Approximate Budget Balanced Mechanisms with Low Communication Costs for the Multicast Cost-Sharing Problem

Markus Bläser*

1 Introduction

We investigate the relation between budget balance and communication for the multicast cost-sharing problem in the context of distributed algorithmic mechanism design. We use the formal model introduced by Feigenbaum, Papadimitriou, and Shenker [3]: Our network is a rooted undirected tree $T = (V, E)$ with n nodes. The root r of T models the service provider. The set P of leaves of T represents the users, who wish to receive the transmission of the provider. Let $p = |P|$. Each $e \in E$ has a weight c_e. This weight represents the costs of using e for the transmission. If a transmission is sent to a subset $R \subseteq P$ of the users, then it is sent along the edges of the smallest subtree of T containing r and all nodes of R. We call this subtree $T(R)$. The costs $c(T(R))$ of this subtree is the sum of the weight of its edges. Each $i \in P$ has a utility u_i which he derives from getting the transmission. The u_i are private information.

A *cost-sharing mechanism* determines which of the users receive the multicast transmission and which price they have to pay. The set of each user's strategies is to report any value $b_i \geq 0$ as their utility. Based on the input vector $b = (b_1, \dots, b_p)$, the mechanism decides which users receive the transmission and assigns prices to the users. The value $x_i(b)$ denotes the price user i has to pay. $\sigma_i(b)$ equals one if i gets the transmission and is zero otherwise. The *receiver set* $R(b)$ is the set of all users receiving the transmission. The *individual welfare* $w_i(b)$ of user i is defined by $w_i(b) = \sigma_i(b)u_i - x_i(b)$. The aim of each user is to maximize his individual welfare.

Like Feigenbaum, Papadimitriou, and Shenker [3], we assume that messages arrive reliably, in order, and without significant delay. Each message here consists of a number that is algebraic[1] in the b_i and the c_e. We mainly care about "hotspot communication costs", that is, the maximum number of messages per edge should be small, say $O(1)$ or $O(\text{polylog}(n))$.

*Institut für Theoretische Informatik, ETH Zürich, 8092 Zürich, Switzerland, mblaeser@inf.ethz.ch . Work performed at the Institut für Theoretische Informatik, Universität zu Lübeck, Germany. Supported by the Deutsche Forschungsgemeinschaft under grant BL 511/5-1.

[1]This prohibits encoding tricks.

1.1 Cost-Sharing Mechanisms.

As a countermeasure against users misreporting their utilities, mechanisms should be (group) strategyproof.

Group Strategyproof (GSP): For every coalition $C \subseteq P$ and every vector $b = (b_1, \dots, b_p)$ with $b_i = u_i$ for all $i \notin C$ the following holds: If $\sigma_i(b)u_i - x_i(b) \geq \sigma_i(u)u_i - x_i(u)$ for all $i \in C$, then this holds with equality for all $i \in C$.

We only treat mechanisms that also satisfy the following three technical properties, which are natural in the context of multicast cost-sharing.

No Positive Transfer (NPT): For all i, $x_i(b) \geq 0$.

Voluntary Participation (VP): For all i, $w_i(b) \geq 0$ provided that i bids truthfully, i.e., $b_i = u_i$.

Consumer Sovereignty (CS): Every user i gets the transmission as long as his bid b_i is high enough.

Two further requirements are usually considered. One is *efficiency* (in a socio-economic sense) which we will not deal with here, because this case is rather well understood [3]. We are concerned with mechanisms that meet GSP, NPT, VP, and CS and are (approximately) budget-balanced as defined below.

Budget Balance (BB): $\sum_{i \in R(b)} x_i(b) = c(T(R(b)))$.

Algorithms for budget balanced mechanisms necessarily have high communication costs (see Section 2.1). Therefore, we investigate mechanisms that are only approximate budget balanced.

α-Approximate Budget Balance (α-BB): $(1/\alpha) \cdot c(T(R(b))) \leq \sum_{i \in R(b)} x_i(b) \leq \alpha \cdot c(T(R(b)))$. ($\alpha$ may be a function depending on the network.)

2 Results

2.1 Previous Results.

The Shapley Value SH and any other mechanism that meets GSP, NPT, VP, CS, and BB and that is also symmetric[2] has bad network complexity: Feigenbaum et al. [2] show that such a mechanism has to sent $\Omega(p)$ bits over $\Omega(n)$ edges.

[2]A mechanism is symmetric if users connected by a path with costs zero are treated equally.

Archer et al. [1] propose an approximation SSF of SH with low communication costs. Their algorithm for SSF sends $\log p/\log\kappa$ numbers over each edge for any fixed parameter $\kappa > 1$. SSF is only κ^h-BB, where h is the height of T. [3]

2.2 New Results. We first observe that SSF is asymptotically budget balanced (i.e., the function α in the definition of α-BB fulfills $\alpha(n) \to 1$ for $n \to \infty$) on trees of polylogarithmic height (which seems to be perfectly reasonable for multicast trees) when choosing κ properly. This conclusion is new and, in particular, is not drawn by Archer et al. Second, we present a mechanism N that meets GSP, NPT, VP, and CS and can be computed with only a polylogarithmic number of messages per edge. N is also reasonably budget balanced, more precisely, it is $O(\log n)$-BB. Compared with the mechanism of [1], which is only κ^n-approximate budget balanced on trees of height $\Omega(n)$ for some $\kappa > 1$, this is almost a doubly exponential improvement.

