Hardness Results for Multicast Cost Sharing*

Joan Feigenbaum† Arvind Krishnamurthy‡ Rahul Sami§ Scott Shenker¶

June 16, 2002

Abstract

We continue the study of *multicast cost sharing* from the viewpoints of both computational complexity and economic mechanism design. We provide fundamental lower bounds on the network complexity of group-strategyproof, budget-balanced mechanisms. We also extend a classical impossibility result in game theory to show that no strategyproof mechanism can be both approximately efficient and approximately budget-balanced. Our results show that one important and natural case of multicast cost sharing is an example (to our knowledge, the first in the literature) of a representative hard problem in distributed, algorithmic mechanism design; in this sense, they represent progress toward the development of a "complexity theory of Internet computation."

Keywords: Algorithmic mechanism design, communication complexity, multicast.

1 Introduction

In the standard *unicast* model of Internet transmission, each packet is sent to a single destination. Although unicast service has great utility and widespread applicability, it cannot efficiently transmit popular content, such as movies or concerts, to a large number of receivers; the source would have to transmit a separate copy of the content to each receiver independently. The *multicast* model of Internet transmission relieves this problem by setting up a shared delivery tree spanning all the receivers; packets sent down this tree are replicated at branch points so that no more than one copy of each packet traverses each link. Multicast thus greatly reduces the transmission costs involved in reaching large user populations.

*This work was supported by the DoD University Research Initiative (URI) program administered by the Office of Naval Research under Grant N00014-01-1-0795.

†Yale University, Computer Science, New Haven, CT 06520-8285 USA, feigenbaum@cs.yale.edu. Supported in part by ONR grants N00014-01-1-0447 and N00014-01-1-0795 and NSF grant CCR-0105337.

‡Yale University, Computer Science, New Haven, CT 06520-8285 USA, arvind@cs.yale.edu. Supported in part by NSF Career grant CCR-9985304.

§Yale University, Computer Science, New Haven, CT 06520-8285 USA, sami@cs.yale.edu. Supported in part by ONR grant N00014-01-1-0795.

¶ICSI, 1947 Center Street, Berkeley, CA 94704-1198 USA, shenker@icsi.berkeley.edu. Supported in part by NSF grants ITR-0081698 and ITR-0121555.

The large-scale, high-bandwidth multicast transmissions required for movies and other potential sources of revenue are likely to incur substantial transmission costs. The costs when using the unicast transmission model are separable in that the total cost of the transmission is merely the sum of the costs of transmission to each receiver. Multicast's use of a shared delivery tree greatly reduces the overall transmission costs, but, because the total cost is now a submodular but nonlinear function of the set of receivers, it is not clear how to share the costs among the receivers. A recent series of papers has addressed the problem of cost sharing for Internet multicast transmissions. In the first paper on the topic, Herzog, Shenker, and Estrin [HSE97], considered axiomatic and implementation aspects of the problem. Subsequently, Moulin and Shenker [MS01] studied the problem from a purely economic point of view. Several more recent papers [FPS01, FKS+01, AR02, FGH+02] adopt the *distributed algorithmic mechanism design* approach, which augments a game-theoretic perspective with distributed computational concerns. In this paper, we extend the results of [FPS01] by considering a more general computational model and approximate solutions. We also extend a classic impossibility [GKL76, Rob79] result by showing that no strategyproof mechanism can be both approximately efficient and approximately budget-balanced.

Before providing a detailed technical statement of our results, we introduce distributed algorithmic mechanism design and our model of multicast cost sharing.

1.1 Distributed Algorithmic Mechanism Design

We consider both complexity and incentive issues inherent in multicast cost sharing. This approach is fairly new, as incentives have rarely been an important consideration in traditional theoretical computer science (TCS). Instead, users (or the computers acting on their behalf) are assumed either to be *obedient* (*i.e.*, to follow the prescribed algorithm) or to be *adversaries* who "play against" each other. The traditional TCS focus is on the design of computationally efficient algorithms and protocols that accomplish the desired aim in the presence of these obedient or adversarial users.

In contrast, the *selfish* users in game theory are neither cooperative nor adversarial. Although one cannot assume that selfish users will obediently follow the prescribed algorithm, one can assume that they will respond to incentives. Thus, one need not design algorithms that achieve correct results in the face of adversarial behavior on the part of some users, but one does need algorithms that work correctly in the presence of predictably selfish behavior by all users. Achieving system-wide goals like Pareto efficiency or fairness in the presence of selfish agents is the primary aim of economic mechanism design (see [J01] for a review). However, this mechanism-design literature has not typically considered the algorithmic efficiency of mechanisms.

Thus, while the economics literature traditionally stressed incentives and downplayed computational complexity, the computer-science literature traditionally did the opposite. The emergence of the Internet as a standard platform for distributed computation has changed this state of affairs. Incentives have become an increasingly important consideration in network-protocol design (see, for example, [FNY89, FS97, HA88, KLO95, KS89, S88, S90, S95]). More recently, the work of Nisan and Ronen [NR01] has inspired the design of *algorithmic mechanisms* for, *e.g.*, scheduling, load balancing, lowest-cost paths, and combinatorial auctions that satisfy both the traditional economic definitions of incentive compatibility and the traditional TCS definitions of efficiency. The examples of *algorithmic mechanism design* in [NR01] use centralized computations and therefore invoke notions of centralized complexity. More recent papers have introduced *distributed algorithmic mechanism design*, in which the computations are distributed and therefore require distributed measures of complexity. Feigenbaum, Papadimitriou, and Shenker [FPS01] propose a general concept of "network complexity" that requires a distributed algorithm executed over an interconnection network T to be modest in four respects: the total number of messages that agents send over T, the maximum number of messages sent over any one link in T, the maximum size of a message, and the local computational burden on agents. If a distributed algorithm requires an excessive expenditure of any one of these four resources, then its "network complexity" is unacceptable. This notion of network complexity allows the mechanism designer to evaluate the feasibility of executing the algorithmic mechanism in a decentralized setting.

A central challenge in the study of algorithmic mechanism design is the search for representative hard problems and, more generally, the development of a full-fledged "complexity theory" of mechanisms. In the Internet-infrastructure problem domain, representative hard problems are those that are hard if they (1) must be solved on a network, (2) must be solved in an incentive-compatible manner, and (3) must be solved in a computationally efficient manner but that become easy if any one of the requirements (1), (2), and (3) is dropped. When combined with earlier work in, *e.g.*, [MS01, FPS01], the results that we present here show that one important and natural case of multicast cost sharing is a representative hard problem; to the best of our knowledge, it is the first such example in the literature.

1.2 Multicast Cost Sharing Model

We use the multicast-transmission model of [FPS01]: There is a user population P residing at a set of network nodes N, which are connected by bidirectional network links L. The multicast flow emanates from a source node $\alpha_s \in N$; given any set of receivers $R \subseteq P$, the transmission flows through a *multicast tree* $T(R) \subseteq L$ rooted at α_s and spans the nodes at which users in R reside. It is assumed that there is a *universal* tree $T(P)$ and that, for each subset $R \subseteq P$, the multicast tree $T(R)$ is merely the minimal subtree of $T(P)$ required to reach the elements in R. This approach is consistent with the design philosophy embedded in essentially all multicast-routing proposals (see, *e.g.,* [BFC93, DEF+96, HC99, PLB+99]).

