.11 *For every NMUFL cost-sharing problem, the DMV mechanism is $O(\log k)$-approximate.*

Proof: Immediate from Lemma 4.5, Lemma 4.10(b), Theorem 5.4, and Theorem 5.9. ■

Every Moulin mechanism for NMUFL problems is $\Omega(\sqrt{k})$-approximate [21, 40].

Remark 5.12 Corollary 5.11 also applies to the 1.61-budget-balanced metric UFL mechanism discussed in Remark 5.7.

Our proof of Theorem 5.9 depends on two lemmas. The first upper bounds the service cost incurred by the mechanism in terms of the service cost and part of the excluded valuations of an optimal solution.

Lemma 5.13 *Let $M = M(\chi, \tau)$ be a β-budget-balanced acyclic mechanism for a cost-sharing problem C with universe U that satisfies property (P2) of Theorem 5.9. 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,*

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

Proof: Since M is β-budget-balanced,

$$C_M(S) \leq \beta \cdot \left(\sum_{i \in S \cap S^*} \chi(i, S) + \sum_{i \in S \setminus S^*} \chi(i, S) \right). \tag{7}$$

By property (P2) and since C is nondecreasing,

$$\sum_{i \in S \cap S^*} \chi(i, S) \leq C(S \cap S^*) \leq C(S^*). \tag{8}$$

By individual rationality (Proposition 3.9(d)), $\chi(i, S) \leq v_i$ for every $i \in S \setminus S^*$; combining this with inequalities (7) and (8) 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 5.14 *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) of Theorem 5.9. 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,*

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

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

$$\chi(i, S_i) \leq \frac{C(S^*)}{\ell - i + 1} \tag{9}$$

for every $i \in \{1, 2, \ldots, \ell\}$. Player i's deletion from S_i implies that $v_i < \chi(i, S_i)$; summing (9) 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 3.11 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:

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

since player i's cost share is the smallest of the $(\ell - i + 1)$ remaining players of $S^* \setminus S$, it is at most $C(S^*)/(\ell - i + 1)$. This establishes (9) and completes the proof. ■

Theorem 5.9 now follows easily.

Proof of Theorem 5.9: Fix a cost-sharing problem with universe U and a valuation profile v for U. Applying Lemmas 5.13 and 5.14, we upper bound the social cost incurred by M in terms of the optimal social cost as follows:

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

■

Remark 5.15 Lemma 5.14 and Theorem 5.9 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 5.9 cannot be removed: even for extremely simple cost-sharing problems, *every* $O(1)$-budget-balanced acyclic mechanism is $\Omega(\log k)$-approximate.

Example 5.16 (Public excludable good) In a *public excludable good* cost-sharing problem, $C(\emptyset) = 0$ and $C(S) = 1$ for every nonempty set S. This problem can be interpreted as a metric UFL problem with all players co-located with a single facility that has unit opening cost. It can also be interpreted as a Vertex Cover problem on a star graph, in which the center has unit weight and the other vertices have infinite weight.

Theorem 5.17 *Every β-budget-balanced acyclic mechanism for a public excludable good problem with k players is at least $(\mathcal{H}_k/\beta)$-approximate.*

Proof: Fix a universe U of k players and a β-budget-balanced 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 $\chi(i, S) \geq 1/\beta|S|$. In proof, let $T \subset S$ denote the players with offer time strictly larger than the minimum. Since χ is β-budget-balanced, $\sum_{i \in S \setminus T} \chi(i, S \setminus T) \geq C(S \setminus T)/\beta = 1/\beta$ and hence $\chi(i, S \setminus T) \geq 1/\beta|S \setminus T| \geq 1/\beta|S|$ for some player $i \in S \setminus T$. Invoking Definition 3.4(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 $i = 1, 2, \ldots, k$, 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 $1/\beta(k - i + 1)$. Now set the valuation v_i of player i to $1/\beta(k - i + 1) - \epsilon$ for small $\epsilon > 0$. The optimal social cost is at most 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 incurs social cost $\sum_{i=1}^{k} v_i \approx \mathcal{H}_k/\beta$. ■

Very recently, Dobzinski et al. [11] extended this lower bound to *all* strategyproof mechanisms: every $O(1)$-budget-balanced randomized or deterministic SP mechanism is $\Omega(\log k)$-approximate for public excludable good problems.

5.2.2 The AKR-GW Mechanism for ST Problems

We now establish the following efficiency guarantee for the AKR-GW mechanism.

