Pricing Multicasting in More Flexible Network Models

Micah Adler* Dan Rubenstein†

Abstract

The problem of designing efficient algorithms for sharing the cost of multicasting has recently seen considerable attention. In this paper, we examine the effect on the complexity of pricing when two flexibility-enhancing mechanisms are incorporated into the network model. In particular, we study a model where the session is offered at a number of different rates of transmission, and where there is a cost for enabling multicasting at each node of the network. We consider two techniques that have been used in practice to provide multiple rates: using a layered transmission scheme (called the *layered paradigm*) and using different multicast groups for each possible rate (called the *split session paradigm*). We demonstrate that the difference between these two paradigms has a significant impact on the complexity of pricing multicasting.

For the layered paradigm, we provide a distributed algorithm for computing pricing efficiently in terms of local computation and message complexity. For the split session paradigm, on the other hand, we demonstrate that this problem can be solved in polynomial time if the number of possible rates is fixed, but if the number of rates is part of the input, then the problem becomes NP-Hard even to approximate. We also examine the effect of delivering the transmissions for the various rates from different locations within the network. We show that in this case, the pricing problem becomes NP-Hard for the split session paradigm even for a fixed constant number of possible rates, but if layering is used, then it can be solved in polynomial time by formulating the problem as a totally unimodular integer program.

1 Introduction

Multicast transmission offers tremendous savings in network bandwidth over unicast transmission for applications that deliver the same content to multiple customers by allowing these customers to "share" the transmission on common access links [8]. However, this sharing of link bandwidth signifi cantly complicates the issue of pricing [9]. Charging all receivers equally is not an adequate solution, since some receivers might be charged more than they would be willing to pay. If such a receiver drops out of the multicast session, the remaining receivers would be charged more than if a lower, acceptable price had been offered to the exiting receiver. An alternative approach to pricing that has received considerable attention recently [11, 17, 19, 24, 25] is to have each receiver place a bid for the content. The network uses these bids to determine the set of receivers that obtain the content, as well as the price these accepted receivers pay. The price charged to an accepted receiver can be no more than its bid, but for reasons discussed below, it is often advantageous to charge receivers a smaller price. The policy that the network uses to make these decisions is referred to as a *pricing mechanism*. The task of making these decisions using a specifi c pricing mechanism is referred to as *realizing* that pricing mechanism.

An algorithm for realizing a pricing mechanism for multicast in a distributed environment such as the Internet should be effi cient in terms of both the computation performed at the distributed nodes of the network, as well as the communication between these nodes. Identifying these types of algorithms was fi rst addressed by Feigenbaum, Papadimitriou and Shenker [11]. They consider two pricing mechanisms: *Marginal Cost* and *Shapley Value*, and provide effi cient algorithms for the Marginal Cost mechanism, as well as algorithms and lower bounds on the effi ciency of algorithms for the Shapley Value mechanism.

In this paper, we address the multicast model used for the work in this area. In particular, the previous work on pricing mechanisms for multicasting [11, 17, 19, 24] uses a transmission model with a number of simplifying assumptions. Here, we address two of these assumptions: (1) that there is only one possible rate of transmission of the multicast session, and (2) that every network router can forward an incoming packet on multiple outgoing interfaces at no additional cost. Both multiple rate sessions and the fact that not every node of the network is capable of multicasting are very real concerns in the current Internet (e.g., see [9, 7]), and thus the ability to realize pricing mechanisms in models that account for these concerns is central to the task of designing algorithms for pricing mechanisms. Incorporating these issues into the transmission model makes the task of realizing pricing mechanisms for multicasting considerably more challenging.

*University of Massachusetts, Department of Computer Science, University of Massachusetts, Amherst, MA 01003-4610. E-mail: micah@cs.umass.edu.

†Columbia University, Department of Electrical Engineering, 500 W. 120th Street, New York, NY 10027. E-mail: danr@ee.columbia.edu.

In the transmission model of this paper, we assume that a multicast session can be sent at any of a set of ℓ pre-determined rates $\rho_1 \leq \rho_2 \leq \cdots \leq \rho_\ell$. The availability of multiple rates increases flexibility of the multicast paradigm in that each receiver can be allocated a rate based on the available capacities along the path to it from the source, as well as on the price charged for that rate and the utility gained by the receiver and network when the information is transmitted at that rate for the specified price. We consider the two most common techniques for providing multiple rates that are used in practice. The first uses a separate multicast *group* for each possible rate [6]. The second uses a separate *layer* for each rate, where layer 1 has rate ρ_1, layer i, $1 < i \leq \ell$, has rate $\rho_i - \rho_{i-1}$, and to obtain rate ρ_j, a receiver is sent layers $1 \ldots j$ [3, 4, 23, 32]. We refer to the first technique as the *split session* paradigm, and the second as the *layered* paradigm. For both paradigms, each receiver places a bid per rate indicating its willingness to pay for delivery at that rate. In addition to determining the set of accepted receivers for the multicast session, the network must now also determine, for each user, what rate is obtained. We here demonstrate that the seemingly small difference between these paradigms has a significant effect on the complexity of realizing pricing mechanisms.

Our model also assumes that for each node of the network, there is a cost for *enabling* that node, i.e., for making it capable of multicasting. If a node is enabled, it implements multicast and can forward a received packet down multiple links; otherwise it unicasts, and forwards a packet down a single link. This cost model can be justified by the observation that most routers are designed to perform unicast forwarding and do so with great efficiency. In contrast, the overhead incurred as a function of processing and memory by a router to execute multicast can be significant [9], and would limit the aggregate levels of traffic the router could support.

To realize a pricing mechanism in this model, the network must still determine the set of accepted receivers, but which set is chosen is now influenced by the cost of enabling multicasting at each node. Furthermore, the network must choose the set of nodes of the network to enable.

As in [11], many of our results assume that there is a single directed multicast tree that defines the routes used by all transmissions. [11] demonstrates that without this assumption, even when there is no cost for enabling a node, and when there is only a single possible rate, it becomes NP-Hard to find constant factor approximations to the pricing mechanism we consider here (the problem becomes a version of the Prize Collecting Steiner Tree problem [21, 13]). Sophisticated techniques for approximating other pricing mechanisms when the routes can vary depending on what receivers are accepted are provided in [19]. Incorporating such techniques into the more involved network model we consider here is an interesting open problem. We also point out that while the hardness result of [11] indicates that removing the single tree assumption entirely leads to intractable problems, it does not rule out the possibility of efficient algorithms for a more modest generalization (described below).

We also mention that there has been investigation into distributed algorithms of a flavor similar to [11] that maximize profit in networks that consist of greedy agents for problems that do not involve multicast. Hershberger and Suri [16] explore distributed mechanisms to find the lowest cost shortest path between two nodes in a graph where each edge is an agent who seeks to maximize its own profit by charging for utilization of its edge.

1.1 Summary of results

In this paper, we focus on the Marginal Cost pricing mechanism [25]. While this is only one of several important mechanisms, it serves as a good test case: we believe that our results for the Marginal Cost mechanism provide important insights on the effect of the practical network considerations we study for pricing mechanisms in general. We describe the Marginal Cost mechanism, a special case of the Vickrey-Clark-Groves mechanism (see [26]), in Section 2. To the best of our knowledge, the version of Marginal Cost we use is the first pricing mechanism that has been used for multicast sessions with multiple rates, i.e., where agents bid on a set of items greater than size one and where each agent is assigned (possibly overlapping) subsets of these items. Previous multicast work assumes that these sets are of size one. Auction mechanisms for multiple goods where there is *no* cost for each additional copy of a good delivered have been studied in [14]. Auctions that partition goods among multiple agents into non-overlapping subsets have also been considered [26]. We note here that this mechanism can also be used to select the network configuration that maximizes the network profit when every accepted receiver pays exactly what it bid for the service it receives. The profit obtained by such a mechanism is referred to as the *network welfare*.

In Section 3, we consider algorithms for realizing Marginal Cost in the layered paradigm with costs for enabling nodes for multicast. We provide a distributed algorithm for this problem that is efficient in terms of the amount of local computation performed at each node, and only requires three messages per edge of the multicast tree. For every edge, the total number of bits in these 3 messages is $O(h\ell K)$, where h is the height of the multicast tree, ℓ is the number of possible layers, and K is the number of bits used to represent the bids and costs of the network.

The algorithm in [11] for the same problem (in the simpler model) achieves the same result using only $O(K)$ bits of communication per edge. However, in Section 4, we provide two lower bounds on the communication required to maximize network welfare: We demonstrate that $\Omega(hK)$ bits are necessary when considering networks with a cost for enabling multicast, even when there is only a single possible rate of transmission, and we demonstrate that $\Omega(\ell K)$ bits are necessary when there are ℓ layers, even when there is no cost for enabling multicasting at any node. While these two bounds together only imply a lower bound of $\Omega((h+\ell)K)$, they provide evidence that the $O(h\ell K)$ achieved by our algorithm is reasonable.

We also study realizing the Marginal Cost mechanism for the split session paradigm with costs for enabling nodes. We demonstrate that our algorithm for the layered case can be adapted to provide a solution for this case. However, the computation required of the adapted algorithm becomes proportional to 2^ℓ, and thus, this algorithm is only applicable for the case where ℓ is small. We also demonstrate that we should not expect to find an efficient algorithm for large ℓ. In particular, we demonstrate that it is NP-Hard to determine even any reasonable approximation to the network welfare when the number of possible rates is part of the input. This result holds even in networks without a cost for enabling multicasting functionality at nodes. This hardness result is significant in that it demonstrates that in terms of realizing pricing mechanisms, the layered paradigm enjoys a considerable advantage over the split session paradigm. One intuition for this difference is that for the layered paradigm, there is a linear number of combinations of layers that a link is allowed to carry, while in the split session paradigm, there is an exponential number of combinations of groups that a link is allowed to carry.