Each link $l \in L$ has an associated cost $c(l) \geq 0$ that is known by the nodes on each end, and each user i assigns a utility value u_i to receiving the transmission. A *cost-sharing mechanism* determines which users receive the multicast transmission and how much each receiver is charged. We let

$x_i \geq 0$ denote how much user i is charged and σ_i denote whether user i receives the transmission; $\sigma_i = 1$ if the user receives the multicast transmission, and $\sigma_i = 0$ otherwise. We use u to denote the input vector $(u_1, u_2, \ldots, u_{|P|})$ and u_{-i} to denote the vector of all utility values *except* u_i. The mechanism M is then a pair of functions $M(u) = (x(u), \sigma(u))$. The practical feasibility of deploying the mechanism on the Internet depends on the network complexity of computing the functions $x(u)$ and $\sigma(u)$. It is important to note that both the inputs and outputs of these functions are distributed throughout the network; that is, each user inputs his u_i from his network location, and the outputs $x_i(u)$ and $\sigma_i(u)$ must be delivered to him at that location.

The *receiver set* for a given input vector is $R(u) = \{i \mid \sigma_i = 1\}$. A user's individual *welfare* is given by $w_i = \sigma_i u_i - x_i$. The cost of the tree $T(R)$ reaching a set of receivers R is $c(T(R))$, and the overall welfare, or *net worth*, is $NW(R) = u_R - c(T(R))$, where $u_R = \sum_{i \in R} u_i$ and $c(T(R)) = \sum_{l \in T(R)} c(l)$. The overall welfare measures the total benefit of providing the multicast transmission (the sum of the utilities minus the cost).

Our goal is to explore the relationship between incentives and computational complexity, but, before we do so, we first comment on several aspects of the model. The cost model we employ is a poor reflection of reality, in that transmission costs are not per-link; current network-pricing schemes typically only involve usage-based or flat-rate access fees, and the true underlying costs of network usage, though hard to determine, involve small incremental costs (*i.e.*, sending additional packets is essentially free) and large fixed costs (*i.e.*, installing a link is expensive). However, we are not aware of a well-validated alternative cost model, and the per-link cost structure is intuitively appealing, relatively tractable, and widely used.

We assume that the total transmission costs are shared among the receivers. There are certainly cases in which the costs would more naturally be born by the source (*e.g.*, broadcasting an infomercial) or the sharing of costs is not relevant (*e.g.*, a teleconference among participants from the same organization); in such cases, our model would not apply. However, we think that there will be many cases, particularly those involving the widespread dissemination of popular content, in which the costs would be born by the receivers.

There are certainly cases, such as the high-bandwidth broadcast of a long-lived event such as a concert or movie, in which the bandwidth required by the transmission is much greater than that required by a centralized cost-sharing mechanism (*i.e.*, sending all the link costs and utility values to a central site at which the receiver set and cost shares could be computed). For these cases, our feasibility concerns would be moot. However, Internet protocols are designed to be general-purpose; what we address here is the design of a protocol that would share multicast costs for a wide variety of uses, not just long-lived and high-bandwidth events. Thus, we need only claim that there are many scenarios in which our feasibility concerns would be relevant, not that our concerns are relevant in all scenarios.

In comparing the bandwidth required for transmission to the bandwidth required for the cost-sharing mechanism, one much consider several factors. First, and most obvious, is the transmission rate of the application b. For large multicast groups, it will be quite likely that there will be at least one user connected to the Internet by a slow modem. Because the multicast rate must be chosen to accommodate the slowest user, one can't assume that b will be large. Second, the bandwidth consumed on any particular link by the centralized cost sharing mechanisms scales linearly with the number of users $p = |P|$, but the multicast's usage of the link is independent of the number of users. Third, one must consider the time increment Δ over which the cost accounting is done. For some events, such as a movie, it would be appropriate to calculate the cost shares once (at the beginning of the transmission) and not allow users to join after the transmission has started. For other events, such as the transmission of a shuttle mission, users would come and go during the course of the transmission. To share costs accurately in such cases, the time increment Δ must be fairly short. The accounting bandwidth on a single link scales roughly as p, which must be compared to the bandwidth used over a single accounting interval Δb. Although small multicast groups with large Δ and b could easily use a centralized mechanism, large multicast groups with small Δ and b could not.

We have assumed that budget-balanced cost sharing, where the sum of the charges exactly covers the total incurred cost, is the goal of the charging mechanism. If the charging mechanism were being designed by a monopoly network operator, then one might expect the goal to be maximizing revenue. There have been some recent investigations of revenue-maximizing charging schemes for multicast (see, *e.g.*, [FGH+02]), but here we assume, as in [HSE97, MS01, FPS01, FKS+01, AR02], that the charging mechanism is decided by society at large (*e.g.*, through standards bodies) or through competition. Competing network providers could not charge more than their real costs (or otherwise their prices would be undercut) nor less than their real costs (or else they would lose money), and so budget balance is a reasonable goal in such a case. For some applications, such as big-budget movies, the bandwidth costs will be insignificant compared to the cost of the content, and then different charging schemes will be needed, but for low-budget or free content (*e.g.*, teleconferences) budget-balanced cost-sharing is appropriate.

Lastly, in our model it is the *users* who are selfish. The routers (represented by tree nodes), links, and other network-infrastructure components are obedient. Thus, the cost-sharing algorithm does not know the individual utilities u_i, and so users could lie about them, but once they report them to the network infrastructure (*e.g.*, by sending them to the nearest router or accounting node), the algorithms for computing $x(u)$ and $\sigma(u)$ can be reliably executed by the network. Ours is the simplest possible strategic model for the distributed algorithmic mechanism-design problem of multicast cost sharing, but, even in this simplest case, determining the inherent network complexity of the problem is non-trivial. Alternative strategic models (*e.g.*, ones in which the routers are selfish and their strategic goals may be aligned or at odds with those of their resident users) may also present interesting distributed algorithmic mechanism-design challenges. Preliminary work along these lines is reported in [MT02].

1.3 Statement of Results

In order to state our results more precisely, we need additional notation and terminology.

A *strategyproof* cost-sharing mechanism is one that satisfies the property that $w_i(u) \geq w_i(u|^i\mu_i)$, for all u, i, and μ_i. (Here, $(u|^i\mu_i)_j = u_j$, for $j \neq i$, and $(u|^i\mu_i)_i = \mu_i$.) Strategyproofness does not preclude the possibility of a group of users colluding to improve their individual welfares. Any reported utility profile v can be considered a group strategy for the group $S = \{i \mid v_i \neq u_i\}$. A mechanism M is *group-strategyproof* (GSP) if there is no group strategy such that at least one member of the strategizing group improves his welfare while the rest of the members do not reduce their welfare. In other words, if M is GSP, the following property holds for all u, v (where $S = \{i|u_i \neq v_i\}$):

$$\text{either } w_i(v) = w_i(u) \ \forall i \in S$$

$$\text{or } \exists i \in S \text{ such that } w_i(v) < w_i(u).$$

In general, we only consider mechanisms that satisfy four natural requirements[1]:

No Positive Transfers (NPT): $x_i(u) \geq 0$; in other words, the mechanism cannot *pay* receivers to receive the transmission.