Theorem 5.18 *For every ST cost-sharing problem, the AKR-GW mechanism is $O(\log^2 k)$-approximate.*

Lemma 5.14 does not apply to this mechanism because its cost shares need not be proportional to its offer times. However, the mechanism satisfies condition (6), enabling the application of Lemma 5.13.

Lemma 5.19 *For every ST cost function, the AKR-GW cost-sharing method χ_{ST} satisfies property (P2) of Theorem 5.9.*

Proof: By the definition of χ_{ST} (5),

$$\sum_{i \in T} \chi_{ST}(i, S) \leq \sum_{A \in \mathcal{C}(T)} y_A^*,$$

where $\{y_A^*\}_{A \in \mathcal{C}(S)}$ is the dual feasible solution produced by the AKR-GW algorithm for $D(S) - ST$. Since $\{y_A^*\}_{A \in \mathcal{C}(T)}$ is feasible for $D(T) - ST$, the lemma follows from weak duality. ■

We require a surrogate for Lemma 5.14—an upper bound on the excluded valuations $\sum_{i \in S^* \setminus S} v_i$. We accomplish this by a "reduction" to another ST cost-sharing method designed by Jain and Vazirani [23]. We omit a detailed description of the method, and note only that it is cross-monotonic [23, Theorem 3] and canonical in the sense of Section 4.3.3 for a primal-dual algorithm of Edmonds [12].

Roughgarden and Sundararajan [40, Theorem 4.5] proved that the Jain-Vazirani method χ_{JV} is $O(\log^2 k)$-*summable*, which has the following consequence.

Proposition 5.20 ([40]) *There is a constant $a > 0$ such that the following holds: for every Steiner tree problem C with universe U of k players, subset $S \subseteq U$, and ordering σ of the players,*

$$\sum_{i=1}^{|S|} \chi_{JV}(i, S_i) \leq (a \log^2 k) \cdot C(U),$$

where i and S_i denote the ith player of S and the set of all players of U that follow i (including i itself) with respect to σ.

We also use the fact that the JV method χ_{JV} dominates the AKR-GW method χ_{ST} in the following sense.

Lemma 5.21 *For every ST cost-sharing problem with universe U, subset $S \subseteq U$, and player $i \in S$, $\chi_{JV}(i, S) \geq \chi_{ST}(i, S)/2$.*

Lemma 5.21 follows from a monotonicity result similar to Lemma 4.13(b). (The factor of 2 arises because of a scaling step required to obtain the method χ_{JV} from the dual variables in Edmonds's algorithm [12].)

Our proxy for Lemma 5.14 is as follows.

Lemma 5.22 *Fix a Steiner tree problem C with universe U of k players and a valuation profile v for U. Let S be the outcome of the AKR-GW mechanism on input v and S^* the outcome with optimal social cost. Then,*

$$\sum_{i \in S^* \setminus S} v_i = O(\log^2 k) \cdot C(S^*).$$

Proof: Let σ denote the order in which players are deleted by the AKR-GW mechanism, with players of S appearing last in an arbitrary relative order. Define ℓ, i, and S_i as in the proof of Lemma 5.14. We have

$$\sum_{i \in S^* \setminus S} v_i < \sum_{i=1}^{\ell} \chi_{ST}(i, S_i) \leq 2 \sum_{i=1}^{\ell} \chi_{JV}(i, S_i) \leq 2 \sum_{i=1}^{\ell} \chi_{JV}(i, S_i \cap S^*) = O(\log^2 k) \cdot C(S^*);$$

the first inequality follows from the definition of the AKR-GW mechanism, the second from Lemma 5.21, the third from the cross-monotonicity of χ_{JV}, and the final bound from applying Proposition 5.20 to the ST cost-sharing problem induced by S^* and to the subset $S^* \setminus S$. ■

Proof of Theorem 5.18: Identical to the proof of Theorem 5.9, with Lemma 5.14 replaced by Lemma 5.22. ■

Adapting an example from [40] shows that the bound in Theorem 5.18 is tight, up to a constant factor. An interesting open question is whether or not $O(1)$-budget-balanced, polynomial-time acyclic mechanisms can achieve $o(\log^2 k)$-approximate efficiency for ST cost-sharing problems. Full budget-balance and $O(\log k)$-approximate efficiency are possible if the polynomial-time constraint is dropped [2].