Finally, in Section 5, we turn to the question of removing the assumption that there is a single fixed multicast tree. We consider the effect of having a single fixed multicast tree for every layer or group comprising the session, but the trees for the different transmissions need not be the same. We demonstrate that in this case, maximizing network welfare for the split session paradigm becomes NP-Hard even for the case of a constant number of possible rates, and no cost for enabling multicasting. Somewhat surprisingly, we find that in the multiple tree case, the Marginal Cost mechanism for the layered paradigm can be realized in polynomial time if there is no cost for enabling multicasting. We demonstrate this by showing that this problem can be expressed as a totally unimodular integer program.

2 Network Model and Optimization Problems

We consider the problem of offering delivery of a single multicast session, in isolation, to a set $R \subset N$ of receivers, over a network modeled as a directed graph $G = (N, E)$. The session emanates from a source $s \in N$, and is delivered to the receivers via the edges and vertices of G. When a node $n \in N$ is enabled, any flow of information entering that node can be forwarded on multiple outgoing edges. If a node is not enabled, then each copy of an incoming flow can only be forwarded on a single outgoing edge. There is a cost c_n for enabling node n. We can also model nodes that cannot be enabled by setting their enabling cost to ∞. We assume here that the source s can be enabled for free (and thus is always enabled), although all of our results can be modified to apply when this is not the case.[1]

There is also a set of costs for using a directed edge $e \in E$, denoted by $\mathbf{c}_e$, an ℓ-dimensional vector. In the split session paradigm, $[\mathbf{c}_e]_j$ (the jth entry of vector $\mathbf{c}_e$) is the cost at e of transmitting the jth group across this edge. In the layered paradigm, $[\mathbf{c}_e]_j$ is the cost of transmitting the jth layer. We assume that receiver r expresses its willingness to pay via a bid, denoted by $\mathbf{b}_r$, an ℓ-component vector such that $[\mathbf{b}_r]_j$ indicates the price that r is willing to pay to receive the jth group or layer. In the split session paradigm, a receiver is sent at most one group. In the layered paradigm, if a receiver is sent layer j, it must also be sent layers 1 through $j-1$.

For most of this paper, we make an assumption analogous to that made in [11]: the multicast session uses a single source, and there is a unique path from that source to each receiver, regardless of the set of receivers or enabled nodes. With this assumption, we can restrict our attention to the nodes and edges of the tree formed by the union of paths from the source to each of the receivers. We refer to this tree as the *multicast tree*. When considering networks where every node is enabled for multicasting, this assumption is justified by the multicast routing strategy, commonly used in practice [8], consisting of a tree of shortest paths from the receivers to the source. In the case of either the layered or the split session paradigm, this kind of routing may be used with different source nodes for the different groups or layers, since this is an effective way to balance the session load throughout the network. Thus, in Section 5, we consider a model where every layer or group uses a fixed multicast tree, but the trees do not have to be the same.

The fixed multicast tree assumption does limit the practical applicability of our model to some scenarios. However, we consider the model of this paper an important step towards understanding algorithms for pricing in such networks.

[1] For instance, a simple modification that would permit enabling of the source is to attach a dummy node as a parent of the original source with an edge whose along which transmissions are free. This makes the original source an intermediate node in a tree whose cost structure is identical to that of the original formulation, and under out model, it is permissible to assign an enabling cost to the original source node.

Furthermore, the hardness result in [11] that considers the selection from among multiple routes applies to a network where the routing depends on which nodes are enabled, even if the cost of enabling those nodes is zero. Thus, unless $P = NP$, we must either make some restriction on the choice of routes, or take an approach similar to that of [19], which considers approximation algorithms for other pricing mechanisms in the simpler network model.

For the optimization problems we consider, the input is distributed as follows: each node n is informed of c_n, $\mathbf{c}_e$ for each e incident to n, and $\mathbf{b}_r$ for any r located at n. The simplest problem we consider is maximizing network welfare. In that problem, the network must determine a set $R' \subset R$ of receivers that are sent the multicast session, for each $r \in R'$ a rate of transmission, and the set of multicast enabled nodes. Each receiver in R' pays exactly what it bid for the group or subset of layers that it receives, and the other receivers pay nothing. The network welfare is the total payments minus the total costs.

2.1 The Marginal Cost Mechanism

A natural definition of a receiver's *satisfaction* is the utility obtained from the service provided by the network minus (the utility of) the amount it must pay to receive this service. The simple network welfare pricing mechanism described above produces a profit for the network equal to the network welfare, but it also gives a receiver an incentive to bid less than its true utility. By bidding a smaller value, a receiver reduces the cost of receiving the session, thereby increasing its own satisfaction but reducing overall network profits. What is needed is a *strategy-proof* mechanism: a mechanism where a receiver maximizes its satisfaction by bidding its true utility. There are a number of other properties that are desirable in a pricing mechanism, including:

- Efficiency: a configuration that maximizes (total utility minus total cost) is chosen.
- No Positive Transfers (NPT): the price that the receiver pays is not negative.
- Voluntary Participation (VP): receivers that are not admitted are not charged anything.
- Consumer Sovereignty (CS): A receiver is always able to guarantee acceptance of a bid if the bid is increased to a sufficiently large value.

It is shown in [25] that the Marginal Cost pricing mechanism is strategy-proof, efficient, NPT, VP, and CS. A drawback to Marginal Cost is that it does not provide another desirable property, called Budget-balance. In a budget-balanced pricing strategy, the amount paid by receivers exactly equals the cost of transmission. Marginal Cost never runs a budget surplus, but may run a deficit. This is one reason to also consider other mechanisms, such as Shapley Value [28], although that mechanism does not satisfy Efficiency. We refer the reader to [11, 25] for motivation and further explanation of these properties.

We note that in the Marginal Cost pricing mechanism used in this paper, each user is not restricted to an "all or nothing" allocation. In all previous work on pricing multicasting that we are aware of, each receiver either obtains the entire multicast session or it does not receive anything. Under our definition of Marginal Cost, by giving each receiver access to a variety of rates, receivers can receive one of several "levels" of the service.

Consider any network G and set of receivers R that wish to join a session under a network model where each receiver i submits a set of bids, $\mathbf{b}_i = \langle b_i^1, b_i^2, \cdots, b_i^n \rangle$, of which at most one is accepted. In the multiple good Marginal Cost mechanism, the network chooses the configuration of the network that maximizes network welfare. The price charged to a receiver i that has an accepted bid of b_i^k is defined to be $\mathcal{M}_i = b_i^k - (P^*(R) - P^*(R \backslash \{i\}))$, where $P^*(X)$ is the maximum network welfare when restricting admission to receivers within the set X. If no bid from i is accepted, then i is charged 0. In other words, the price that receiver i must pay is the amount of the bid that was accepted, minus the amount that the network welfare is increased by receiver i participating in the multicast session. The fact that this definition of Marginal Cost is strategy-proof follows from the well known Vickrey-Clarke-Groves Theorem in auction theory [30, 25].

3 Efficient Distributed Algorithms for Marginal Cost

In this section, we present an efficient distributed algorithm for realizing the Marginal Cost mechanism. We first provide an algorithm that maximizes network welfare in the layered paradigm. We then modify the algorithm to maximize network welfare in the split session paradigm, albeit at the cost of an exponential dependence on the number of groups. We then show that our algorithms for both the layered paradigm and the split session paradigm can be converted into algorithms that compute Marginal Cost. The algorithm uses the observation that *the profit contributed of a particular configuration of a subtree rooted at any node n in the multicast tree depends only on the configuration of nodes within that subtree and on the*

configuration of n's ancestor nodes in the tree. In particular, the profit is independent of the configuration of the subtrees rooted at its sibling nodes. A node can therefore determine the maximum profit that the subtree for which it is the root can provide for a given configuration of its ancestor nodes (that determines the set of layers or groups that are available to it) by choosing its own configuration that maximizes the sum of the profits of the subtrees rooted at its child nodes.

For any node n of the multicast tree, let $\pi_{1,n}$ be the parent node of n in the tree. Let $\pi_{k+1,n}$ be the parent of node $\pi_{k,n}$. We define h_n to be the value of k such that $\pi_{k,n}$ is the source node of the tree. For any node n, let $D(n)$ be the set of children of n in the tree. An important value computed during the course of our algorithm is $S_{j,k}(n)$, which is computed for $0 \leq j \leq \ell$ (recall ℓ is the number of layers or groups), $1 \leq k \leq h_n$. $S_{j,k}(n)$ is the maximum network welfare of the subtree rooted at node n, minus the cost of transmitting all necessary layers from node $\pi_{k,n}$ to node n, subject to the following two conditions:

- $\pi_{r,n}$ is not enabled for all $1 \leq r < k$, i.e., packets are unicast through nodes $\pi_{k-1,n}, \pi_{k-2,n}, \cdots, \pi_{1,n}$. Nodes n and $\pi_{n,k}$ may or may not be enabled, but only the cost of enabling n counts against $S_{j,k}(n)$ (since $\pi_{n,k}$ and other potentially enabled nodes nearer to the source are not part of the subtree being considered).
- At most j layers are transmitted from $\pi_{k,n}$ to n (and thus to any node that is a descendent of n.)

In other words, $S_{j,k}(n)$ is the maximum value of the sum of a set of accepted bids that are in the subtree rooted at n, minus the sum of the costs in the subtree rooted at n to deliver those bids, minus the cost of transmitting the necessary layers from node $\pi_{r,n}$ to node n, subject to the two conditions above. Another set of intermediate values used by our algorithm are represented by $\mathbf{c}_{(\pi_{n,k},n)}$, a vector such that $[\mathbf{c}_{(\pi_{n,k},n)}]_j$ is the cost of transmitting one copy of layer j from $\pi_{n,k}$ directly to n (i.e., no intermediate nodes are enabled).