3 Proofs

3.1 Trees with polylogarithmic height. SSF performs particularly well on trees of polylogarithmic height when setting $\kappa = 1 + 1/\bar{h}$, where $\bar{h} = (\max\{\log n, h\})^{1+\epsilon}$, h is the height of the multicast tree, and $\epsilon > 0$ is a fixed constant. Then SSF sends at most $(1 + o(1)) \cdot \bar{h} \cdot \log p$ numbers over each edge, as $\log(1+x) \geq \frac{x}{x+1}$ for $x \geq 0$. Moreover, SSF is $(1+1/\bar{h})^h$-BB. Since $(1+x)^m \leq 1 + \frac{mx}{1-mx}$ for all $m \in \mathbb{N}$ and $x \geq 0$ with $mx < 1$, $(1 + 1/\bar{h})^h = 1 + o(1)$ as a function in n.

Theorem 3.1. *On trees of polylogarithmic height, SSF is asymptotically budget balanced while sending only a polylogarithmic number of messages over each edge.*

3.2 Trees with arbitrary height. To construct N, we basically use the mechanism SSF but on a modified tree T' with height $O(\log n)$. This modification is only "virtual" in the sense that T' can be embedded into T in an appropriate way.

T' will be a topology tree for T as defined by Frederickson.[4] Throughout this section, we refer to the definitions in [4, Section 2]. Topology trees are only defined for binary trees. Therefore, we first have to make T binary. If v is a node of T with children $v_1, \ldots, v_\ell$, then we insert a binary tree of height $\log \ell$ and give all the new edges weight zero. This modification is only "virtual" in the sense that we do not have to change T. All the changes are simulated by v when running the mechanism.

Next, we compute a topology tree T' for T and embed it into T. T' can be computed by calling the procedure *cluster* in [4, Section 2] $O(\log n)$ times. Procedure *cluster* consists of one top-down and one bottom-up pass and sends a constant number of messages over each edge.

We call the node v of a cluster C that is closest to the root r of T the root of C. In T', an edge connecting a cluster C at level ℓ with a cluster C' at level $\ell + 1$ gets the weight of the path from v to v', where v and v' are the roots of C and C'. By induction, it follows that the weight of a path in T from any leaf u to the root r of T equals the weight of the path in T' from u to the root of T'. The following lemma summarizes these ideas.

Lemma 3.1. *The tree T' has height $O(\log n)$. It can be computed on T with $O(\log n)$ messages per edges. The weight of the path in T from a leaf u to the root equals the weight of the path in T' from u to the root. Each edge of T contributes to $O(\log n)$ edge weights of T'.*

Our mechanism N now works as follows: It simply runs SSF on T'. N inherits all game-theoretic properties of SSF. Furthermore, it is $O(\log n)$-BB.[5] To execute N on T, any computation of SSF at a cluster C is carried out in T at its root v. We send messages that are sent from cluster C to C' in T' from v to v' in T. Since each edge of T contributes weight to $O(\log n)$ edges of T', this increases the number of messages send over each edge by a factor of $O(\log n)$.

Theorem 3.2. *Mechanism N meets GSP, NPT, VP, and CS. It is $O(\log n)$-approximate budget-balanced. Mechanism N can be executed by sending $O((\log n)^{3+\epsilon})$ messages over each edge.*

References

[1] A. Archer, J. Feigenbaum, A. Krishnamurthy, R. Sami, S. Shenker. Approximation and collusion in multicast cost sharing. *Games and Economical Behaviour*, to appear. `www.cs.yale.edu/homes/jf/AFKSS.ps`.

[2] J. Feigenbaum, A. Krishnamurthy, R. Sami, S. Shenker. Hardness results for multicast cost sharing. In *Proc. 22nd Conf. on Found. of Software Technology and Theoret. Comput. Sci. (FSTTCS)*, LNCS 2556, pp. 133–144, 2002.

[3] J. Feigenbaum, C. Papadimitriou, S. Shenker. Sharing the cost of multicast transmissions. *J. Comput. Sys. Sci.* 63:21–41, 2001.

[4] G. N. Frederickson. Ambivalent data structures for dynamic 2-edge connectivity and k smallest spanning trees. *SIAM J. Comput.* 26(2):484–538, 1997.

[3] Archer et al. also bound the efficiency loss. Due to space limitations we do not deal with this issue here.

[4] We thank an anonymous referee for pointing out the usefulness of topology trees for our purposes.

[5] Note that the users always overpay under N. By scaling with $(\log n)^{-1/2}$, we could N even make $O((\log n)^{1/2})$-BB.