5.2.3 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* [40] for a cost-function C if, for every ordering σ of the players of U and every subset $S \subseteq U$,

$$\sum_{\ell=1}^{|S|} \chi(i_\ell, S_\ell) \leq \alpha \cdot C(S) \tag{10}$$

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, β-budget-balanced cost-sharing method, then it is $\Theta(\alpha + \beta)$-approximate [40].

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 (10).

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. An important open problem is to obtain useful and widely applicable upper or lower bounds on the approximate efficiency of an acyclic mechanism in terms of its cost-sharing method and offer function.

6 General Demand Cost-Sharing Problems

We now extend acyclic mechanisms to general demand cost-sharing problems, in which players can be allocated one of several "levels of service". The next section applies this framework to fault-tolerant facility location problems.

6.1 Preliminaries

In a *general demand cost-sharing problem*, there is a universe $U = \{1, 2, \ldots, k\}$ of players. Each player i has a publicly known maximum level of service R_i, a positive integer. An allocation S is now a vector $(s_1, \ldots, s_k)$ of nonnegative integers with $s_i \leq R_i$ for every i, which describes the level of service offered to each player. The cost function C describes the cost incurred by the mechanism as a function of the allocation S. We assume that the cost of the all-zero vector is zero, and that the cost is nondecreasing in each component. We also assume that every player prefers higher levels of service, but obtains diminishing returns. In other words, the private valuation of a player i is a nonnegative vector v_i, where $v_i(j)$ denotes the marginal value of level j (over level $j-1$) to player i, and we assume that $v_i(j)$ is nonincreasing in j. A player's bid $b_i = (b_i(1), \ldots, b_i(R_i))$ is a vector of announced marginal values, which must also be nonincreasing in the level of service. Given a bid vector from each player, a mechanism must determine an allocation S, a feasible solution to the optimization problem induced by S, and a price p_i to charge each player. The utility of the player is then $u_i(S, p_i) = v_i(s_i) - p_i$, where $v_i(s_i) = \sum_{1 \leq j \leq s_i} v_i(j)$ denotes the total value the player has for the allocation.

Example 6.1 (FTUFL) A *fault-tolerant uncapacitated facility location (FTUFL)* cost function is induced by a UFL instance (Example 2.1)—demands (players) U, facilities F with opening costs f, connection costs c—and also a requirement vector R, indexed by U. The value R_i is the maximum number of *distinct* open facilities to which a player might be connected; in UFL, $R_i = 1$ for every player. The cost $C(S)$ is the cost of the optimal way to open facilities and connect each demand i to s_i distinct open facilities. For example, if $S = (2, 1, 0)$ in the instance shown in Figure 1, then the optimal solution is to open both facilities and connect each player i to the nearest s_i facilities. The cost of the solution is 17/2. A valuation $v_i(1), \ldots, v_i(R_i)$ for a player i in an FTUFL problem describes, for each level of service j, the additional value i derives from being connected to j facilities over $j-1$ facilities.

Our objectives of incentive-compatibility, budget-balance, and economic efficiency extend to general demand problems in a straightforward way. The definition of SP and WGSP mechanisms are identical to those in Section 2.1, with player bids and utilities defined as above. The definition of (approximate) budget-balance requires no modification. As in the binary demand case, the *social cost* incurred by a mechanism M is the service cost $C_M(S)$ incurred plus the total excluded valuation: $\sum_{i \in U} \sum_{j > s_i} v_i(j)$. The optimal social cost is now

$$\min_{S \subseteq U} \left[C(S) + \sum_{i \in U} \sum_{j=s_i+1}^{R_i} v_i(j) \right].$$

As before, a mechanism is *α-approximate* if its social cost is always at most α times the minimum possible.

A *cost-sharing method* χ for a general demand problem takes as input an allocation S and returns a feasible solution for the optimization problem induced by S, as well as a cost share $\chi(i, j, S)$ for

each player $i \in U$ and $j \leq s_i$. The total cost share assigned to player i is $\sum_{j \leq s_i} \chi(i, j, S)$. We call a cost-sharing method *demand-monotone* if, for every player i and allocation S, $\chi(i, j, S)$ is nondecreasing in $j \in \{1, \ldots, s_i\}$. Finally, an *offer function* τ assigns an offer time $\tau(i, j, S)$ for each $i \in U$ and $j \leq s_i$.

6.2 Acyclic Mechanisms for General Demand Problems

As in the binary demand case, a cost-sharing method and an offer function together define a mechanism that simulates an iterative auction.