We now describe our algorithm, which we call **Max-Layered-Welfare**. For ease of exposition, we here describe the slightly simpler case where the set of possible receivers is exactly the same as the set of leaves of the multicast tree, but it is not hard to modify this to account for the general case.

Algorithm **Max-Layered-Welfare:**

- The source initiates a phase of the algorithm where every node n of the multicast tree sends each of its children $n_i \in D(n)$ the vector $\mathbf{c}_{(\pi_{n,k},n)}$, for each k, $1 \leq k \leq h_n$. Each child n_i uses these values to compute each $\mathbf{c}_{(\pi_{n_i,k+1},n_i)} = \mathbf{c}_{(\pi_{n,k},n_i)} + \mathbf{c}_{(n,n_i)}$ (i.e., a vector addition).
- Each leaf node n of the multicast tree computes, for each k and j, $1 \leq k \leq h_n$, $0 \leq j \leq \ell$, $S_{j,k}(n)$. Each $S_{0,k}(n) = 0$, and then the remainder are computed in order of increasing j, using the formula $S_{j,k}(n) = \max(\sum_{r=1}^{j}[\mathbf{b}_n]_r - \sum_{r=1}^{j}[\mathbf{c}_{(\pi_{n,k},n)}]_r, S_{j-1,k})$.
- The next phase of the algorithm proceeds from the leaves to the source, and each node n_i sends to its parent n, the value $S_{j,k}(n_i)$, for each k and j, $1 \leq k \leq h_{n_i}$, $1 \leq j \leq \ell$. The node n sets each $S_{0,k}(n) = 0$, and then computes all other values of $S_{j,k}(n)$, proceeding from $j = 1$ to $j = \ell$, using the formula

$$S_{j,k}(n) = \max\left\{ \sum_{n_i \in D(n)} S_{j,k+1}(n_i), \right. \tag{3.1}$$
$$\left. \sum_{n_i \in D(n)} S_{j,1}(n_i) - c_n - \sum_{r=1}^{j}[\mathbf{c}_{(\pi_{n,k},n)}]_r,\ S_{j-1,k}(n) \right\}.$$

- The source node s returns the value $\sum_{n_i \in D(s)} S_{j,1}(n_i)$.

THEOREM 3.1. *The value returned by Algorithm* **Max-Layered-Welfare** *is the maximum possible network welfare under the layered paradigm. Furthermore, the computation performed by any node n requires time $O(\ell h_n |D(n)|)$, and exactly two messages are communicated between node n and each child $n_i \in D(n)$.*

Proof. It is not difficult to implement this algorithm with the stated computation and communication complexity, and thus we here only describe the proof that the value returned by the algorithm is correct. To do so, we prove that for every n, j, and k, the value of $S_{j,k}(n)$ computed by the algorithm is correct. Assuming this, the correctness of the algorithm follows from the fact that the source determines the welfare obtained by sending the optimal number of layers to each of its children. Also

note that it is easy to show that the values of $\mathbf{c}_{(\pi_{n,k},n)}$ computed are correct. Thus, we only need to show the correctness of the $S_{j,k}(n)$. The proof is by a double induction on j and the maximum number of hops from n to a leaf (counting each node regardless of whether or not it is enabled). For the base of the induction, we show that $S_{j,k}(n)$ is correct if either $j = 0$, or n is a leaf. When $j = 0$, $S_{j,k}(n) = 0$, since no receiver in the subtree rooted at n can receive any layers. When n is a leaf, but $j > 0$, we see that $S_{j,k}(n)$ is correct by induction on j, since when up to j layers can be sent to node n, either we send all j layers to n, or we use the best solution using less than j layers (which, by the inductive assumption on j, has been computed correctly and is held in $S_{j-1,k}(n)$).

For the inductive step of the double induction, consider $S_{j,k}(n)$ for any k, any layer $j > 0$, and any non-leaf node n. By the inductive hypothesis on n, we can assume that for every child $n_i \in D(n)$, both $S_{j,k+1}(n_i)$ and $S_{j,1}(n_i)$ are computed correctly, and by the inductive assumption on j that $S_{j-1,k}(n)$ is computed correctly. We consider four possible configurations of the optimal solution: : either node $\pi_{n,k}$ transmits layer j to node n or it does not, and either node n is enabled, or it is not. If, in the optimal solution, node $\pi_{n,k}$ does not transmit layer j to node n, then regardless of whether n is enabled or not, $S_{j,k}(n) = S_{j-1,k}(n)$. Since, by the inductive assumption, $S_{j-1,k}(n)$ is computed correctly (maximizing the welfare of the subtree rooted at n when $j-1$ layers are transmitted from $\pi_{n,k}$ to n), both cases when j is not transmitted are considered. If the optimal solution transmits all j layers to node n, and n is enabled, then $S_{j,k}(n) = \sum_{n_i \in D(n)} S_{j,1}(n_i) - c_n - \sum_{j=1}^{r} [\mathbf{c}_{(\pi_{n,k},n)}]_r$. This holds because we maximize the welfare at the subtree rooted at n when n is enabled by maximizing the welfare of the subtrees rooted at each of its children subject to the condition that each of them receives the multicast session directly from n. From this maximization, we subtract the cost of enabling n and of transmitting one copy of layers 1 through j from $\pi_{n,k}$ to n. Finally, if the optimal solution transmits all j layers to node n, but n is not enabled, then all layers are unicast through n, and the maximum welfare gained through the subtree rooted at n is $S_{j,k}(n) = \sum_{n_i \in D(n)} S_{j,k+1}(n_i)$. Since the algorithm takes the maximum of these possibilities, it does in fact compute the correct value of $S_{j,k}(n)$.

The total number of bits communicated in this algorithm is $O(\ell h K)$, where K is the maximum number of bits required to represent any value $S_{j,k}(n_i)$ or $\mathbf{c}_{(\pi_{n,k},n)}$, and h is the height of the tree. While this is not as small as might be hoped for, we prove in Section 4 that $\Omega(hK)$ bits are necessary when considering networks with a cost for enabling multicast, even when there is only a single possible rate of transmission, and that $\Omega(\ell K)$ bits are necessary when there are ℓ layers, even when there is no cost for enabling multicasting at any node. These two bounds provide evidence that the linear dependence of the number of bits communicated on both h and ℓ is reasonable.

We next show that algorithm **Max-Layered-Welfare** can be modified to compute the maximum welfare for the split session paradigm. We provide a brief sketch of how to do so. We define $S_{\sigma,k}(n)$, where σ is any subset of the ℓ groups, analogously to $S_{j,k}(n)$, except that the second condition becomes

- A group s is transmitted from $\pi_{k,n}$ to n only if $s \in \sigma$.

The algorithm follows along the same lines as **Max-Layered-Welfare**, with small modifications. To compute $S_{\sigma,k}(n)$ for all σ and k at a node n, the algorithm starts with σ being the empty set, then considers all subsets of size 1, followed by all subsets of size two, until σ contains all groups. The main formula of the algorithm (analogous to (3)) is as follows:

$$S_{\sigma,k}(n) = \max \left\{ \sum_{n_i \in D(n)} S_{\sigma,k+1}(n_i),\ \max_{\sigma' \in m(\sigma)} S_{\sigma',k}(n), \right.$$
$$\left. \sum_{n_i \in D(n)} S_{\sigma,1}(n_i) - c_n - \sum_{s \in \sigma} [\mathbf{c}_{(\pi_{n,k},n)}]_s \right\},$$

where $m(\sigma)$ is the set of all subsets of σ containing exactly one less element. We call the resulting algorithm **Max-Split-Welfare**. The proof of the following follows along the lines of the proof of Theorem 3.1.

THEOREM 3.2. *The value returned by Algorithm* **Max-Split-Welfare** *is the maximum possible network welfare under the split session paradigm. Furthermore, the computation performed by any node n requires time $O(\ell 2^{\ell} h_n |D(n)|)$ and exactly two messages are communicated between node n and its child $n_i \in D(n)$.*

3.1 Computing the Marginal Cost We now show how to use algorithms **Max-Layered-Welfare** and **Max-Split-Welfare** to compute Marginal Cost. The following information is sufficient for a receiver r to know what it pays and what it receives:

the network welfare, r's accepted bid (if any), as well as the maximum network profit obtainable when r is not admitted. We provide this information to each receiver r using a single downward phase of the algorithm from the root to the leaves of the tree that immediately follows the upward phase described previously that calculates the maximum network welfare.

The network welfare computed during the upward phase is simply passed back down the tree during the downward phase. To inform every receiver of which bid is accepted, every node stores, during the upward phase, which term of the maximization in (3) provides the largest value. The root informs its children of what configuration they are in, which, combined with the stored information, is sufficient for them to determine the configuration of their children, and so on, until every node knows what configuration it must be in to achieve the maximum network welfare. This informs every node of its closest enabled parent in the multicast tree, whether or not it itself is enabled, which layers or groups it receives from this enabled parent, and what groups or layers to forward to its children, if it has any.

More difficult is computing, for each receiver r, the maximum network profit when r is not admitted. This could of course be computed easily using one phase per receiver, but our objective is to provide all the receivers with this information using a single downward phase. We here describe how to achieve this in the layered paradigm, although it is easy to modify what we describe here to work in the split session paradigm. We assume that every node n stores the values $S_{j,k}(n_i)$, for all $n_i \in D(n)$, j and k that it learned during its computations performed in the upward phase. In addition, in the downward phase, each node n will compute a quantity $B_{j,k}(n)$, for $0 \leq j \leq \ell$ and $1 \leq k \leq h_n$. $B_{j,k}(n)$ is the maximum total network welfare possible, subject to the following conditions:

- The closest ancestor of n that is enabled is $\pi_{n,k}$.
- Node $\pi_{n,k}$ transmits the first j layers to n, but no other layers.
- If node n is an internal node of the tree, no layers are transmitted to any child of node n.
- If node n is a receiver, the profit from that receiver is not included in the network welfare.