Voluntary Participation (VP): $w_i(u) \geq 0$; this implies that $x_i = 0$ whenever $\sigma_i = 0$ and that users are always free to not receive the transmission and not be charged.

Consumer Sovereignty (CS): For all u_{-i}, $\sigma_i = 1$ if u_i is big enough; this implies that the network cannot exclude any user who is willing to pay a sufficiently large amount regardless of other users' utilities.

Symmetry[2] (SYM): If i and j are at the same node or are at different nodes separated by a zero-cost path, and $u_i = u_j$, then $x_i = x_j$.

In addition to these basic requirements, there are certain other desirable properties that one could expect a cost-sharing mechanism to possess. A cost-sharing mechanism is said to be *efficient* if it maximizes the overall welfare, and it is said to be *budget-balanced* if the revenue raised from the receivers covers the cost of the transmission exactly. It is a classical result in game theory [GKL76, Rob79] that a strategyproof cost-sharing mechanism that satisfies NPT, VP, and CS cannot be both budget-balanced and efficient. Moulin and Shenker [MS01] have shown that there is only one strategyproof, efficient mechanism, called *marginal cost* (MC), defined in Section 4 below, that satisfies NPT, VP, and CS. They have also shown that, while there are many GSP, budget-balanced mechanisms that satisfy NPT, VP, and CS, the most natural one to consider is the *Shapley value* (SH), defined in Section 2 below, because it minimizes the worst-case efficiency loss.

[1] The one exception is Section 4, in which we do not assume SYM; that section contains an impossibility result, and so not making this assumption only makes the section stronger.

[2] This straightforward definition is less restrictive than the one given by Moulin and Shenker [MS01]. The SH, JV, and EG mechanisms that we use as examples satisfy the more stringent definition of symmetry in [MS01] as well.

Both MC and SH also satisfy the SYM property. The *egalitarian* (EG) mechanism of Dutta and Ray [DR89] is another well studied GSP, budget-balanced mechanism that satisfies the four basic requirements. Jain and Vazirani [JV01] present a novel family of GSP, approximately budget-balanced mechanisms[3] that satisfy NPT, VP, and CS. Each mechanism in the family is defined by its underlying cost-sharing function, and the resulting mechanism satisfies the SYM property whenever the underlying function satisfies it. We use the notation JV to refer to the members of the Jain-Vazirani family that satisfy SYM.

It is easy to see (and is noted in [FPS01]) that both MC and SH are polynomial-time computable by centralized algorithms. [FPS01] further shows that there is a distributed algorithm that computes MC using only two short messages per link and two simple calculations per node. By contrast, [FPS01] notes that the obvious algorithm that computes SH requires $\Omega(|P| \cdot |N|)$ messages in the worst case and shows that, for a natural class of algorithms (called "linear distributed algorithms"), there is an infinite set of instances with $|P| = O(|N|)$ that require $\Omega(|N|^2)$ messages. Jain and Vazirani [JV01] give centralized, polynomial-time algorithms to compute the approximately budget-balanced mechanisms in the class JV.

In this paper, we show that:

- Any distributed algorithm, deterministic or randomized, that computes a budget-balanced, GSP multicast cost-sharing mechanism must send $\Omega(|P|)$ bits over linearly many links in the worst case. This lower bound applies, in particular, to the SH and EG mechanisms.
- Any distributed algorithm, deterministic or randomized, that computes an approximately budget-balanced, cross-monotonic, GSP multicast cost-sharing mechanism must send $\Omega(\log(|P|))$ bits over linearly many links in the worst case. This lower bound applies, in particular, to the SH, EG, and JV mechanisms.

In order to prove the first of these lower bounds (*i.e.*, the one for exact budget balance), we first prove a lower bound that holds for all mechanisms that correspond to *strictly cross-monotonic* cost-sharing functions. Cross-monotonicity, a technical property defined precisely in Section 2, means roughly that the cost share attributed to any particular receiver cannot increase as the receiver set grows; the cost-sharing functions defined by a broad class of multicast trees are examples of cross-monotonic functions but not the only examples. Our lower bound on the network complexity of strictly cross-monotonic mechanisms may be applicable to problems other than multicast.

Finally, we prove the following generalization of a classical result in game theory [GKL76, Rob79]:

- There is no strategyproof multicast cost-sharing mechanism satisfying NPT, VP, and CS that is both approximately efficient and approximately budget-balanced.

[3]The mechanisms in [JV01] actually satisfy a more stringent definition of approximate budget balance than we use; thus, our network-complexity lower bounds apply to them *a fortiori*.

2 Exact submodular cost sharing

In this section, we prove a basic communication-complexity lower bound that applies to the distributed computation of many submodular cost-sharing mechanisms. We first prove this lower bound for all mechanisms that satisfy "strict cross-monotonicity" as well as the four basic properties discussed in Section 1. We then show that, whenever the underlying cost function is strictly subadditive, the resulting Shapley-value mechanism is strictly cross-monotonic and hence has poor network complexity. Finally, we discuss the special case of multicast cost sharing and describe very general conditions under which the multicast cost will be strictly subadditive. In particular, we present an infinite family of instances that have strictly subadditive costs and show that any cost-sharing mechanism that satisfies the four basic requirements must have poor network complexity on these instances.

Consider the general situation in which we want a mechanism to allow the users to share the cost of a common service. We restrict our attention to the case of binary preferences: User i is either "included," by which he attains utility u_i, or he is "excluded" from the service, giving him 0 utility. A mechanism can use the utility vector u as input to compute a set $R(u)$ of users who receive the service and a payment vector $x(u)$. Further, suppose that the cost of serving a set $S \subseteq P$ of the users is given by $C(S)$. This cost function is called *submodular* if, for all $S, T \subseteq P$, it satisfies:

$$C(S \cup T) + C(S \cap T) \leq C(S) + C(T).$$

Submodularity is often used to model economies of scale, in which the marginal costs decrease as the serviced set grows. One example of a submodular cost function is the one presented in Section 1, where the cost of delivering a multicast to a set R of receivers is the sum of the link costs in the smallest subtree of the universal tree that includes all locations of users in R.

Moulin and Shenker [M99, MS01] have shown that any mechanism for submodular cost sharing that satisfies budget-balance, GSP, VP, and NPT must belong to the class of *cross-monotonic cost-sharing mechanisms.* A mechanism in this class is completely characterized by its set of cost-sharing functions $g = \{g_i : 2^P \to \Re_{\geq 0}\}$. Here $g_i(S)$ is the cost that g attributes to user i if the receiver set is S. For brevity, we will refer to $g = \{g_i\}$ as a "cost-sharing function," rather than a set of cost-sharing functions. We say that g is *cross-monotonic* if, $\forall i \in S, \forall T \subseteq P,\ g_i(S \cup T) \leq g_i(S)$. In addition, we require that $g_i(S) \geq 0$ and, $\forall j \notin S,\ g_j(S) = 0$. Then, the corresponding cross-monotonic mechanism $M_g = (\sigma(u), x(u))$ is defined as follows: The receiver set $R(u)$ is the unique largest set S for which $g_i(S) \leq u_i$, for all i. This is well defined, because, if sets S and T each satisfy this property, then cross-monotonicity implies that $S \cup T$ satisfies it. The cost shares are then set at $x_i(u) = g_i(R(u))$.