Definition 6.2 Let U be a universe of players, where player i has maximum level of service R_i. Let χ and τ denote a cost-sharing method and an offer function defined on the possible allocations. 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 $S := (R_1, \ldots, R_k)$.
3. If $b_i(j) \geq \chi(i, j, S)$ for every $i \in U$ and $j \leq s_i$, then halt. Output the allocation S, the feasible solution constructed by χ, and charge each player $i \in U$ the price $p_i = \sum_{j=1}^{s_i} \chi(i, j, S)$.
4. Among all players $i \in U$ and levels $j \in \{1, \ldots, s_i\}$ with $b_i(j) < \chi(i, j, S)$, let (i^*, j^*) be one with minimum $\tau(i, j, S)$. (Break ties arbitrarily.)
5. Set $s_{i^*} = j^* - 1$ and return to Step 3.

As in Proposition 3.9, the mechanism M inherits the budget-balance guarantee of its underlying cost-sharing method. Also, if χ and τ are polynomial-time algorithms, then so is $M(\chi, \tau)$. The mechanism clearly has no positive transfers, and it satisfies a strengthened form of individual rationality. Precisely, the *marginal price* $p_i(j)$ charged to a player i for level of service j by such a mechanism $M(\chi, \tau)$ in an outcome S is defined as $\chi(i, j, S)$ if $j \leq s_i$ and 0 if $j > s_i$.

Proposition 6.3 *Let $M = M(\chi, \tau)$ be a general demand mechanism induced by the cost-sharing method χ and offer function τ. For every bid vector b, the mechanism M computes an allocation S and marginal prices p such that, for every player i and level of service j: (i) if $j \leq s_i$, then $p_i(j) \leq b_i(j)$; and (ii) if $j > s_i$, then $p_i(j) = 0$.*

Individual rationality in the standard sense (Section 2.1) follows from Proposition 6.3 by summing marginal prices and bids over the service levels.

The next definition identifies conditions on a cost-sharing method and offer function so that the induced general demand mechanism is truthful (and even WGSP). Call a set of pairs $P = \{(i, j)\}$ of positive integers *closed* if $(i, j) \in P$ whenever $(i, j + 1) \in P$. Allocations S are defined as nonnegative vectors but correspond to closed sets of pairs in a natural way, with a pair (i, j) indicating that player i receives level of service at least j. We use these two representations of allocations interchangeably. For an allocation S and $(i, j) \in S$, define the sets $L(i, j, S)$, $E(i, j, S)$, and $G(i, j, S)$ by $\{(i', j') \in S : \tau(i', j', S) < \tau(i, j, S)\}$, $\{(i', j') \in S : \tau(i', j', S) = \tau(i, j, S)\}$, and $\{(i', j') \in S : \tau(i', j', S) > \tau(i, j, S)\}$, respectively.

Definition 6.4 Let χ and τ be a cost-sharing method and an offer function, respectively, defined on a universe U in which player i has maximum level of service R_i. The function τ is *valid for* χ if the following three properties hold for every allocation S and player $i \in U$:

(a) for every $j \leq s_i$ and $T \subseteq G(i,j,S)$ with $S \setminus T$ closed, $\chi(i,j,S \setminus T) = \chi(i,j,S)$;

(b) for every $j \leq s_i$ and $T \subseteq G(i,j,S) \cup (E(i,j,S) \setminus \{(i,j)\})$ with $S \setminus T$ closed, $\chi(i,j,S \setminus T) \geq \chi(i,j,S)$;

(c) offer times $\tau(i,j,S)$ are strictly increasing in j.

The first two conditions are natural generalizations of those in Definition 3.4. The general demand setting also necessitates the third condition.

Remark 6.5 Definition 6.4(c) can be relaxed so that offer times are only nondecreasing in j, provided ties in Step 4 in Definition 6.2 are broken in favor of higher service levels.

Definition 6.6 A *general demand acyclic mechanism* is a mechanism $M(\chi, \tau)$ induced by a demand-monotone cost-sharing method χ and an offer function τ that is valid for χ.

We then have the following incentive-compatibility guarantee.