Note that the network configurations prescribed by $B_{j,k}(n)$ for $j > 0$ are wasteful in the sense that the layers transmitted to node n are not used by any receiver that lies below n. Note that $\max_{k=1}^{h_n} B_{0,k}(r)$ is the maximum profit when receiver r is precluded from receiving any transmission.

Each node n learns every value of $B_{j,k}(n)$ from its parent during the downward phase. Our algorithm for computing Marginal Cost works as follows:

Algorithm **Layered-Marginal-Cost**

- The source node s sends every $n_i \in D(s)$ the value of $B_{j,1}(n_i)$, for $0 \leq j \leq \ell$, computed using

$$B_{j,1}(n_i) = \left(\sum_{n_i' \in D(s); n_i' \neq n_i} S_{\ell,1}(n_i') \right) - \sum_{r=1}^{j} [\mathbf{c}_{(s,n_i)}]_r.$$

  This represents the maximum profit that can be achieved in a network where exactly j layers are transmitted to n_i, but n_i transmits nothing to any of its children.

- Every node n, on receiving $B_{j,k}(n)$, for $0 \leq j \leq \ell$ and $1 \leq k \leq h_n$, from its parent, computes $B_{j,k}(n_i)$, for each $n_i \in D(n)$, $0 \leq j \leq \ell$ and $1 \leq k \leq h_{n_i}$, and sends these values to n_i. For the case where $2 \leq k \leq h_{n_i}$ (i.e., n is not enabled), we have

$$B_{j,k}(n_i) = \max_{t=j}^{\ell} \left(B_{t,k-1}(n) + \sum_{n_i' \in D(n); n_i' \neq n_i} S_{t,k}(n_i') \right) - \sum_{r=1}^{j} [\mathbf{c}_{(n,n_i)}]_r.$$

  The first term inside the maximum equals the maximum profit that can be provided from the part of the network that lies outside the subtree rooted at n when t layers are sent to n and $\pi_{n,k}$ is the closest ancestor node to n_i that is enabled. The summand term inside the maximum equals the maximum profit yielded by the subtrees of n when t layers are delivered to n. The last term subtracts the cost of delivering j layers to n_i.

For the case where $k = 1$ (such that n is enabled), we have

$$B_{j,1}(n_i) = \max_{t=j}^{\ell} \left(\max_{u=1}^{h_n} B_{t,u}(n) + \sum_{n'_i \in D(n); n'_i \neq n_i} S_{t,1}(n'_i) \right) - \sum_{r=1}^{j} [\mathbf{c}_{(n,n_i)}]_r - c_n.$$

The first term chooses which node above should be enabled to maximize the contribution from all parts of the tree that do not lie below n. We add to this the contributions of the subtrees rooted at the sibling nodes of n_i when n is enabled, subtract the cost of delivering j layers from n to n_i and the cost of enabling n.

- Each receiver r returns $\max_{k=1}^{h_r} B_{0,k}(r)$.

THEOREM 3.3. *The value returned by each receiver r in Algorithm* **Layered-Marginal-Cost** *is the maximum possible network welfare obtainable under the layered paradigm, without admitting r. Furthermore, the computation required of any node can be performed in polynomial time and each node n sends only one message to each child $n_i \in D(n)$.*

Proof. Proving the bounds on the amount of computation and communication is straightforward, and thus we here focus on proving correctness. Note first that if receiver r obtains the correct values of $B_{0,k}$, for $1 \leq k \leq h_r$, then it returns the correct value, since in the optimal configuration without r, there is some k such that $\pi_{r,k}$ is enabled (recall that the source is always enabled), and so for some value of k, $B_{0,k}(r)$ contains the maximum welfare.

We prove that all values of $B_{j,k}(n)$ are correct by induction on h_n. For the base case, consider some n that is a child of the source. The maximum network welfare obtainable without any profit provided by the subtree rooted at n is the sum of the maximum welfares obtainable in the subtrees rooted at the other children of the source. Thus, since we require that layers 1 through j are sent to n, $B_{j,1}(n)$ is the maximum profit offered by each other subtree (stored in $S_{t,1}(n'_j)$) minus the cost to send the j layers to n. For the inductive step, we assume that node n receives the correct values of $B_{j,k}(n)$ from its parent, and show that this implies that for any $n_i \in D(n)$, n sends the correct values of $B_{j,k}(n_i)$ to n_i.

For $1 \leq t \leq \ell$, let $T_{t,k}(n_i)$ be the network welfare of the optimal configuration of the set of configurations where all of the following hold: (i) t layers must reach n_i's parent, n, (ii) $\pi_{n,k-1}$ is the first enabled ancestor of n, and no receiver in the subtree rooted at n_i receives any layers. Note that node n is enabled when and only when $k = 1$. We first consider the case where $k > 1$, i.e., when node n is not enabled. In this case, the value of $T_{t,k}(n_i)$ is $B_{t,k-1}(n)$ plus the sum of the maximum welfares possible in the subtrees rooted at each $n'_i \in D(n)$, $n'_i \neq n_i$, minus the cost of sending any required layers (up to t) from $\pi_{n,k-1}$ to n'_i is included. The amount added for each n'_i is exactly the value $S_{t,k}(n'_i)$, computed during the up-phase of the algorithm **Max-Layered-Welfare**. Thus, $T_{t,k}(n_i) = B_{t,k-1}(n) + \sum_{n'_i \in D(n); n'_i \neq n_i} S_{t,k}(n'_i)$. The network configuration that achieves the correct value of $B_{j,k}(n_i)$ is the configuration that maximizes $T_{t,k}(n_i)$, subject to the requirement that $j \leq t \leq \ell$. Since $B_{j,k}(n_i)$ computes profit when j layers reach to n_i, we have $B_{j,k}(n_i) = \max_{t=j}^{\ell} T_{t,k}(n_i) - \sum_{r=1}^{j} [\mathbf{c}_{(n,n_i)}]_r$. In Algorithm **Layered-Marginal-Cost**, since we assume that node n receives the correct values of $B_{j,k}(n)$ from its parent, it follows here that node n sends the correct values of the $B_{j,k}(n_i)$ to n_i, completing the inductive step.

We next consider the case where $k = 1$, i.e., when node n is enabled. In this case, $T_{t,1}(n_i) = \max_{u=1}^{h_n} B_{t,u}(n) + \sum_{n'_i \in D(n); n'_i \neq n_i} S_{t,1}(n'_i) - c_n$, since in the optimal configuration for $T_{t,1}(n_i)$, there is some first ancestor of n, $\pi_{n,u}$, that is enabled (recall that the source is always enabled). The network configuration that achieves the correct value of $B_{1,k}(n_i)$ is the configuration that maximizes $T_{t,1}(n_i)$, subject to $j \leq t \leq \ell$. Since we require that j layers travel to n_i, we have $B_{j,1}(n_i) = \max_{t=j}^{\ell} T_{t,1}(n_i) - \sum_{r=1}^{j} [\mathbf{c}_{(n,n_i)}]_r$. In Algorithm **Layered-Marginal-Cost**, since we assume that node n receives the correct values of $B_{j,k}(n)$ from its parent, node n sends the correct values of the $B_{j,1}(n_i)$ to n_i.

4 Lower Bounds and Hardness Results

In this section, we examine the question of whether it is possible to design algorithms that performs better than those of the previous section. We first examine the communication requirements of **Max-Layered-Welfare**. For all of our communication lower bounds, we assume that all costs and bids are integers that are allowed to be anywhere in the range 0 to $2^k - 1$. We also assume that communication is restricted to the edges of the multicast tree. We first demonstrate a lower bound of $\Omega(kh)$, (provided that k is not too small), on the number of bits per edge that must be communicated when there

is a cost for enabling multicast in a tree of height h, even with only a single layer. We then demonstrate a lower bound of $\Omega(\ell k)$, (again, provided that k is not too small), on the number of bits per edge that must be communicated when there there are ℓ layers, even when there is no cost for enabling a node. Of course, the combination of these two results only imply a $\Omega(k(\ell + h))$ lower bound; an interesting open problem is determining whether or not $O(k\ell h)$ is in fact optimal. Finally, we provide evidence that the exponential dependence on the number of groups that arises in our algorithm for the split session paradigm is inherent to the problem.

4.1 Communication lower bound for enable-cost networks is $\Omega(kh)$ We consider a class of networks $\mathcal{N}$, where for each positive integer i, there is one network $N_i \in \mathcal{N}$. The network N_i has a path consisting of $2i + 1$ edges connected to the source node s. The last node of this path is connected to i other nodes besides the path. Each of these in turn is connected to two leaves of the multicast tree. There are a total of $2i$ receivers: one per leaf node. The network N_3 (with costs defined below) is depicted in Figure 1. For any network N, let $h(N)$ be the maximum length of any path from the source to a receiver (the height). Note that $h(N_i) = 2i + 3$. An input to the problem, I, consists of costs for transmission on the links of the network, costs for enabling the nodes of the network, as well as the bids of the receivers. We assume here that all inputs have only a single layer. For protocol $\mathcal{P}$, input I, and network N, let $B(N, I, \mathcal{P})$ be the average over the edges of N, of the number of bits that cross an edge using protocol $\mathcal{P}$ on N with input I.