There is a natural iterative algorithm to compute a cross-monotonic cost-sharing mechanism [MS01, FPS01]: Start by assuming the receiver set $R^0 = P$, and compute the resulting cost shares $x_i^0 = g_i(R^0)$. Then drop out any user j such that $u_j < x_j^0$; call the set of remaining users R^1.

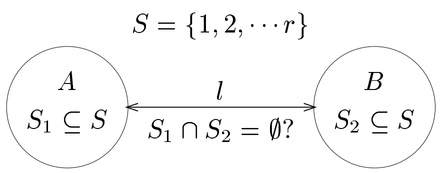

Figure 1: The set disjointness problem

The cost shares of other users may have increased, so we need to compute the new cost shares $x_i^1 = g_i(R^1)$ and iterate. This process ultimately converges, terminating with the receiver set $R(u)$ and the final cost shares $x_i(u)$.

Now, we consider a subclass of the cross-monotonic mechanisms:

Definition 1 *A cross-monotonic cost-sharing function* $g = \{g_i : 2^P \to \Re_{\geq 0}\}$ *is called* **strictly cross-monotonic** *if, for all* $S \subset P, i \in S,$ *and* $j \notin S$, $g_i(S \cup j) < g_i(S)$. *The corresponding mechanism* M_g *is called a strictly cross-monotonic mechanism.*

We now prove a lower bound on the communication complexity of strictly cross-monotonic cost-sharing mechanisms. Our proof is a reduction from the *set disjointness* problem: Consider a network consisting of two nodes A and B, separated by a link l (see *Figure 1*). Node A has a set $S_1 \subseteq \{1, 2, \ldots, r\}$, node B has another set $S_2 \subseteq \{1, 2, \ldots, r\}$, and one must determine whether the sets S_1 and S_2 are disjoint. It is known that any deterministic or randomized algorithm to solve this problem must send $\Omega(r)$ bits between A and B. (Proofs of this and other basic results in communication complexity can be found in the book by Kushilevitz and Nisan [KN97].)

Theorem 1 *Suppose* M_g *is a strictly cross-monotonic mechanism corresponding to a cost-sharing function* g *and satisfying VP, CS, and NPT. Further, suppose that the mechanism must be computed in a network in which a link (or set of links)* l *is a cut and there are* $\Omega(|P|)$ *users on each side of* l*. Then, any deterministic or randomized algorithm to compute* M_g *must send* $\Omega(|P|)$ *bits across* l *in the worst case.*

Proof: For simplicity, assume that the network consists of two nodes A and B connected by one link l and that there are $r = |P|/2$ users at each of the two nodes. (The proof of the more general case is identical.) Arbitrarily order the users at each node. We can now call the users $a_1, a_2, \ldots, a_r$ and $b_1, b_2, \ldots, b_r$. Because the mechanism M_g is strictly cross-monotonic, we can find a real value $d > 0$ such that, for all $S \subset P, i \in S, j \notin S$,

$$g_i(S \cup j) < g_i(S) - d$$

For each user $i \in P$, we will define two possible utility values t_i^L and t_i^H as follows:

- First, the values for a_1 and b_1 are

$$t_{a_1}^H = g_{a_1}(\{a_1, b_1\}), \qquad t_{a_1}^L = t_{a_1}^H - d$$

$$t_{b_1}^H = g_{b_1}(\{a_1, b_1\}), \qquad t_{b_1}^L = t_{b_1}^H - d$$

- Similarly, the values for a_k and b_k are

$$t_{a_k}^H = g_{a_k}(\{a_1, a_2, \ldots, a_k, b_1, b_2, \ldots, b_k\}), \qquad t_{a_k}^L = t_{a_k}^H - d$$

$$t_{b_k}^H = g_{b_k}(\{a_1, a_2, \ldots, a_k, b_1, b_2, \ldots, b_k\}), \qquad t_{b_k}^L = t_{b_k}^H - d$$

Now, we show how to reduce from the set disjointness problem to the mechanism M_g. Node A gets a subset $S_1 \subseteq \{1, \ldots, r\}$ and constructs a utility vector u for the users at A, defined by

$$\forall i \in S_1 \;\; u_{a_i} = t_{a_i}^H$$

$$\forall i \notin S_1 \;\; u_{a_i} = t_{a_i}^L$$

Similarly, node B is given set S_2 and constructs a utility vector v for the users at B, defined by

$$\forall i \in S_2 \;\; v_{b_i} = t_{b_i}^H$$

$$\forall i \notin S_2 \;\; v_{b_i} = t_{b_i}^L$$

They now run mechanism M_g on input (u, v) and check whether the receiver set $R_g(u, v)$ is empty.

Claim: $R_g(u, v)$ is empty iff S_1 and S_2 are disjoint.

Proof of claim: To show the "if" direction, we can simulate the iterative algorithm to compute the receiver set. We start with $R = P$. Then, because S_1 and S_2 are disjoint, either $r \notin S_1$ or $r \notin S_2$. Assume, without loss of generality, that $r \notin S_1$. Now, $u_{a_r} = t_{a_r}^L < g_{a_r}(R)$, and hence a_r must drop out of the receiver set R. But now, because of strict cross-monotonicity, it follows that $g_{b_r}(P - \{a_r\}) > g_{b_r}(P) + d > t_{b_r}^H$, and so b_r must also drop out of the receiver set. Repeating this argument for $r-1, r-2, \ldots, 1$, we can show that the receiver set must be empty.

To show the "only if" direction, assume that $i \in S_1 \cap S_2$. Then, let $T = \{a_1, \ldots, a_i, b_1, \ldots, b_i\}$. $u_{a_i} = t_{a_i}^H = g_{a_i}(T)$, and $v_{a_i} = t_{b_i}^H = g_{b_i}(T)$. Further, for all $j < i$, it follows from strict cross-monotonicity that $g_{a_j}(T) < t_{a_j}^L \leq u_{a_j}$, and $g_{b_j}(T) < t_{b_j}^L \leq v_{b_j}(T)$. Thus, the receiver set $R_g(u, v) \supseteq T$, and hence it is nonempty. □

Theorem 1 follows from this claim and the communication complexity of set disjointness. □

2.1 Strictly Subadditive Cost Functions

In this section, we show that, for a class of submodular cost functions, the Shapley-value mechanism (which is perhaps the most compelling mechanism from an economic point of view) is strictly cross-monotonic and hence has poor network complexity. We also show that this is not a property peculiar to the Shapley-value mechanism alone; for these cost functions, the poor network complexity holds for a large class of mechanisms.