Theorem 6.7 *Every general demand acyclic mechanism is WGSP.*

To prove Theorem 6.7, we require analogues of Lemmas 3.10 and 3.12. We say that player i is *offered the marginal price $p_i(j)$ in iteration ℓ* of a general demand acyclic mechanism $M(\chi, \tau)$ if the following conditions hold: first, if S is the current allocation at the beginning of the ℓth iteration, then $(i,j) \in S$; second, if player i^*'s service level is decreased to $j^* - 1$ in this iteration, then $\tau(i,j,S) \leq \tau(i^*, j^*, S)$; third, the price $p_i(j)$ is the cost share $\chi(i,j,S)$. The first lemma then states that, in an acyclic mechanism, the marginal price of a player-service level pair (i,j) is fixed once a marginal price is offered for some pair with a subsequent offer time.

Lemma 6.8 *Suppose an acyclic mechanism $M(\chi, \tau)$ offers marginal prices to players i and i' for service levels j and j' in an iteration with current allocation S, and $\tau(i,j,S) < \tau(i',j',S)$. Then $\chi(i,j,S)$ is the only marginal price offered to i for service level j in subsequent iterations.*

The second lemma proves that marginal prices only increase throughout the execution of an acyclic mechanism.

Lemma 6.9 *If a general demand acyclic mechanism $M(\chi, \tau)$ offers a player i the marginal price $p_i^1(j)$ in some iteration and the marginal price $p_i^2(j)$ in a subsequent iteration, then $p_i^1(j) \leq p_i^2(j)$.*

The proofs of Lemmas 6.8 and 6.9 rely only on parts (a) and (b) of Definition 6.4 and are the same as those of Lemmas 3.10 and 3.12. We can now prove Theorem 6.7.

Proof of Theorem 6.7: Let $M(\chi, \tau)$ be a general demand acyclic mechanism with universe U and vector R of maximum levels of service. Fix a coalition $T \subseteq U$, a valuation v_i and bid b_i for each player $i \in T$, and bids for the players of $U \setminus T$. Let $\mathcal{E}_v$ and $\mathcal{E}_b$ denote the executions of M in which each player $i \in T$ bids v_i and b_i, respectively. If these executions are identical, they terminate with equal allocations and prices, and every player obtains equal utility in both. So consider the first iteration in which $\mathcal{E}_v$ and $\mathcal{E}_b$ differ, necessarily because player $i \in T$ is offered a marginal price for some service level j that lies between $b_i(j)$ and $v_i(j)$. Since offer times are strictly increasing with the service level (Definition 6.4(c)), Lemma 6.8 implies that the marginal prices offered to i for service levels 1 through $j-1$ are fixed at their current values throughout the remainder of $\mathcal{E}_v$

and $\mathcal{E}_b$. Therefore, player i derives the same utility from these service levels in both executions. For service levels j and above, we consider two cases.

Case 1: Suppose $b_i(j) > v_i(j)$. In the first iteration in which $\mathcal{E}_v$ and $\mathcal{E}_b$ differ, player i is offered a marginal price $p_i(j) \in (v_i(j), b_i(j)]$. Since offered marginal prices only increase (Lemma 6.9), at this or some subsequent iteration in $\mathcal{E}_v$, player i's service level will be reduced to $j-1$ or less. Thus in $\mathcal{E}_v$, player i receives zero utility for service levels j and above. In $\mathcal{E}_b$, since offered marginal prices for level j only increase (Lemma 6.9), and since χ is demand-monotone (Definition 6.6), player i is charged at least $p_i(j)$ for each service level j and above. By assumption, player i's marginal valuations are decreasing with service level, and hence i receives nonpositive utility from service levels j and above in $\mathcal{E}_b$.

Case 2: Suppose $b_i(j) < v_i(j)$. In the first iteration in which $\mathcal{E}_v$ and $\mathcal{E}_b$ differ, player i is offered a marginal price $p_i(j) \in (b_i(j), v_i(j)]$. Arguing as in Case 1, player i receives zero utility from service levels j and above in $\mathcal{E}_b$. By Proposition 6.3, player i receives nonnegative utility from these service levels in $\mathcal{E}_v$.

In both cases, player i's total utility in $\mathcal{E}_v$ is at least that in $\mathcal{E}_b$, so the proof is complete. ■

7 Fault Tolerant Facility Location

This section applies the mechanism design framework of the previous section to FTUFL problems (Example 6.1). Our main result is as follows.

Theorem 7.1 *There is an acyclic mechanism for non-metric FTUFL that is $\mathcal{H}_k$-budget-balanced and $(2\mathcal{H}_k + \mathcal{H}_{R_{max}})$-approximate, where k is the number of players and $R_{\max} = \max_{i \in U} R_i$ is the highest level of service that can be offered to a player.*

We prove Theorem 7.1 by extending the DMV mechanism of Section 4.2.2. Essentially the same mechanism was studied in [10].