THEOREM 4.1. *For any $N_i \in \mathcal{N}$, there is a distribution D of inputs such that for any deterministic $\mathcal{P}$, the expectation of $B(N, I, \mathcal{P})$, taken over I distributed according to D, is $\Omega((k - 2\log h(N_i) - 2)h(N_i))$. For any randomized protocol $\mathcal{P}$ that always returns the correct answer, there is some input I such that $B(N_i, I, \mathcal{P}) = \Omega((k - 2\log h(N_i) - 2)h(N_i))$.*

Proof. We define the distribution D in terms of a function $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$ that maps two i-tuples of integers to an input for the network N_i. Each of the $2i$ integers is allowed to be in the range from 0 to $\frac{2^k}{4i^2} - 1$. An important class of inputs is what we call *symmetric* inputs. Input $I = f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$ is a symmetric input if $\forall j, 1 \leq j \leq i$, $a_j = b_j$. We also say that input I is A-dominating (B-dominating), if there is any j such that $a_j > b_j$ ($a_j < b_j$, resp.), and $a_r = b_r$ for $j < r \leq i$. Note that an input is either symmetric, A-dominating, or B-dominating.

Below, we define the function f_i, but we first present a number of properties that hold for this function. Let p_j be the j^{th} node of the path, starting with the node adjacent to s. Let e_j be the $(j + 1)^{st}$ edge of the path (starting from the source s). The following claims all hold for f_i:

CLAIM 1. *If $b_1 \ldots b_i$ are fixed, but $a_1 \ldots a_i$ are not fixed, the only edge and node costs that are not fixed for inputs $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$ lie between edge e_{i+1} and the source. If $a_1 \ldots a_i$ are fixed but $b_1 \ldots b_i$ are not fixed, the only edge and node costs that are not fixed for inputs $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$ lie between edge e_{2i} and the leaves of the tree.*

CLAIM 2. *If input I is a symmetric input, then the maximum network welfare solution on input I has multicasting enabled at node p_{i+1}.*

CLAIM 3. *If input I is an A-dominating input, then the maximum network welfare solution on input I does not have multicasting enabled at node p_{i+1}.*

Before we define the function f_i and prove these claims for this function, we describe the distribution D, and show how the theorem holds for this choice of D. With probability $\frac{1}{2} - \frac{\gamma}{2(1-\gamma)}$, D forces a symmetric input, where $\gamma = \left(\frac{4i^2}{2^k}\right)^i$. In this case, $a_1, \ldots a_i$ are each chosen independently and uniformly at random from the integers in $[0 \ldots \frac{2^k}{4i^2} - 1]$, and D returns the input $f_i(a_1, \ldots, a_i, a_1, \ldots, a_i)$. With probability $\frac{1}{2} + \frac{\gamma}{2(1-\gamma)}$, $a_1, \ldots a_i, b_1 \ldots b_i$ are each chosen independently and uniformly at random, and D returns the input $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$. Choosing the input in this manner ensures that every pair of inputs is possible, and that the probability of the input being symmetric is exactly $\frac{1}{2}$. Claims 1, 2, and 3 imply the following:

LEMMA 4.1. *Let e be any edge e_j, for $i + 1 \leq j \leq 2i$. For any deterministic protocol $\mathcal{P}$, when the input is chosen using the distribution D, the expected number of bits that cross e during $\mathcal{P}$ is $\frac{i(k-2\log i-2)}{2}$.*

Proof. Consider a pair of distinct inputs $I = f_i(a_1, \ldots, a_i, a_1, \ldots, a_i)$ and $I' = f_i(a'_1, \ldots, a'_i, a'_1, \ldots, a'_i)$ where I and I' are both symmetric. By Claim 2, on each of these inputs, the optimal solution has node p_{i+1} multicast enabled. W.l.o.g., assume that $I'' = f_i(a_1, \ldots, a_i, a'_1, \ldots, a'_i)$ is A-dominating. Thus, by Claim 3, on input I'', the optimal solution does

not have node p_{i+1} multicast enabled. Therefore, using Claim 1 and the standard but classical communication complexity argument of Yao [33], we can show that the communication transcript across e on I must be different from the transcript on I'. Otherwise, node p_{i+1} would be incorrectly multicast enabled on input I''. The lemma then follows from the fact that there are $(\frac{2^k}{4i^2})^i$ possible equally likely symmetric inputs, and the input is symmetric with probability $\frac{1}{2}$. Thus the entropy of the communication transcript over e is at least $\frac{i(k-2\log i-2)}{2}$.

The theorem now follows for deterministic protocols, since from the linearity of expectation, the expected sum of the number of bits that must cross all of the edges e_j, where $i+1 \leq j \leq 2i$, is at least $\frac{i^2(k-2\log i-2)}{2}$. Since the total number of edges in N_i is $O(i)$, we get that the expectation of $B(N_i, I, \mathcal{P})$, when I is distributed according to D, is $\Omega(i(k-2\log i-2)) = \Omega(h(N_i)(k-2\log h(N_i)-2))$. For randomized algorithms, we use Yao's Lemma [34], which tells us that any lower bound for deterministic algorithms running on a known distribution of inputs also provides a lower bound for randomized algorithms running on a worst case input.

Thus, to prove the theorem, we only need to describe f_i, and demonstrate that Claims 1, 2, and 3 hold. We next define the function $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$. The main difficulty in doing so comes in ensuring that if there is any j such that $a_j > b_j$ and $a_r = b_r$ for $j < r \leq i$, then regardless of the values of $a_1, \ldots, a_{j-1}$ and $b_1, \ldots, b_{j-1}$, the node p_{i+1} is not multicast enabled in the optimal allocation of $f_i(a_1, \ldots, a_i, b_1, \ldots, b_i)$. For example, it is not difficult to define a function such that if $\sum_{j=1}^{i} a_j > \sum_{j=1}^{i} b_j$, then p_{i+1} is not multicast enabled, but this is not sufficient for our purposes.

Call the i nodes that are adjacent to the last node of the path $n_1 \ldots n_i$. The cost of enabling multicast at node n_r is $\sum_{j=0}^{r-1} 2b_{i-j}$. Let $M = 2^k - 1$. The cost of enabling multicast at any node p_j, for $i+2 \leq j \leq 2i+1$ is always M, and the cost of using any edge between node p_{i+1} and the leaves of the tree is always 0. The cost of using the edge incident to the source s is always M. The cost of using e_j, for $1 \leq j \leq i$, is $2a_j$. Let $c_1(j) = \sum_{r=1}^{i-j} 2ra_{i-r}$, for $1 \leq j \leq i-1$, and $c_1(j) = 0$ for all other j. Let $c_2(j) = \sum_{r=1}^{j-1}(4i-2)a_r$ for $2 \leq j \leq i+1$, and $c_2(j) = 0$ for all other j. The cost of enabling multicast at node p_j, for $1 \leq j \leq i$, is $c_1(j) + c_2(j)$. The cost of enabling multicast at node p_{i+1} is $c_2(i+1) - 1$. Note that because of the assumptions that $a_j < \frac{2^k}{4i^2}$ and $b_j < \frac{2^k}{4i^2}$, the costs of all edges and nodes in each input defined by f_i is less than M. We also specify that on every input, each receiver makes a bid of M. The network N_3 is depicted with costs in Figure 1. The proof of Claim 1 holds trivially from the description of the function f_i. We next prove the other two claims.

Proof (of Claim 2): We first point out that the network welfare is always maximized by sending the multicast session to every receiver. Thus, maximizing network welfare is equivalent to minimizing the cost to do so. In the case of a symmetric input, the optimal solution is when node p_{i+1} is multicast enabled, and no other node is multicast enabled. This yields a cost of $OPT = M + \sum_{r=1}^{i} 4ia_r - 1$. Now consider any solution where node p_{i+1} is not enabled. If no other node p_j is enabled, then the edge incident to the source is used i times, leading to a cost of $iM > OPT$. If node p_j is enabled, for $j > i+1$, there is a cost of M for enabling that node, plus the cost of the edge incident to the source, and thus the cost is at least $2M > OPT$.

Otherwise, p_j is enabled, for some $j \leq i$. If there is more than one such node enabled, let p_j be the node with the highest value of j. The cost of enabling p_j is $c_1(j) + c_2(j)$. In addition, the cost of using the edges of the path is $M + \sum_{r=1}^{i} 2a_r + (2i-t-1)\sum_{r=j}^{i} 2a_r$, where t is the number of indices r such that node n_r is multicast enabled. The cost of enabling those nodes is minimized by enabling nodes n_1 through n_t, for a cost of $\sum_{r=1}^{t}(t-r+1)2a_{i-r+1}$. Thus, the cost is minimized by setting $t = i-j+1$. Summing all the terms we see that the total cost is at least $M + \sum_{r=1}^{i} 4ia_r > OPT$. ■

Proof (of Claim 3): Assume the input is A-Dominating, and let j be such that $a_j > b_j$ and $a_r = b_r$ for $j < r \leq i$. The cost of any solution where p_{i+1} is enabled is at least $M + \sum_{r=1}^{i} 4ia_r - 1$. However, the cost of the solution where the nodes p_j and $n_1, \ldots, n_{i-j+1}$ are multicast enabled, but no other nodes are enabled, has cost $M + \sum_{r=1}^{i} 4ia_r - 2a_j + 2b_j$, which must be strictly smaller. ■

4.2 Communication lower bound for the layered paradigm is $\Omega(k\ell)$We next provide our lower bound for the case where there is no cost for enabling any node of the network, but there are ℓ layers in the multicast session. We call a network topology *2-tall* if there is no leaf of the multicast tree that is directly connected to the source.