Theorem 1 provides a sufficient condition, strict cross-monotonicity, for a mechanism to have poor network complexity. However, for some submodular cost functions, it is possible that no mechanism satisfies this condition: If the costs are additive, *i.e.*, if the cost of serving a set S is exactly the sum of the costs of serving each of its members individually, then there is a unique mechanism satisfying the basic properties. This mechanism is defined by:

$$R(u) = \{i | u_i \geq C(\{i\})\}$$

$$x_i(u) = C(\{i\}) \text{ if } i \in R(u), \text{ and } x_i(u) = 0 \text{ otherwise.}$$

This mechanism is very easy to compute, either centrally or in a distributed manner, because there is no interaction among the users' utilities; in essence, we have $|P|$ independent local computations to perform.

We need to exclude these trivial cost functions in order to prove general lower bounds for a class of submodular functions. This leads us to consider submodular cost functions that are *strictly subadditive*:

$$\forall S \subseteq P, S \neq \emptyset, \ \ \forall i \in P, \ \ C(S \cup \{i\}) < C(S) + C(\{i\})$$

For a given cost function C, there may be many $g = \{g_i\}$ for which the corresponding mechanism M_g satisfies the basic properties NPT, VP, CS, and SYM. However, Moulin and Shenker [M99, MS01] have shown that, for any given submodular cost function, the cross-monotonic mechanism that minimizes the worst-case efficiency loss is the Shapley-value mechanism (SH). This is a cross-monotonic cost-sharing mechanism corresponding to a function g^{SH}, defined by:

$$\forall S \subseteq P \quad \forall i \in S, \qquad g_i^{SH}(S) = \sum_{R \subseteq S - \{i\}} \frac{|R|!(|S| - |R| - 1)!}{|S|!} [C(R \cup \{i\}) - C(R)] \tag{1}$$

The SH mechanism is therefore a natural mechanism to choose for submodular cost sharing. The following lemma shows that this mechanism has poor network complexity.

Lemma 1 *The Shapley-value mechanism for a strictly subadditive cost function is strictly cross-monotonic.*

Proof: We need to show that, for all sets S, for all $i \in S, j \notin S$, $g_i^{SH}(S \cup \{j\}) < g_i^{SH}(S)$. The proof follows directly from the definition of $g_i^{SH}(S)$ in Equation 1. We use $\mathrm{MC}_i(R)$ to denote $[C(R \cup \{i\}) - C(R)]$, the *marginal cost* of serving i in set R. Consider a set $S \subseteq P - \{j\}$ and a user $i \in S$. Let $r = |R|, s = |S|$.

$$\begin{aligned}
g_i^{SH}(S \cup \{j\}) &= \sum_{R \subseteq S \cup \{j\} - \{i\}} \frac{r!(s-r)!}{(s+1)!} \mathrm{MC}_i(R) \\
&= \sum_{R \subseteq S - \{i\}} \left[\frac{r!(s-r)!}{(s+1)!} \mathrm{MC}_i(R) + \frac{(r+1)!(s-r-1)!}{(s+1)!} \mathrm{MC}_i(R \cup \{j\}) \right] \\
&= \sum_{R \subseteq S - \{i\}} \frac{r!(s-r-1)!}{s!} \left[\frac{s-r}{s+1} \mathrm{MC}_i(R) + \frac{r+1}{s+1} \mathrm{MC}_i(R \cup \{j\}) \right] \qquad (2)
\end{aligned}$$

It follows from submodularity of costs that, for all R, $\mathrm{MC}_i(R \cup \{j\}) \leq \mathrm{MC}_i(R)$. Further, strict subadditivity implies that, for $R = \emptyset$, $\mathrm{MC}_i(R \cup \{j\}) < \mathrm{MC}_i(R)$. Thus, Equation 2 yields

$$\begin{aligned} g_i^{SH}(S \cup \{j\}) &< \sum_{R \subseteq S - \{i\}} \frac{r!(s-r-1)!}{s!} \mathrm{MC}_i(R) \\ g_i^{SH}(S \cup \{j\}) &< g_i^{SH}(S) \end{aligned}$$

□

Corollary 1 *For a strictly subadditive cost function, any algorithm (deterministic or randomized) that computes the SH mechanism in a network must communicate $\Omega(|P|)$ bits across any cut that has $\Theta(|P|)$ users on each side of the cut.* □

Note that the network may consist of a root node α_s with no resident users, a node A with $\frac{|P|}{2}$ resident users, another node B with $\frac{|P|}{2}$ resident users, and a path from A to B consisting of $|N| - 3$ nodes, each with no resident users. Each link in the path from A to B is a cut with $\Theta(|P|)$ users on each side, and thus $\Omega(|P|)$ bits must be sent across linearly many links. In what follows, we call these the *path instances*.

2.2 Multicast cost sharing

We now return to the special case of multicast cost sharing. Recall that the cost function associated with an instance of the multicast cost-sharing problem is determined by the structure of the universal multicast tree T, the link costs, and the locations of the users in the tree; so the cost $C(S)$ of serving user set $S \subseteq P$ is $\sum_{l \in T(S)} c(l)$, where $T(S)$ is the smallest subtree of T that includes all nodes at which users in S reside. It is not hard to show that there are many instances that give rise to strictly subadditive functions C. In fact, we have the following lemma:

Lemma 2 *Consider any instance of multicast cost sharing in which, for any two potential receivers i and j, there exists a link $l \in T(\{i\}) \cap T(\{j\})$ such that $c(l) > 0$. The cost function associated with this instance is strictly subadditive.*

Proof: Given $S \subseteq P - i$, pick any $j \in S$. Let l be any link in $T(\{j\}) \cap T(\{i\})$ with $c(l) > 0$. Then, $C(S \cup \{i\}) \leq C(S) + C(\{i\}) - c(l) < C(S) + C(\{i\})$. □

For example, whenever the source of the multicast has only one link from it, and this link has non-zero cost, the associated cost function is strictly subadditive. One such family of instances (parametrized by $p = |P|$) is shown in *Figure 2*. There are three nodes, α_s, A, and B, and p users. There are $(p/2)$ users at each of B and A, with utilities $u_1, u_2, \ldots, u_{\frac{p}{2}}$ and $v_1, v_2, \ldots, v_{\frac{p}{2}}$, respectively. The link l_1 between α_s and A costs C, while the link l_2 between A and B is free (has 0 cost).

It follows immediately from Corollary 1 that the Shapley-value mechanism for this tree requires $\Omega(|P|)$ bits of communication across link l. In addition, we now show that any mechanism that

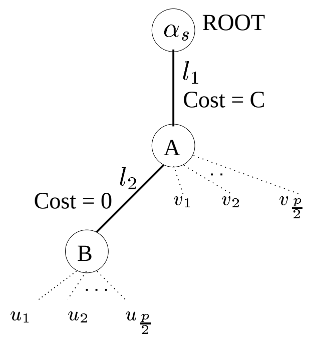

Figure 2: Multicast tree with strictly subadditive costs

satisfies the basic properties must be identical to the SH mechanism on instances of the type shown in *Figure 2*; thus, the lower bound extends to all such mechanisms.