We derive a demand-monotone cost-sharing method and an offer function for FTUFL using a generalization of the DF algorithm (Section 4.2.2), which we call the *FTDF algorithm*. Recall from Remark 4.8 that the DF algorithm can be viewed as a greedy algorithm: at each step, it chooses a star (q, T) comprising active players T with minimum cost effectiveness, where the cost effectiveness of a star (q, T) is $(\sum_{i \in T} c(q, i))/|T|$ if facility q is already open and $(f_q + \sum_{i \in T} c(q, i))/|T|$ otherwise. The FTDF algorithm, when supplied with a UFL instance and a nonnegative requirement s_i for each player $i \in U$, repeatedly chooses the star (q, T) with minimum cost effectiveness, where T is a set of players that each require at least one further connection, until each player i is connected to s_i facilities. Of course, a player i cannot participate in a star (q, T) if i is already connected to q. We define the offer time $\tau_{FTDF}(i, j, S)$ to be the cost effectiveness of the jth star in which player i participates, and the cost share $\chi_{FTDF}(i, j, S)$ to be this same value, scaled down by an $\mathcal{H}_{|U|}$ factor. We call the induced mechanism $M(\chi_{FTDF}, \tau_{FTDF})$ the *FTDMV mechanism*.

A variant on an argument of Rajagopolan and Vazirani [39], also described in Vazirani [43, Section 13.2], shows that the FTDF algorithm can be interpreted as a dual fitting algorithm. In particular, the (scaled) cost shares can be mapped to a dual feasible solution to a FTUFL linear programming relaxation given by Jain and Vazirani [26]. Since the sum of the cost shares assigned by the method χ_{FTDF} equals the cost of the feasible solution that it constructs, divided by $\mathcal{H}_{|U|}$, budget-balance of the FTDMV mechanism follows.

Lemma 7.2 *For every non-metric FTUFL cost-sharing problem with k players, the FTDMV mechanism is $\mathcal{H}_k$-budget-balanced.*

Next we discuss acyclicity.

Lemma 7.3 *The FTDMV mechanism is acyclic.*

Proof: First consider two iterations, not necessarily consecutive, of the FTDF algorithm. Let (q_1, T_1) and (q_2, T_2) denote the stars chosen in these iterations. Every player active in the later iteration was active in the earlier one, and every player already connected to q_2 in the earlier one is also connected to q_2 prior to the later one. Thus, the star (q_2, T_2) was eligible for selection in the earlier iteration. That (q_1, T_1) was selected instead implies at least one of the following two statements: (i) the cost effectiveness of (q_1, T_1) is at most that of (q_2, T_2); (ii) $q_1 = q_2$. Since a player i can only participate in a single star with a given facility, the cost effectiveness of the stars in which i participates is nondecreasing throughout the FTDF algorithm.

This fact immediately implies that the offer function τ_{FTDF} satisfies the relaxation of Definition 6.4(c) discussed in Remark 6.5. Since cost shares are proportional to offer times, it also immediately implies that the cost-sharing method χ_{FTDF} is demand-monotone. Finally, a variant on the proof of Theorem 4.11 shows that parts (a) and (b) of Definition 6.4 hold, which completes the proof of acyclicity. ■

We establish an efficiency guarantee for the FTDMV mechanism via the following extension of Theorem 5.9 to general demand mechanisms.

Theorem 7.4 *Let C be a general demand cost-sharing problem with universe U of k players and maximum offer level R_{max}. Let $M(\chi, \tau)$ be a β-budget-balanced acyclic mechanism for C such that:*

(P1) for some constant $\gamma > 0$, $\chi(i, j, S) = \gamma \cdot \tau(i, j, S)$ for every requirement vector S, player $i \in U$, and service level $j \leq s_i$;

(P2) for all allocation vectors S, T with $t_i \leq s_i \leq R_i$ for all $i \in U$,

$$\sum_{i \in U} \sum_{1 \leq j \leq t_i} \chi(i, j, S) \leq C(T).$$

Then, $M(\chi, \tau)$ is $(\mathcal{H}_k + \mathcal{H}_{R_{max}} + \beta)$-approximate.