THEOREM 4.2. *For any 2-tall network topology N, there is a distribution D of inputs such that for any deterministic protocol $\mathcal{P}$, the expectation of $B(N, I, \mathcal{P})$, taken over I distributed according to D, is $\Omega(\ell(k - \log\Delta))$, where Δ is the maximum degree of any node in N. For any randomized protocol $\mathcal{P}$ that always returns the correct answer, there is some*

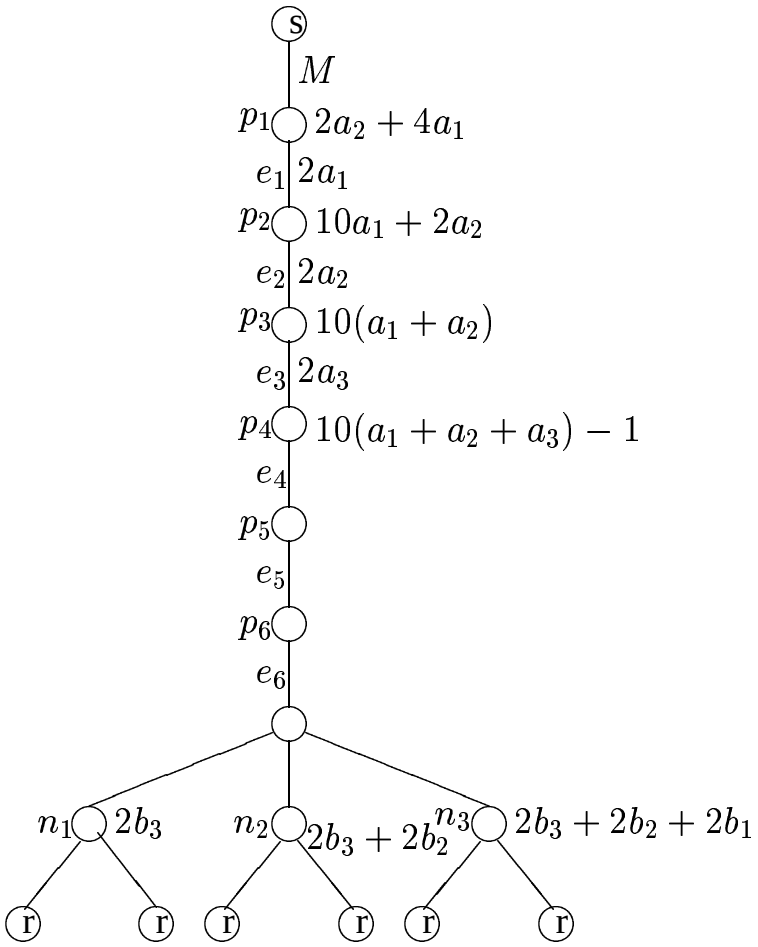

Figure 1: The network N_3. Node (edge) names are depicted to the left of the node (edge, resp.). The cost of enabling multicast at a node is depicted to the right of the node, and the cost of using an edge is depicted to the right of the edge. When the cost of enabling multicast at a node is not depicted, it is M. An edge without a cost has cost 0.

input I such that $B(N, I, \mathcal{P}) = \Omega(\ell(k - \log \Delta))$.

Proof. In order to prove this bound, we first partition the edges of the network N into smaller subsets. In particular, let a *rake* be a subset of N consisting of a designated node called a *root* node, a path of at least one edge from the root to a node called the *center* of the rake, and a set of *leaves* of the rake, each of which is adjacent to the center of the rake. A rake must have at least one leaf. The leaves of the rake are required to also be leaves of the original multicast tree, and the root of the rake must be the node in the rake that is closest to the source of the multicast tree. Let a *rake partition* be a partition of the edges of N such that the set of edges in each group of the partition and their incident nodes form a rake. A node is allowed to appear in more than one rake. However, since the root of the rake must be the closest node to the source, a node of the multicast tree can only serve as the center of a single rake, since otherwise the two rakes would also share (at least) the first edge towards the root from the center. This implies that for any pair of rakes r_1 and r_2, the set of nodes serving as leaves for r_1 is disjoint from the set of nodes serving as leaves for r_2.

CLAIM 4. *For any 2-tall multicast tree network N, there is a rake partition.*

Proof. We give an algorithm that constructs such a partition. We define a *rake forest* to be a forest where every node corresponds to a node of N, and if there is an edge in the rake forest between nodes u and v, then there must be an edge between the nodes corresponding to u and v in N. Only one edge of the forest can correspond to an edge of N, but multiple nodes can correspond to a nodes of N. Every tree of the rake forest has a root node that corresponds to the node in N closest to the source. Also, no vertex adjacent to such a root is allowed to be a leaf. The network N defines a rake forest with a root at the source node (since it is 2-tall), and we also allow for a forest with no edges (the empty forest).

We show that given a non-empty rake forest, we can remove from the forest a set of edges defining a rake in such a way that the resulting forest is still a rake forest. By repeating this process until the rake forest is empty, we construct a rake partition. Hence, this proves that a rake forest is also a rake partition. To construct a rake r_c from a rake forest, we start by picking the root of any tree in the forest. Choose some edge e incident to the root, and include e in r_c, and remove e from the rake forest. Let v be the node incident to e that is not the root. If v has any children that are leaves, all edges that are incident to both v and a leaf are added to r_c, and removed from the rake forest. In that case, the construction of r_c is complete. We also remove any nodes that no longer have any incident edges, and for each v' adjacent to v such that v' is not a leaf, a new node that corresponds to the same node of N as v is placed adjacent to v', and this new node serves as the root of a new tree. Note that the resulting forest is still a rake forest, since any new roots that are formed cannot be incident to leaves.

If v has no children that are leaves, then let e' be any edge remaining in the rake forest incident to v (one such edge must exist, since v is not a leaf). Add e' to the rake, and remove it from the rake forest. Any other edges incident to v are detached from v, and v is replaced by a new node that corresponds to the same node of N as v. The new node becomes the root of the resulting tree. Note that again, no node adjacent to the new root can be a leaf node. This process is then repeated from the node v' that is incident to e', until we reach a node that is adjacent to at least one leaf, where the first case applies. At the end of this process, we have added a rake to the rake partition, while at the same time maintaining the properties of the rake forest.

We now provide a probability distribution over inputs that leads to the lower bound. To do so, assume that we have some rake partition of N. We distinguish between *long rakes*, where the number of edges on the path from the root to the center of the rake is at least as large as the number of leaves of the rake, and *wide rakes*, where the number of edges from the root to the center of the rake is less than the number of leaves. In the input distribution, the only edges that may have non-zero cost are the edges incident to the root of each rake. The only receivers that may have non-zero bids are the leaves. Also, for each long rake, there is a designated leaf that may have non-zero bids; all other leaves of a long rake always make zero bids.

With probability 1/2, the input is a *symmetric* input. In this case, for each long rake, the designated leaf makes a bid such that the additional amount bid for each layer is chosen independently and uniformly at random from the range $[0, 2^k - 1]$. The additional cost for each layer on the edge incident to the root in that rake is exactly equal to the corresponding bid made by the designated leaf. For a wide rake with t leaves, each leaf makes a bid such that the additional amount for each layer is chosen independently and uniformly at random from the range $[0, (2^k - 1)/t]$. The cost for each layer of the edge incident to the root is equal to the sum of the bids for that layer made by the leaves of that rake. For each rake, the costs and bids are chosen independently.

With probability 1/2 the input is not a symmetric input. In that case, we first construct a symmetric input. Then, some rake is chosen uniformly at random. If this rake is a long rake, the bids of the designated leaf are changed so that the

additional amount bid for each layer is a new random number in the range $[0, 2^k - 1]$. In the case of a wide rake, some leaf is chosen uniformly at random. The additional amount bid for each layer by that leaf is changed to be a new random number in the range $[0, (2^k - 1)/t]$. The theorem now follows from the following lemma:

LEMMA 4.2. *Let R be a set of edges that form a rake of the rake partition. For any deterministic algorithm running on this probability distribution, the expected number of bits that need to traverse the edges of R is $\Omega(\ell|R|(k - \log \Delta))$.*

Proof. We use the following two claims:

CLAIM 5. *Let e be any edge incident to a leaf of a wide rake. For the described distribution of inputs, the expected number of bits that cross e in any deterministic algorithm is at least $\Omega(\ell(k - \log \Delta))$.*

CLAIM 6. *Let e be any edge on the path from the root to the designated leaf of a long rake. For the described distribution of inputs, the expected number of bits that cross e in any deterministic algorithm is at least $\Omega(\ell(k - \log \Delta))$.*

For wide rakes, the lemma follows from Claim 5, the linearity of expectation, and the fact that in a wide rake, at least half the edges are incident to leaves. For long rakes, the lemma follows from Claim 6, the linearity of expectation, and the fact that in a long rake, at least half the edges are on the path between the node adjacent to the root and the designated leaf.

The details of the proofs of the two claims depend on the exact rules for how ties are broken. We here describe the case where if there are multiple ways to achieve the maximum network welfare, the network transmits the highest set of layers that does so. This implies that if the maximum profi t is zero, but it is possible to achieve this by transmitting at some layer, the network transmits at that layer. Other tie breaking rules are similar. The proofs of Claim 5 and Claim 6 follow from the classical, two party communication complexity lower bound technique of [33] (see also [22]) using fooling sets. However, since the details are slightly complicated by the fact that we may have many more than two parties present, and the fact that we have multiple layers, we provide a brief sketch of the argument.

