Approximation and Collusion in Multicast Cost Sharing

(Abstract) *

J. Feigenbaum† A. Krishnamurthy‡ R. Sami§
Yale University
Computer Science Department
New Haven, CT 06520-8285 USA
{jf, arvind, sami}@cs.yale.edu

S. Shenker¶
ACIRI/ICSI
1947 Center Street
Berkeley, CA 94704-1198 USA
shenker@icsi.berkeley.edu

1. INTRODUCTION

Despite their prominent role in some of the more applied areas of computer science, incentives have rarely been an important consideration in traditional algorithm design where, typically, users are assumed either to be *cooperative* (*i.e.*, to follow the prescribed algorithm) or to be *adversaries* who "play against" each other. In contrast, the *selfish* users in game theory are neither cooperative nor adversarial. Although one cannot assume that selfish users will 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 byzantine behavior on the part of some users, but one does need algorithms that work correctly in the presence of predictably selfish behavior. This type of "correctness" is a primary goal of economic mechanism design, but standard notions of algorithmic efficiency are not.

In short, the economics literature traditionally stressed incentives and downplayed computational complexity, and the theoretical 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. In particular, the work of Nisan and Ronen [4] inspired the design of algorithms for a range of problems, including scheduling, load balancing, shortest paths, and combinatorial auctions, that satisfy both the traditional economic definitions of incentive compatibility and the traditional computer-science definitions of efficiency.

One of the problems that has been studied is *multicast cost sharing*, and we continue the study here. Recent work in economics [3] leads naturally to the consideration of two mechanisms: *marginal cost* (MC), which is efficient and strategyproof, and *Shapley value* (SH), which is budget-balanced and group-strategyproof and, among all mechanisms with these two properties, minimizes the worst-case efficiency loss. Subsequent work in computer science [2] shows that the MC mechanism can be computed by a simple, distributed algorithm that uses only two messages per link of the multicast tree but that, for a broad class of algorithms, computing the SH mechanism requires, in the worst case, a number of messages that is quadratic in the size of the multicast tree.

Here, we extend these results in two directions. First, we give a group-strategyproof mechanism that exhibits a tradeoff between the other properties of SH: It can be computed by an algorithm that is more communication-efficient than the natural SH algorithm (exponentially more so in the worst case), but it might fail to achieve exact budget balance or exact minimum efficiency loss (albeit by a bounded amount). Second, we completely characterize the groups that can strategize successfully against the MC mechanism.

In this abstract, we provide only the statement of our results, because of space limitations. Please refer to our journal submission [1] for a more thorough introduction, a discussion of related work, proofs, and open problems.

*The full paper has been submitted for journal publication and is available in preprint form [1].

†Supported in part by ONR grants N00014-01-1-0795 and N00014-01-1-0447 and NSF grant CCR-0105337.

‡Supported in part by NSF Career grant CCR-9985304.

§Supported in part by NSF Career grant CCR-9702980 and ONR grant N00014-01-1-0795.

¶Supported in part by NSF grant ANI-9730162.

EC'01, October 14-17, 2001, Tampa, Florida, USA.

2. TERMINOLOGY AND NOTATION

We use the following multicast-transmission model, following [2]: 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 $R \subseteq P$, the multicast tree $T(R)$ is merely the minimal subtree of $T(P)$ required to reach the elements in R.

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 $(u_1, u_2, \cdots, u_{|P|})$. The mechanism M is then a pair of functions $M(u) = (x(u), \sigma(u))$. The receiver set for a given input vector is $R(u) = \{i \mid \sigma_i = 1\}$. A user's individual *welfare* is therefore 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).

A *strategyproof* cost-sharing mechanism is one that satisfies $w_i(u) \geq w_i(u|^i\mu_i)$, 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, M is GSP if the following property holds for all u, v (where $S = \{i | u_i \neq v_i\}$): Either $w_i(v) = w_i(u)$, $\forall i \in S$, or $\exists i \in S$ such that $w_i(v) < w_i(u)$.

Economic considerations [3] lead naturally to the study of two cost-sharing mechanisms: *marginal cost* (MC) and *Shapley value* (SH). The MC mechanism is *efficient*, *i.e.*, it maximizes the overall welfare NW, and strategyproof [3], and it can be computed by a simple distributed algorithm that sends exactly two messages over each link of $T(P)$, the universal multicast tree [2]. The SH mechanism is *budget-balanced*, *i.e.*, it computes a receiver set R and cost shares x_i such that $\sum_{i \in R} x_i = \sum_{l \in T(R)} c(l)$, and GSP; among all mechanisms with these two properties, it minimizes the worst-case loss in overall welfare [3]. The natural distributed algorithm for SH requires a total of $\Omega(|P| \cdot |N|)$ messages in the worst case, and there is evidence that this unacceptably high number of messages is inherent [2].