Theorem 7.4 follows easily from analogues of Lemmas 5.13 and 5.14 for general demand mechanisms, where the $\mathcal{H}_k$ bound in Lemma 5.14 is replaced by $\mathcal{H}_k + \mathcal{H}_{R_{max}}$.

Corollary 7.5 *For every non-metric FTUFL cost-sharing problem with k players and maximum offer level R_{max}, the FTDMV mechanism is $(2\mathcal{H}_k + \mathcal{H}_{R_{max}})$-approximate.*

Proof Sketch: The FTDMV mechanism clearly satisfies property (P1) of Theorem 7.4. As in Corollary 5.10, it satisfies property (P2) because it employs cost shares that can be mapped to a dual feasible solution of a suitably structured linear programming relaxation [26, 39]. The corollary now follows from Lemma 7.2 and Theorem 7.4. ■

Theorem 7.1 follows immediately from Lemma 7.2, Lemma 7.3, and Corollary 7.5.

Remark 7.6 Can Theorem 7.1 be improved for metric instances of FTUFL? One approach would be to prove that, for metric instances, scaling the offer times of the FTDF algorithm by a factor of $\beta \ll \mathcal{H}_{|U|}$ produces a budget-balanced cost-sharing method (cf., Theorem 5.6). For the special case of *uniform* metric FTUFL instances, where all players have a common connectivity requirement, Jain et al. [22, 32] proved that scaling offer times by a 1.81 factor is enough. For non-uniform metric instances, which are unavoidable in a mechanism design context, no upper bound on this scaling factor smaller than $\mathcal{H}_k$ is known.

On the other hand, we can use a different mechanism to obtain better budget-balance and efficiency guarantees when the maximum offer level R_{max} is small. Specifically, the binary demand metric UFL cost-sharing method of Pál and Tardos [36] can be invoked iteratively to define an $O(R_{max}^2)$-budget-balanced, $O(R_{max}^2 + \log k)$-approximate acyclic mechanism for metric FTUFL problems. The details are technical and deferred to a future paper. Very recently, Bleischwitz and Schoppmann [3] modified the Pál-Tardos method so that applying it iteratively directly gives an $O(R_{max})$-budget-balanced and $O(R_{max} \cdot \log k)$-approximate metric FTUFL mechanism that is also GSP.

8 Conclusions and Open Problems

We developed a framework for designing approximately budget-balanced and efficient cost-sharing mechanisms that subsumes previous work of Moulin [34]. We demonstrated its applicability by showing how well-known algorithms naturally induce mechanisms with performance guarantees provably superior to those achievable via Moulin mechanisms. Our work suggests a large number of directions for future research; we list some of them below, loosely organized by topic.

8.1 Better Approximation Guarantees

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

1. 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?

2. Is there a polynomial-time, $O(1)$-budget-balanced, $o(\log^2 k)$-approximate acyclic mechanism for Steiner tree cost-sharing problems?

3. Metric UFL algorithms with approximation ratio less than 1.61 are known [6, 33]. Can these be used to obtain polynomial-time acyclic mechanisms with comparable budget-balance and reasonable approximate efficiency?

A recent result by Bleischwitz, Monien, and Schoppmann [2] gives acyclic mechanisms for the above problems that are fully budget-balanced and $O(\log k)$-approximate, but that do not run in polynomial time (unless $P = NP$).

Since acyclicity is only the means to the end of incentive-compatibility, we can ask the same questions for wider classes of mechanisms.

4. Answer questions 1–3 with "acyclic mechanism" replaced by "WGSP mechanism" and by "SP mechanism".

The quest for better performance guarantees could be aided by general proof techniques. For Moulin mechanisms, upper bounding approximate efficiency reduces to upper bounding the summability of the underlying cost-sharing method (see Section 5.2.3) [40]. No such result is known for acyclic mechanisms.

5. Identify conditions under which the approximate efficiency of an acyclic mechanism is characterized, or at least bounded above, by the summability of its cost-sharing method. Or, design an alternative to summability for this purpose.

8.2 General Demand Mechanisms

General demand cost-sharing problems should be studied in much greater depth.

6. Is there a polynomial-time, $O(1)$-budget-balanced acyclic mechanism for metric FTUFL that has reasonable economic efficiency?

7. Is there a general mechanism design technique when marginal valuations can be increasing?

8. What other general demand problems admit good acyclic mechanisms? In particular, connectivity cost-sharing problems—in which each player seeks a prescribed number of disjoint paths in a network between its source and sink vertices—pose a concrete challenge for our techniques.