For Claim 5, one of the two communication complexity parties is the leaf L incident to edge e. The communication medium is the edge e, and the other communication complexity party is the remainder of the network. We demonstrate that the transcript of the bits that cross e must be different on all input bids to L that are possible as part of a symmetric input. To see this, consider two symmetric inputs I_1 and I_2 such that L receives a different set of bids on I_1 than on I_2. Let b_i^j be the total amount bid by L on input I_j for layers up to i. Let r be the value of i that maximizes $|b_i^1 - b_i^2|$. If $r = \ell$ (i.e., r is the highest layer overall), then we assume w.l.o.g. that $b_r^1 > b_r^2$. Otherwise, assume w.l.o.g. that $b_r^1 < b_r^2$.

Let I_1^2 be the input which is identical to I_1, with the exception that the leaf L receives the bids it would receive on input I_2. Note that I_1^2 is a possible (not symmetric) input to the network. Using the argument from [33], we see that if the communication over the edge e on the inputs I_1 and I_2 is identical, then at the end of the algorithm, the only node in the network that is able to distinguish input I_1^2 from input I_1 is the node L. This means that the network would send the highest layer stream to all leaves in the network, resulting in a profi t of 0. However, in the case where $r < \ell$, if the network sends at layer r to all nodes below the edge incident to the root of the rake containing e, then all layers outside the rake containing e break even, and that rake achieves a profi t of $b_r^2 - b_r^1$. In the case where $r = \ell$, the network would still send at the highest layer to all leaves, even though this results in a loss. The network would do better by not sending at all.

Thus, the communication over the edge e must be different for any pair of symmetric inputs where the bids received by L are different. The probability of an symmetric input is 1/2, and on an symmetric input there are at least $2^{\ell k}/\Delta^\ell$ input bids to L that are all equally likely, the entropy of the communication over e is at least $\frac{\ell(k - \log \Delta)}{2}$, and thus the expected number of bits that cross that edge is in fact $\Omega(\ell(k - \log \Delta))$. For the proof of Claim 6, the argument is similar, except that the two parties for the communication complexity argument are the entire network above the edge e', and the entire network below the edge e'. Thus, for the inputs analogous to I_1 and I_2, we can create the input analogous to I_1^2 by providing inputs from I_1 to the network above the edge e' and from I_2 to the nodes below the edge.

To complete the proof of the theorem for deterministic algorithms, we see that from the linearity of expectation, the expected number of bits that traverse the edges of the network is $\Omega(\ell m(k - \log \Delta))$. For randomized algorithms, we use Yao's Lemma [34], which tells us that any lower bound for deterministic algorithms running on a known distribution of inputs also provides a lower bound for randomized algorithms running on a worst case input.

4.3 Hardness of the split session paradigm We next examine the computational complexity of maximizing network welfare in the split session paradigm. The algorithm **Max-Split-Welfare** has a running time that is exponential in ℓ, but

runs in polynomial time for any fixed ℓ. We here demonstrate that if ℓ is given as part of the input, then the problem becomes NP-Hard to even approximate. Thus, we should not expect a significantly more efficient algorithm for maximizing network welfare in the split session paradigm.

THEOREM 4.3. *If ℓ is specified as part of the input, then there is no polynomial time $\ell^{1-\epsilon}$-approximation algorithm, for any $\epsilon > 0$, for the problem of maximizing network welfare in the split session paradigm, unless $NP = ZPP$. This holds even if there is no cost for enabling multicast at the nodes of the network.*

Proof. We show that such an approximation algorithm can be converted into a polynomial time $|V|^{1-\epsilon}$-approximation algorithm for the largest independent set in a graph. Since [15] demonstrates that such an algorithm only exists if $NP = ZPP$, the theorem follows. Given an arbitrary input graph $G = (V, E)$, we construct an input for the network welfare maximization problem. This input can be constructed for any network topology where there is a subset R of $|E| + |V|$ receivers that share the first edge from the source, but each has a distinct last edge from the source. Furthermore, this input can be constructed even if we are subject to the requirement that edge costs are proportional to the rate of the flow crossing that edge. In other words, for any edge e, there is a constant $c(e)$, such that the cost for transmission of any group g to use e is $B(g)c(e)$, where $B(g)$ is the rate of g.

For each vertex $v \in V$, we have one group $g(v)$. All that we require of the group rates is that $B(g(v)) \geq 2 \cdot deg(v) - 1$, where $B(g(v))$ is the rate of $g(v)$ (we assume that there are no degree 0 vertices in V: such vertices can easily be dealt with separately). For each edge $e = (u, v) \in E$, we have one receiver $r(e) \in R$, and for each vertex $v \in V$, we have one receiver $r(v) \in R$. For each edge of the communication network, the cost of a group using that edge is either equal to the rate of the group, or 0. In particular, the cost for sending $g(v)$ over the shared first edge from the source is $B(g(v))$, the cost of $g(v)$ traversing the last edge from the source to any receiver in R is also $B(g(v))$, and the cost of all other edges in the network is 0. Each receiver $r(u)$ bids $B(g(v))$ for group $g(v)$, for $u \neq v$, and $2B(g(u)) - 2 \cdot deg(u) + 1$ for group $g(u)$. Each receiver $r(e)$, where $e = (u_1, u_2)$, bids $B(g(v))$ for group $g(v)$, for $u_1, u_2 \neq v$, and $B(g(v)) + 2$, for $v \in \{u_1, u_2\}$.

We next show that maximizing the network welfare for the described input is equivalent to finding the largest independent set in the graph G. Assume first that G has an independent set I of size $|I|$. To convert this into a solution to the network welfare maximization problem, the network sends group $g(v)$, for each $v \in I$, to each $r(e)$ such that $e = (u, v)$ for any u. This is a valid solution, since I is an independent set. Also, the network sends such $g(v)$ to each $r(v)$. The revenue from group $g(v)$ for $v \in I$ is $2B(g(v)) - 2 \cdot deg(v) + 1$ from receiver $r(v)$ if $v \in V$, plus $deg(v)(B(g(v)) + 2)$, from all $r(e)$ where $e = (u, v)$ or $e = (v, u)$ for some vertex u. The total cost for group $g(v)$ is $B(g(v))(deg(v) + 2)$ (to deliver the group to receiver $r(v)$, the group must pass through the shared edge, to the edge to reach $r(v)$, and then to the $deg(v)$ receivers $r(e)$ where $e = (v, u)$ for some u), for a total welfare of 1 for each such group. Thus, the total welfare is exactly I.

Also, any solution to the network welfare maximization problem with profit P can be converted to an independent set of size P. To do so, we consider only those groups that result in a profit. For any such group $g(v)$, the only way to achieve a profit is if every receiver that bids more than $B(g(v))$ actually receives group $g(v)$. If this is done, then the total profit for group $g(v)$ is exactly 1. No two groups $g(u_1)$ and $g(u_2)$ that have a profit of 1 can correspond to vertices that share an edge, or else $r(e)$ where $e = (u_1, u_2)$ would join both groups $g(u_1)$ and $g(u_2)$. Since under the split paradigm, a receiver joins at most one group, the set of groups that achieve a profit must correspond to an independent set of size at least P. The theorem follows.

5 The multiple tree case

In this section, we examine the effect of removing the assumption that there is a single multicast tree that defines the routing for all groups or layers. In particular, we consider the case where for every layer or group, there is a single fixed tree that must be used, but the trees for the different layers or groups do not have the same tree, either because they originate from different sources, or because they use a different routing scheme. We refer to this as the *multiple tree* case. This is actually a property that is desirable in practice, since it helps balance the network load. Unfortunately, many problems become NP-Hard with this relaxation, even if there is no cost for enabling multicasting. Throughout this section we assume a model where there is no such cost.

We first demonstrate that maximizing network welfare becomes more difficult for the split session paradigm in the multiple tree case. Recall that in the single tree case, we have an algorithm for computing Marginal Cost in the split session paradigm that requires computation that is exponential in the number of sessions, but is polynomial for the case of a constant number of sessions.

THEOREM 5.1. *In the multiple tree case, the problem of maximizing network welfare using the split session paradigm is NP-Hard for any fixed constant number of groups larger than 12. This holds even if there is no cost for enabling multicast at any node of the network.*

Proof. We reduce from BOUNDED-3-SAT, the version of 3-SAT where every variable appears at most 5 times, and every clause has exactly 3 literals. This restriction of 3-SAT is NP-Complete [12]. Given a bounded 3-SAT formula Φ, we first assign a color to every literal, such that if two literals are assigned the same color, then they never appear in the same clause, and they are not negated versions of the same variable. This can be done using 12 colors in a greedy fashion. In particular, for each variable in turn, we assign the colors for both literals of that variable at the same time. Since a variable's literals can appear in a total of at most 5 clauses, at most 10 colors are ruled out by previous assignments to the other literals in those 5 clauses. Since there are 12 colors, there are always at least two colors available to color the variable and its complement during its turn.

We then construct the welfare maximization problem. Here, we provide the proof for the case where there is a single source, and each group has an arbitrary multicast tree rooted at that source. However, it is not difficult to modify this reduction so that it holds when the input is required to be a weighted directed graph, the source for each group is placed at some vertex of the graph, and the tree for a group is defined by shortest paths from the source to all the receivers. Furthermore, we assume here that for every edge of the communication network, the cost to use that edge is the same for all groups, although this is also easy to modify so that costs are proportional to rate. In the network, there are two nodes for each variable x_i, labeled x_i^1 and x_i^2. There are also five nodes for each clause c_j, labeled c_j^1 through c_j^5. The edges of the network are as follows. There is an edge from the source node to x_i^1, for each variable x_i. These edges have cost 6. There is an edge from x_i^1 to x_i^2 for each variable x_i. These edges have no cost. For any variable x_i and clause c_j, there is an edge from x_i^1 to c_j^1 if x_i appears in c_j. These edges have cost 2. For each clause c_j, there is an edge of no cost from c_j^1 to c_j^k, for $k \in \{2,3,4,5\}$. Finally, for each clause c_j, there is an edge from the source to c_j^k, for $k \in \{2,3,4,5\}$. These edges have cost ∞. For each variable x_i, there is an edge from the source to x_i^2 with cost ∞.