3. TOWARDS APPROXIMATING THE SH MECHANISM

The SH mechanism divides the cost of each link in $T(R)$ equally among all users in R who receive the transmission through that link. Consider, as in [2], the following natural algorithm for SH: Set $R = P$, and do a bottom-up traversal of the tree that determines, for each node α, the number p_α of members of R in the subtree rooted at α. Then, do a top-down traversal, which the root initiates by sending the number $md = 0$ to its children. After receiving message md, node α computes $md' \equiv \left(\frac{c(l)}{p_\alpha}\right) + md$, where l is the network link between α and its parent, assigns the cost share md' to each of its resident users, and sends md' to each of its children. We cannot assume that $u_i \geq md'$ for all i, and so some users may prefer not to receive the transmission. After one pass up and down the tree, we update R by dropping all users i such that $u_i < md'$ and repeat. The algorithm terminates when no more users need to be dropped.

Unfortunately, this algorithm could make as many as $|P|$ passes up and down the tree and send a total of $\Omega(|N| \cdot |P|)$ messages in the worst case. Moreover, [2] contains a corresponding lower bound for a broad family of algorithms: There is an infinite class of inputs, with $|P| = \mathcal{O}(|N|)$, for which any "linear distributed algorithm" that computes SH sends $\Omega(|N|^2)$ messages in the worst case.

Our first step towards a more communication-efficient mechanism that has some of the desirable properties of SH is to present an SH algorithm that makes just one pass up and down $T(P)$.

Let v be the (reported) utility profile. Then, for every link l in $T(P)$, we compute:

$n_l(p, v) \stackrel{def}{=}$ *the number of players in the subtree beneath l who are each willing to pay p for the links above l* (*i.e.*, the number of players in this subtree who will not drop out of the receiver set when their cost share for the links from the root down to but excluding l is p).

Note that this definition requires that the cost from the leaves through l has already been adjusted for. A single definition is sufficient for this purpose, because the SH mechanism does not discriminate among receivers downstream of l when sharing the cost of l or its ancestors; all such receivers pay the same amount for these links. For each link, we compute this function at all prices p. The function $n_l(p, v)$ is monotonically decreasing with p, and, for any given utility profile v, can be represented by at most $|P|$ points with coordinates (p_i, n_i). We use this *list-of-points* representation of $n_l(p, v)$ in our algorithm.

It is not obvious that $n_l(\cdot)$ can be computed in one bottom-up pass of $T(P)$; the computation is described in detail in Section 2.2 of our journal submission [1]. After the bottom-up pass is complete, our algorithm does a top-down pass to compute the prices charged to each user. Assuming that the nodes have stored the functions $n_l(\cdot)$ on the way up the tree, we compute the prices on the way down as follows: For each link l, we let x_l be the cost share of any receiver below l for the path down to (but not including) l. If l is the link from node β to β's parent, then we use x_l and x_β interchangeably, and we refer to the links from β to its children as "the child links of l." Then, $x_{root} = 0$ and, if l has child links $l_1, l_2, \cdots l_k$, we put $x_{l_j} = x_l + c(l)/n_l(x_l, v)$.

We descend the tree in this manner until we get a price x_i for every player $i \in P$: If i is at node β, and l is the link from β to its parent, then $x_i = x_l + c(l)/n_l(x_l, v)$. Then, we include i in $R(v)$ iff $x_i \leq v_i$, and if included i pays x_i.

Although this algorithm computes SH in one pass up and down $T(P)$, the function $n_l(\cdot)$ requires, in the worst case, $|P|$ points (p_i, n_i) to represent. To address this problem, we define a new mechanism, called SF (for "step function"), that uses an under-approximation $\tilde{n}_l(p, v)$ instead. For some parameter $\kappa > 1$, we round down all values of $n_l(p, v)$ to the nearest power of κ. The resulting monotonically decreasing function $\tilde{n}_l(p, v)$ has at most $(\log |P| / \log \kappa)$ "downward steps," and so it can be represented by a list of $\mathcal{O}(\log |P|)$ points.

The computation of $\tilde{n}_l(\cdot)$ in one bottom-up pass of $T(P)$ is similar to the computation of $n_l(\cdot)$ and is described in Section 2.3 of our journal submission [1]. The top-down calculation of cost shares x_i is done exactly as before, except that x_{l_j} is set to $x_l + c(l)/\hat{n}_l(x_l, v)$, instead of $x_l + c(l)/n_l(x_l, v)$, where $\hat{n}$ is an over-estimate of $\tilde{n}$ computed and stored during the bottom-up pass.

Mechanism SF does not achieve exact budget balance; there may be a budget surplus. Our final step is a further modification, called mechanism SSF (for "scaled SF"), defined as follows: Let h_l be the height of link l in the tree. (If one of the endpoints of link l is a leaf, then $h_l = 1$.) Then, define the *scaled cost* $c^\kappa(l)$ of the link l to be $c(l)/(\kappa^{h_l})$. Let h be the height of $T(P)$. Run mechanism SF assuming link costs $c^\kappa(l)$, instead of $c(l)$, to compute a receiver set $R^\kappa(u)$ and cost shares $x_i^\kappa(u)$.