8.3 Characterizations

Finally, we have few characterizations of cost-sharing mechanisms. Moulin [34] provides characterizations under the assumptions of GSP and full budget-balance. Immorlica, Mahdian, and Mirrokni [21] provide a partial characterization of GSP mechanisms without any budget-balance assumptions.

9. Is there a simple characterization of WGSP mechanisms? To what extent do acyclic mechanisms exhaust the class of WGSP mechanisms? (See Juarez [27] for recent progress on these questions.)

10. Is there a simple characterization of the mechanisms that are implementable as acyclic mechanisms? When does a "natural" primal-dual algorithm induce an acyclic mechanism? Are there general techniques other than the primal-dual method for designing good acyclic mechanisms? (See [2, 5] for recent results along these lines.)

References

[1] 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.

[2] 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.

[3] Y. Bleischwitz and F. Schoppmann. Group-strategyproof cost sharing for metric fault tolerant facility location. In *Proceedings of the First International Symposium on Algorithmic Game Theory (SAGT)*, 2008. To appear.

[4] 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.

[5] J. Brenner and G. Schäfer. Singleton acyclic mechanisms and their applications to scheduling problems. In *Proceedings of the First International Symposium on Algorithmic Game Theory (SAGT)*, 2008. To appear.

[6] 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.

[7] F. Chudak and D. B. Shmoys. Improved algorithms for the uncapacitated facility location problem. *SIAM Journal on Computing*, 33(1):1–25, 2003.

[8] V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[9] V. Chvátal. *Linear Programming*. Freeman, 1983.

[10] 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.

[11] 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)*, 2008. To appear.

[12] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards, Series B*, 71(4):233–240, 1967.

[13] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[14] J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Hardness results for multicast cost sharing. *Theoretical Computer Science*, 304(1-3):215–236, 2003.

[15] M. X. Goemans and M. Skutella. Cooperative facility location games. *Journal of Algorithms*, 50(2):194–214, 2004.

[16] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[17] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.

[18] A. Gupta, J. Könemann, S. Leonardi, R. Ravi, and G. Schäfer. An efficient cost-sharing mechanism for the prize-collecting Steiner forest problem. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1153–1162, 2007.

[19] A. Gupta, A. Srinivasan, and É. Tardos. Cost-sharing mechanisms for network design. *Algorithmica*, 50(1):98–119, 2008.

[20] D. S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22(2):148–162, 1982.

[21] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Limitations of cross-monotonic cost-sharing schemes. *ACM Transactions on Algorithms*, 2008. To appear. Preliminary version in *SODA '05*.

[22] 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.

[23] K. Jain and V. 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.

[24] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001.

[25] K. Jain and V. V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 313–321, 2002. Full version to appear in *SIAM Journal on Computing*.

[26] K. Jain and V. V. Vazirani. An approximation algorithm for the fault tolerant metric facility location problem. *Algorithmica*, 38(3):433–439, 2003.

[27] R. Juarez. Group strategyproof cost sharing: the role of indifferences. Working paper, 2007.

[28] 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.

[29] 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.

[30] J. Könemann, S. Leonardi, and G. Schäfer. A group-strategyproof mechanism for Steiner forests. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 612–619, 2005.

[31] S. Leonardi and G. Schäfer. Cross-monotonic cost-sharing methods for connected facility location. *Theoretical Computer Science*, 326(1-3):431–442, 2004.

[32] M. Mahdian. *Facility Location and the Analysis of Algorithms through Factor-Revealing Programs*. PhD thesis, Massachusetts Institute of Technology, 2004.

[33] M. Mahdian, Y. Ye, and J. Zhang. Approximation algorithms for metric facility location problems. *SIAM Journal on Computing*, 36(2):411–432, 2006.

[34] H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16(2):279–320, 1999.

[35] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: Budget balance versus efficiency. *Economic Theory*, 18(3):511–533, 2001.

[36] 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.

[37] D. C. Parkes and L. H. Ungar. Iterative combinatorial auctions: Theory and practice. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, pages 74–81, 2000.

[38] 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.

[39] S. Rajagopalan and V. V. Vazirani. Primal-dual RNC approximation algorithms for set cover and covering integer programs. *SIAM Journal on Computing*, 28(2):525–540, 1998.

[40] 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. Full version available from `http://theory.stanford.edu/~tim/`.

[41] 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.

[42] D. B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithm for facility location problems. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 265–274, 1997.

[43] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.