Lemma 3 *Consider multicast cost-sharing instances of the type shown in* Figure 2. *Let M_g be a cross-monotonic cost-sharing mechanism that satisfies SYM, corresponding to a cost-sharing function $g = \{g_i\}$. Then, g (and M_g) are completely determined on these instances by*

$$\forall S \subseteq P, \forall i \in S \quad g_i(S) = \frac{C}{|S|}$$

Proof: Because the mechanism satisfies SYM, we must have $g_i(S) = g_j(S)$ for any receiver set S and any pair i, j in this set. The budget-balance requirement then forces $g_i(S) = \frac{C}{|S|}$ for all i in set S. □

It follows from Lemma 3 that any such mechanism must be strictly cross-monotonic on this family of instances. Thus, Theorem 1 and Lemma 3 imply the following lower bound for multicast cost sharing. The worst-case instances include the path instances defined in Section 2.1, with cost C on the link from α_s to A and cost 0 on all the other links; these instances are identical to the ones in *Figure 2*, except that they contain a zero-cost path of length $|N| - 2$ from A to B instead of a single zero-cost link.

Theorem 2 *Any distributed algorithm, deterministic or randomized, that computes a budget-balanced, GSP multicast cost-sharing mechanism exactly must send $\Omega(|P|)$ bits over linearly many links in the worst case.* □

Note that this lower bound applies to the EG mechanism for multicast cost-sharing defined in Section 1.

3 Network complexity of approximately budget-balanced mechanisms

In view of the lower bounds presented in Section 2, it is natural to ask whether one can *approximate* a budget-balanced, GSP mechanism in a communication-efficient manner. In this case, we do not have a clean analogue of Corollary 1. However, for the special case of multicast cost sharing, we can still prove a result similar to Theorem 2 that provides a lower bound on the network complexity of approximations that satisfy the cross-monotonicity property.

First, we recall the definition of an "approximately budget-balanced" mechanism. As explained at length in, *e.g.*, [NR00, FKS+01], one cannot define an approximation of a cost-sharing mechanism (σ, x) simply as a pair (σ', x') such that σ' and x' approximate σ and x, respectively, *as functions*. Such an approach may destroy the game-theoretic properties of (σ, x), *e.g.*, the resulting "mechanism" (σ', x') may not be strategyproof! For our purposes in this section, a *κ-approximately budget-balanced mechanism*, where $\kappa > 1$ is a constant, is a mechanism (σ, x) with the following properties: GSP, VP, NPT, CS, SYM, and

$$\forall c(\cdot), T(P), \text{ and } u : (1/\kappa) \cdot c(T(R(u))) \leq \sum_{i \in R(u)} x_i(u) \leq \kappa \cdot c(T(R(u))).$$

An approximation to a specific budget-balanced mechanism such as SH or EG would have to satisfy at least one additional (non-strategic) condition. For example, because SH is the GSP, budget-balanced mechanism that minimizes worst-case efficiency loss, an approximation to SH would have to come within a constant factor of SH's efficiency loss in the worst case.

Our network-complexity lower bound on κ-approximations applies only to the family of cross-monotonic cost-sharing mechanisms. In contrast to the case of exactly budget-balanced mechanisms discussed earlier, not all approximately budget-balanced GSP mechanisms are cross-monotonic. However, this is still a large class of mechanisms, which includes the approximate mechanisms described in [FKS+01] and [JV01].

We now extend the lower-bound technique of the previous section so that it applies to cross-monotonic, κ-approximately budget-balanced mechanisms, when κ is upper-bounded away from $\sqrt{2}$, *i.e.*, when $\kappa \leq \sqrt{2} - \epsilon$, for some fixed $\epsilon > 0$. We want to reduce from set-disjointness as before. We again consider the multicast tree shown in *Figure 2* with $(p/2)$ users at each of B and A, with utilities $u_1, u_2, \ldots, u_{\frac{p}{2}}$ and $v_1, v_2, \ldots, v_{\frac{p}{2}}$, respectively.

It is more convenient to work with an alternative representation of the input utility vectors. We use only a restricted set of the possible utility vectors u and v, where (u, v) satisfies the following conditions:

- Let $\beta = \kappa^2/(2 - \kappa^2)$. Restrict the set of allowable utilities to $T = \{0, t_{\beta,1}, \ldots, t_{\beta,r}\}$, where

$$t_{\beta,i} = \frac{\kappa C}{2\beta^{i+1}} \; .$$

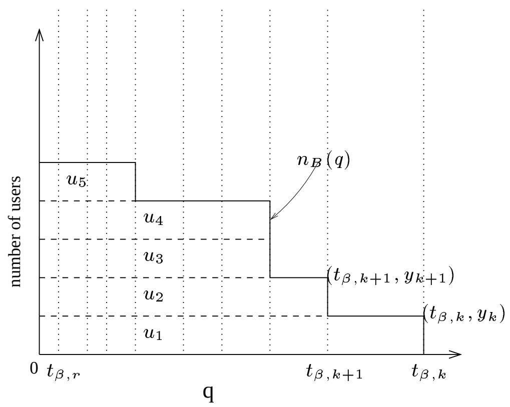

Figure 3: Graphical representation of u

- Each of u and v is (internally) sorted, *i.e.*, $i < j \Rightarrow u_i \geq u_j$ and $v_i \geq v_j$. There is no restriction on the relationship between u_i and v_j.

Consider node B. Define $n_B(q)$ to be the number of users at node B who have utility at least q. Let $y = (y_1, y_2, \ldots, y_r)$, where $y_i = n_B(t_{\beta,i})$. Similarly, let $n_A(q)$ be the number of users at A with utility at least q; let $z = (z_1, z_2, \ldots, z_r)$, where $z_i = n_A(t_{\beta,i})$.

For this class of utility profiles, there is a one-to-one mapping between values of u and y. To see this, look at *Figure 3*, a graph of $n_B(q)$ against q. Because u is sorted, this monotonically decreasing function completely defines the utility vector; u_1 must be the largest q for which $n_B(q) > 0$, and so on. The function $n_B(q)$ is completely defined by its "corners" and hence by y. Furthermore, by definition, there is a unique y for any u. A similar correspondence holds for v and z.

We first prove a useful lemma about approximately budget-balanced mechanisms on this class of instances.

Lemma 4 *Let M_g be a cross-monotonic, symmetric, κ-approximately budget-balanced mechanism, and consider instances of the type shown in* Figure 2. *Let vectors y, z be defined corresponding to the utility profiles u and v, as described above. Then, M_g satisfies the following two properties:*

(i). If there is an i such that $(y_i + z_i) \geq 2\beta^{i+1}$, then mechanism M_g will compute a non-empty receiver set on this instance.

(ii). If there is no i such that $(y_i + z_i) \geq (\beta^i + \beta^{i+1})$, then mechanism M_g will compute an empty receiver set on this instance.