THEOREM 1. *Mechanism SSF is group-strategyproof.*

THEOREM 2. *Mechanism SSF may run a budget deficit, but the budget always obeys the following bounds:*

$$\frac{c(T(R^\kappa(u)))}{\kappa^h} \leq \sum_{i \in R^\kappa(u)} x_i^\kappa(u) \leq c(T(R^\kappa(u)))$$

THEOREM 3. *Let $R(u)$ be the SH receiver set. Then mechanism SSF obeys the following worst-case bound on loss in overall welfare, where $U = \sum_{i \in P} u_i$:*

$$NW(R^\kappa(u)) \geq NW(R(u)) - (\kappa^h - 1)U$$

For example, when $|P| = 100,000$ and $h = 5$, the natural algorithm for the SH mechanism given in [2] would require about $100,000$ messages to be sent across a link in the worst case. Our algorithm for SSF requires one bottom-up pass and one top-down pass, *i.e.*, exactly two messages over each link. The maximum size of each point (p_i, n_i) in a message in the bottom-up pass is always bounded by $O(\log |P| + \max_{i \in P} \log u_i)$ bits, and the maximum size of a message sent in the top-down pass is always bounded by $O(\log(\sum_{l \in L} c(l)))$ bits. For $|P| = 100,000$, $h = 5$, and $\kappa = 1.03$, SSF has a budget deficit of at most 14% of the tree cost and a worst-case welfare loss with respect to SH of at most 16% of the total utility, and the largest message sent in the bottom-up pass contains at most 400 points (p_i, n_i). As another example, when $|P| = 10^6$ and $h = 10$, we can use $\kappa = 1.02$ to achieve a worst-case deficit of 18% and worst-case welfare loss of 22% of the total utility, with maximum bottom-up message size of 700 points, or use $\kappa = 1.04$ to achieve corresponding bounds 33%, 48%, and 350 points.

4. GROUP STRATEGIES THAT SUCCEED AGAINST THE MC MECHANISM

The MC algorithm given in [2] highlights interesting features of the mechanism, and we describe them here.

Given an input utility profile u, the receiver set is the unique *maximal efficient set* of players. To compute it, we recursively compute the *welfare* 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 maximal efficient set $R(u)$ is the set of all players i such that every node on the path from i to the root has nonnegative welfare.

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 players at nodes in the subtree $T_1(u)$ containing the root.

Once $F(u)$ has been computed, for each player 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 payment $x_i(u)$ of each player i is defined as

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

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

We say that a strategy v for a group S is *a successful group strategy at the given utility profile u* if, $\forall i \in S$, $w_i(v) \geq w_i(u)$, and $\exists j \in S$ such that $w_j(v) > w_j(u)$. In other words, a successful group strategy at u is one that benefits at least one member of the group and harms none of the members of the group. If the group S has only two members, we call it a *successful pair strategy.* If there is no group that has a successful strategy at u, then we say that the mechanism is *group-strategyproof (GSP) at u.* A GSP mechanism is one that is GSP at all u.

It is well known that the MC mechanism is not GSP. Here we ask at which utility profiles, if any, is MC group-strategyproof, and, for a utility profile u at which MC is not GSP, what are the successful strategies?

THEOREM 4. *Let u be the true utility profile, and let $P'(u) \subseteq P$ be the set of players who do not maximize their welfare at u: $P'(u) = \{i \in P \mid w_i(u) < u_i\}$. Then, the MC mechanism is GSP at utility profile u if and only if, for every player $i \in P'(u)$, the following condition is satisfied:*

> **There is no player j in the same component of $F(u)$ as i such that j is in the subtree rooted at $X(i,u)$.**

THEOREM 5. *Consider a true utility profile u. Suppose coalition S has a successful strategy v. Then there exists a pair of players $i, j \in S$ such that i and j have a successful pair strategy v'. Furthermore, i and j are in the same set $T_l(u)$ in $F(u)$, and j is in the subtree rooted at $X(i,u)$.*

THEOREM 6. *If S has a successful group strategy at a profile u, then any superset of S has a successful group strategy at u.*

5. REFERENCES

[1] J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker, "Approximation and Collusion in Multicast Cost Sharing," submitted. Available at `http://www.cs.yale.edu/homes/jf/FKSS.pdf`

[2] J. Feigenbaum, C. Papadimitriou, and S. Shenker, "Sharing the Cost of Multicast Transmissions," to appear in *Journal of Computer and System Sciences.* Available at `http://www.cs.yale.edu/homes/jf/FPS.pdf`

[3] H. Moulin and S. Shenker, "Strategyproof Sharing of Submodular Costs: Budget Balance Versus Efficiency," to appear in *Economic Theory.* Available at `http://www.aciri.org/shenker/cost.pdf`

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