Proof: Because the mechanism M_g satisfies SYM, $g_i(S) = g_j(S)$, for all $i, j \in S$. Thus, the revenue raised from receiver set S is exactly $g_i(S)|S|$, for any $i \in S$. The approximate budget-balance condition then yields the inequalities

$$\frac{C}{\kappa|S|} \leq g_i(S) \leq \frac{\kappa C}{|S|} \ .$$

Now, if $|S| \geq 2\beta^{i+1}$, it follows that $g_i(S) \leq t_{\beta,i}$, and hence property (i) must hold. Similarly, because $\frac{1+\beta}{2\beta} = \frac{1}{\kappa^2}$, property (ii) must hold as well. □

We now use Lemma 4 to provide a reduction from the set disjointness problem as follows. As before, node A has a set $S_1 \subseteq \{1, 2, \ldots, r\}$, node B has another set $S_2 \subseteq \{1, 2, \ldots, r\}$, and one must determine whether the sets S_1 and S_2 are disjoint. We must make sure that, if $S_1 \cap S_2 \neq \emptyset$, there is a set of receivers who can share κC, and, if $S_1 \cap S_2 = \emptyset$, there is no set of receivers who can share even C/κ. For this, we construct the vectors y and z using the rules:

$$\begin{aligned} y_i &= \lceil \beta^{i+1} \rceil \quad \text{if } i \in S_2 \\ y_i &= \lceil \beta^{i} \rceil \quad \text{otherwise} \\ z_i &= \lceil \beta^{i+1} \rceil \quad \text{if } i \in S_1 \\ z_i &= \lceil \beta^{i} \rceil \quad \text{otherwise} \end{aligned}$$

These are valid input vectors, because $y_i \leq y_{i+1}$ and similarly for z. If $i \in S_1 \cap S_2$, then $y_i + z_i \geq 2\beta^{i+1}$, and so there is transmission. If S_1 and S_2 are disjoint, then, for all i,

$$y_i + z_i \leq \beta^i + \beta^{i+1} + 2 \ .$$

Here, the 2 in the right-hand side arises because of the ceiling terms; it can be ignored, because we can always make a small upward adjustment to β. This means that, in the target instance,

$$(y_i + z_i) t_{\beta,i} \leq \frac{C}{\kappa} \ ,$$

and consequently there is no transmission.

Thus, we can use this κ-approximate mechanism to solve the set-disjointness problem, and this implies that the mechanism must use $\Omega(r)$ bits of communication, where

$$r = \Theta\left(\frac{\log p}{\log \beta}\right) = \Theta\left(\frac{\log p}{2\log\kappa - \log(2-\kappa^2)}\right) .$$

If we require κ to be upper-bounded away from $\sqrt{2}$, then the right-hand side is $\Theta(\frac{\log p}{\log \kappa})$. Thus, we have:

Theorem 3 *Any distributed algorithm, deterministic or randomized, that computes a κ-approximately budget-balanced, cross-monotonic, GSP multicast cost-sharing mechanism, where $\kappa \leq \sqrt{2} - \epsilon$, for some fixed $\epsilon > 0$, must send $\Omega(\frac{\log |P|}{\log \kappa})$ bits of communication over linearly many links in the worst case.*

Once again, the worst-case instances include the path instances defined in Section 2.

4 An impossibility result for approximate budget-balance and approximate efficiency

As stated in Section 1, it is a classical result in game theory that no strategyproof cost-sharing mechanism can be both budget-balanced and efficient [GKL76, Rob79]. We now consider whether this fundamental impossibility result holds when the budget-balance and efficiency considerations are replaced by their approximate counterparts. In this section, we do not assume that the cost-sharing mechanisms have the SYM property; the impossibility result that we present here does not require this assumption. Furthermore, in this section, we require only that a κ-approximately budget-balanced mechanism be strategyproof, not GSP as in Section 3.

We first review the definition of the MC mechanism, which was shown by Moulin and Shenker [MS01] to be the only efficient mechanism that satisfies VP, NPT, and CS. Given an input utility profile u, the MC receiver set is the unique *largest efficient set* of users. To compute it, as shown in [FPS01], one recursively computes the *welfare* (also known as *net worth* or *efficiency*) of each node $\beta \in N$:

$$W(\beta) = \left(\sum_{\substack{\gamma \in Ch(\beta) \\ W(\gamma) \geq 0}} W(\gamma) \right) - c(l),$$

where $Ch(\beta)$ is the set of children of β in the tree, and $c(l)$ is the cost of the link connecting β to its parent node. Then, the largest efficient set $R(u)$ is the set of all users i such that every node on the path from i to the root α_S has nonnegative welfare. The *total efficiency* is $\mathrm{NW}(R(u)) = W(\alpha_S)$.

Another way to view this is as follows: The algorithm partitions the universal tree $T(P)$ into a forest $F(u) = \{T_1(u), T_2(u), \ldots, T_k(u)\}$. A link from $T(P)$ is included in the forest if and only if the child node has nonnegative welfare. $R(u)$ is then the set of users at nodes in the subtree $T_1(u)$ containing the root.

Once $F(u)$ has been computed, for each user i, define $X(i,u)$ to be the node with minimum welfare value in the path from i to its root in its partition. Then, the cost share $x_i(u)$ of user i is defined as

$$\begin{aligned} x_i(u) = &\quad \max(0, u_i - W(X(i,u))) && \forall i \in R(u) \\ x_i(u) = &\quad 0 && \forall i \notin R(u) \end{aligned}$$

If multiple nodes on the path have the same welfare value, we let $X(i,u)$ be the one nearest to i.

By a γ-*approximately efficient mechanism*, where $0 < \gamma < 1$, we mean one that always achieves total efficiency that is at least γ times the total efficiency achieved by MC.

We now ask whether there is any strategyproof mechanism that satisfies the basic requirements of NPT, VP, and CS and is both approximately efficient and approximately budget-balanced.

Consider the tree in *Figure 4*. There are p users, each with utility C/p resident at a node A that is separated from the root node by a link of cost $((p-1)C/p) + \delta$. It is easy to see that this instance of multicast cost-sharing displays the following properties.

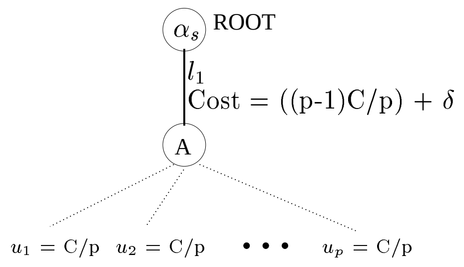

Figure 4: An example of a multicast tree that fails to achieve approximate efficiency and approximate budget-balance.

Property 1 *Any γ-approximately efficient mechanism must transmit to all p users if $0 < \delta < C/p$.*

Property 2 *Any γ-approximately efficiency mechanism must transmit to all p users even if one user, say i, lowers his utility to $\delta + \epsilon$, as long as $\delta + \epsilon < C/p$.*

Property 3 *Any γ-approximately efficient, strategyproof mechanism that satisfies VP assigns to each user a cost share of at most δ.*

If the cost share $x_i(u)$ were greater than δ, the user i could strategize by claiming that his utility was $v_i = \delta + \epsilon < x_i(u)$. By VP and the requirement of γ-approximate efficiency, the mechanism would need to include user i and assign him a cost share $x_i(u|^i v_i) \leq v_i < x_i(u)$, which would imply a violation of strategyproofness.

Therefore, the revenue collected by a strategyproof mechanism that achieves γ-approximate efficiency is bounded by $p\delta$.

Property 4 *A γ-approximately efficient, strategyproof mechanism cannot be κ-approximately budget-balanced if $\delta < (C(p-1))/(p(\kappa p - 1))$.*

In summary, we have:

Theorem 4 *A strategyproof mechanism for multicast cost sharing that satisfies the basic requirements of NPT, VP, and CS cannot achieve both γ-approximate efficiency and κ-approximate budget-balance for any pair of constants κ and γ.*

5 Conclusions and open problems

As we explained in Section 1, one central challenge in the study of algorithmic mechanism design is the search for representative hard problems. In the network-infrastructure problem domain, a problem may be considered "hard" if it (1) must be solved on a network, (2) must be solved in an incentive-compatible manner, and (3) must be solved in a computationally efficient manner but becomes easy if any one of (1), (2), and (3) is dropped. Our results show that, under the basic requirements of NPT, VP, CS, and SYM, exact computation of group-strategyproof budget-balanced multicast cost shares is such a problem. Theorem 2 shows that the problem is hard if (1), (2) and (3) are required. If (3) is dropped, the problem is easy: Just use the SH mechanism. If (1) is dropped, it is also easy: Use the natural (centralized) polynomial-time algorithm explained in Section 2 above. Finally, if (2) is dropped, it is easy as well: Just do one bottom-up pass of $T(P)$, computing $U = \sum_{i \in P} u_i$ and $C = \sum_{l \in L} c(l)$. If $C > U$, no one receives the transmission, and the mechanism does one top-down pass to inform all members of P that this is the outcome; if $C \leq U$, everyone receives the transmission, and the mechanism does one top-down pass to communicate the cost share $(u_i \cdot C)/U$ to user i, for all $i \in P$.

The most important general direction for future work is the search for more good distributed algorithmic mechanisms and for more representative hard problems.

More specifically, we ask whether the lower bound in Section 3 is tight. The "scaled step-function" (SSF) mechanism considered in [FKS+01] gives some hope that it is, but the bounds on budget-imbalance and efficiency-loss for SSF are not constant factors.

References

[AR02] Adler, M. and Rubenstein, D. (2002). "Pricing Multicast in More Practical Network Models," in *Proceedings of the 13th Symposium on Discrete Algorithms*, ACM Press/SIAM, pp. 981–990, New York/Philadelphia.

[BFC93] Ballardie, A., Francis, P., and Crowcroft, J. (1993). "Core Based Trees (CBT)," in *Proceedings of Sigcomm '93*, pp. 85–95, ACM Press, New York.

[DEF+96] Deering, S., Estrin, D., Farinacci, D., Jacobson, V., Liu, C., and Wei, L. (1996). "The PIM architecture for wide-area multicast routing," *ACM/IEEE Transactions on Networking* **54**, pp. 153–162.

[DR89] Dutta, B. and Ray, D. (1989). "A concept of egalitarianism under participation constraints," *Econometrica* **57**, pp. 615-635.

[FGH+02] Fiat, A., Goldberg, A., Hartline, J., and Karlin, A. (2002). "Competitive Generalized Auctions," in *Proceedings of the 34th Symposium on the Theory of Computing*, pp. 72–81, ACM Press, New York.

[FKS+01] Feigenbaum, J., Krishnamurthy, A., Sami, R., and Shenker, S. (2001). "Approximation and Collusion in Multicast Cost Sharing," submitted. Available in preprint form at `http://www.cs.yale.edu/homes/jf/FKSS.pdf`.

[FNY89] Ferguson, D., Nikolaou, C., and Yemini, Y. (1989). "An economy for flow control in computer networks," in *Proceedings of the 8th Infocom*, pp. 100–118, IEEE Computer Society Press, Los Alamitos.

[FPS01] Feigenbaum, J., Papadimitriou, C., and Shenker, S. (2001). "Sharing the cost of multicast transmissions," *Journal of Computer and System Sciences* **63**, pp. 21–41.

[FS97] Friedman, E., and Shenker, S. (1997). "Learning and Implementation in the Internet," preprint. Available at `http://www.icir.org/shenker/decent.ps`

[GKL76] Green, J., Kohlberg, E., and Laffont, J. (1976). "Partial equilibrium approach to the free rider problem," *Journal of Public Economics* **6**, pp. 375–394.

[HA88] Hsiao, M.-T. and Lazar, A. (1988). "A game theoretic approach to decentralized flow control of markovian queueing networks," in *Proceedings of Performance '87*, pp. 55–74, North-Holland, Amsterdam.

[HC99] Holbrook, H. and Cheriton, D. (1999). "IP multicast channels: EXPRESS support for large-scale single-source applications," in *Proceedings of Sigcomm '99*, pp. 65–78, ACM Press, New York.

[HSE97] Herzog, S., Shenker, S., and Estrin, D. (1997). "Sharing the 'cost' of multicast trees: An axiomatic analysis," *IEEE/ACM Transactions on Networking* **5**, pp. 847–860.

[J01] Jackson, M. (2001). "A Crash Course in Implementation Theory," *Social Choice and Welfare* **18**, pp. 655–708.

[JV01] Jain, K. and Vazirani, V. (2001). "Applications of approximation to cooperative games," in *Proceedings of the 33rd Symposium on the Theory of Computing*, pp. 364–372, ACM Press, New York.

[KLO95] Korilis, Y., Lazar, A., and Orda, A. (1995). "Architecting noncooperative networks," *Journal on Selected Areas in Communications* **13**, 1241–1251.

[KN97] Kushilevitz, E. and Nisan, N. (1997). **Communication Complexity**, Cambridge University Press, Cambridge.

[KS89] Kurose, J. F. and Simha, R. (1989). "A microeconomic approach to optimal resource allocation in distributed computer systems," *IEEE Transactions on Computers* **38**, 705–717.

[M99] Moulin, H. (1999). "Incremental cost sharing: characterization by strategyproofness," *Social Choice and Welfare* **16**, pp. 279–320.

[MS01] Moulin, H. and Shenker, S. (2001). "Strategyproof Sharing of Submodular Costs: Budget Balance Versus Efficiency," *Economic Theory* **18**, pp. 511–533.

[MT02] Mitchell, J. and Teague, V. (2002). Private communication.

[NR00] Nisan, N. and Ronen, A. (2000). "Computationally feasible VCG mechanisms," in *Proceedings of the 2nd Conference on Electronic Commerce (EC-00)*, pp. 242–252, ACM Press, New York.

[NR01] Nisan, N. and Ronen, A. (2001). "Algorithmic mechanism design," *Games and Economic Behavior* **35**, pp. 166–196.

[PLB+99] Perlman, R., Lee, C.-Y., Ballardie, A., Crowcroft., J., Wang, Z., Maufer, T., Diot, C., and Green, M. (1999). "Simple multicast: A design for simple low-overhead multicast," IETF Internet Draft (Work in Progress).

[Rob79] Roberts, K. (1979). "The Characterization of Implementable Choice Rules," in **Aggregation and Revelation of Preferences**, pp. 321–348, North-Holland, Amsterdam.

[S88] Sanders, B. (1988). "An incentive compatible flow control algorithm for rate allocation in computer networks," *IEEE Transactions on Computers* **37**, 1067–1072.

[S90] Shenker, S. (1990). "Efficient network allocations with selfish users," in *Proceedings of Performance '90*, pp. 279–285, North-Holland, Amsterdam.

[S95] Shenker, S. (1995). "Making greed work in networks," *ACM/IEEE Transactions on Networking* **3**, 819–831.