There are four receivers for each clause c_j, one at each c_j^k, $k \in \{2,3,4,5\}$. There is also one receiver for each variable x_i at x_i^2. There are no other receivers. The receivers at any c_j^k, for $k \in \{3,4,5\}$ each make a bid of 1 for each group that is successfully delivered to that receiver. The receivers at any c_j^2 each bid 2 for all groups. The receivers at any x_i^2 each bid 6 for all groups. There are a total of 12 groups, one for each of the colors. We next describe the multicast trees for each group. For group t, the tree is routed from the source to every x_i^1 whenever either of the literals x_i or $\bar{x_i}$ is assigned to color t. Note that since x_i and $\bar{x_i}$ are assigned different colors, there are exactly two groups that are routed from the source to each node x_i^1.

From each x_i^1 that group t is routed to, t continues to x_i^2. From x_i^1, t is also routed to every c_j^1 where c_j is a clause that contains the literal of x_i that is assigned to color t. Note that since literals appearing in the same clause are assigned to different groups, the routing for any group t is in fact a tree (even though the underlying network is not a tree). Also note that exactly 3 groups are routed to each c_1. All three of these groups are routed on the edge from c_j^1 to c_j^2. Also from c_j^1, the three groups diverge: one group each goes to c_j^3, c_j^4, and c_j^5. The trees described thus far allow exactly one group to be routed to each node c_j^k, for any c_j and $k \in \{3,4,5\}$ (specifically, the group that is associated with the color of one of the literals within the clause c_j). The remainder of the groups are routed to c_j^k by using the edge directly from the source to c_j^k. Similarly, the remainder of the groups are routed to c_j^2 and x_i^2 using the edge directly from the source. This completes the construction of the multicast problem. The network is depicted in Figure 2. The theorem now follows from the following two claims; their proofs appear in [1].

CLAIM 7. *If the formula Φ is satisfiable, then the resulting multicast problem can achieve network welfare at least C, where C is the number of clauses in Φ.*

Proof. Let X be a satisfying assignment to the variables of Φ. For each true literal $x_i \in X$ or $\bar{x_i} \in X$ that is assigned to color t, route group t from the source to x_i^1, and then to x_i^2. Since there is only one true literal per variable, exactly one group is routed to each x_i^2. Since X is a satisfying assignment, for each clause c_j, there is at least one x_i^1 that receives a group that can continue from x_i^1 to c_j^1. Choose any one such group, route it from x_i^1 to c_j^1. That group is then routed from c_j^1 to c_j^2, and also to one of c_j^3, c_j^4, or c_j^5, whichever is possible. The cost of using the edge from the source to each x_i^1 is offset by the payment of the receiver at x_i^2, and the cost of using an edge from any x_i^1 to any c_j^1 is offset by the payment of the receiver at c_j^2. Thus, the total network welfare is exactly 1 per clause, for a total of C.

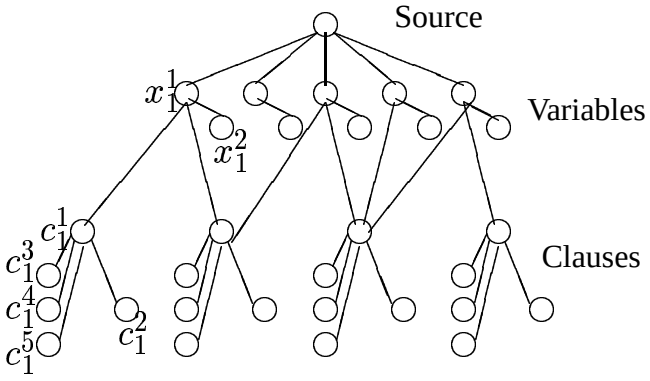

Figure 2: The network used in the proof of Theorem 5.1.

CLAIM 8. *If, in the multicast problem resulting from the formula Φ, it is possible to achieve network welfare at least C, where C is the number of clauses in Φ, then the formula Φ is satisfiable.*

Proof. Consider first any node x_i^1. If there is any such node that receives 2 groups from the source, then only one of those groups is routed to the receiver at x_i^2. Call the other group g. Since each variable appears in at most 5 clauses, the total network welfare provided by g in its multicast tree below x_i^1 is at most 5, but the cost of using the edge from the source to x_i^1 is 6, and so not routing g to x_i^1 and any descendants of x_i^1 in its multicast tree would increase the network welfare of the solution by at least 1. Thus, we can assume that we have a solution with welfare at least C, where each x_i^1 receives at most one group.

Using a similar argument, we see that we can assume that each node c_j^1 receives at most one group. Since the network welfare provided by a group in its multicast tree below c_j^1 is at most 1, if we have a solution with network welfare C, we can assume that we have such a solution where each node x_i^1 receives at most 1 group, and each node c_j^1 receives exactly 1 group. Thus, if we set each variable x_i to make the literal true that is assigned to the group that is routed from the source to x_i^1, and arbitrarily set any variable x_i where there is no group routed x_i^1, then we are guaranteed that we have at least one true literal per clause. Thus, Φ is satisfiable.

5.1 An algorithm for the layered paradigm

The increased difficulty in finding good solutions for the split session paradigm might suggest that maximizing welfare in the layered paradigm in the multiple tree case is also NP-Hard, but this turns out to not be the case.We next provide a polynomial time algorithm for this problem by demonstrating that the welfare maximization problem can be stated as a totally unimodular integer program. Such integer programs can be solved in polynomial time using linear programming techniques (see for example [27]). This leads to a centralized polynomial time algorithm that finds the optimal solution. This may require significant communication overhead in a distributed setting, and thus an interesting open problem is finding a communication efficient distributed algorithm for this problem. Some approaches to solving linear programming problems in a distributed fashion have been studied in [2], but in our scenario, the objective function is distributed across the users, and thus the techniques from [2] do not apply here. We also note that this algorithm can also be used to compute the Marginal Cost allocation in polynomial time by computing the cost for every receiver individually.

DEFINITION 1. *A* comparison integer program *is a restricted integer program where every constraint must be of the form $\alpha \leq \beta$, where α and β can each be any variable of the integer program, or any integer. We allow an arbitrary linear objective function.*

For any variable x_i in a comparison integer program, we say that the *implied upper bound* on x_i is c if c is the minimum value such that for every feasible solution, $x_i \leq c$. Similarly, we say that the *implied lower bound* on x_i is c if c is the maximum value such that for every feasible solution, $x_i \geq c$. Note that we can compute the implied upper bound on any variable by defining a graph, where the vertices of the graph are the variables and constraint integers of the integer program, and the constraint $\alpha \leq \beta$ is represented by an edge from β to α. The implied upper bound is simply the smallest constraint integer c such that x_i is reachable from c. The implied lower bound can be found in a similar fashion.

THEOREM 5.2. *For any feasible and bounded comparison integer program, an optimal solution can be found by taking any optimal solution to the linear programming relaxation of the problem, and setting every variable that is not equal to either its implied upper bound, or its implied lower bound, equal to its implied upper bound.*

We could also set every variable that is not equal to either its implied upper bound or its implied lower bound equal to its implied lower bound; the proof of this is similar to the one below for the case where the implied upper bound is used.

Proof. Let s be any optimal solution to the linear programming relaxation of the problem. Let $V(s)$ be the set of variables in s that are not set to either their implied upper bound or their implied lower bound. The proof is by induction on $|V(s)|$. The base case is the trivial one where $|V(s)| = 0$: since every implied upper and lower bound must be an integer, if $|V(s)| = 0$, then s must also be an optimal solution to the integer program. For the inductive step, assume that the theorem is true for any s such that $|V(s)| \leq k$. We show that the theorem is also true for any s such that $|V(s)| = k + 1$.

Let $x(s)$ be the value of the variable x in the solution s. For any real number t, let $s \oplus t$ be the solution such that $x(s \oplus t) = x(s) + t$ for $x \in V(s)$, and $x(s \oplus t) = x(s)$ for $x \notin V(s)$. Denote the implied upper bound (implied lower bound) on the variable x by $b^+(x)$ ($b^-(x)$, respectively). Let

$$b^+(s) = \min_{x \in V(s)} b^+(x) - x(s),$$

and

$$b^-(s) = \min_{x \in V(s)} x(s) - b^-(x).$$

CLAIM 9. *Let s be any optimal feasible solution to the linear programming relaxation. For any t such that $-b^-(s) \leq t \leq b^+(s)$, $s \oplus t$ is also an optimal feasible solution.*

Proof. We first show that for any such t, $s \oplus t$ is a feasible solution. We here provide the proof for the case where $t > 0$; the case where $t < 0$ is similar. Assume for the sake of contradiction that a constraint of the form $x_i \leq x_j$ is violated in $s \oplus t$. Since that constraint was not violated in the solution s, $x_i(s \oplus t) > x_i(s)$, and thus $x_i \in V(s)$. It must also be the case that $x_j \notin V(s)$, since otherwise $x_i(s \oplus t) - x_i(s) = x_j(s \oplus t) - x_j(s)$, and the constraint would not be violated in $s \oplus t$. However, if $x_i \leq x_j$ is a constraint, then $b^+(x_i) \leq b^+(x_j)$. Thus, if $x_i(s \oplus t) > x_j(s \oplus t) = b^+(x_j)$, then $x_i(s \oplus t) - x_i(s) > b^+(x_i) - x_i(s)$. This contradicts the assumption that $t \leq b^+(s)$. The only other type of constraint that could be violated in $s \oplus t$ is a constraint of the form $x_i \leq c$, for an integer c. However, this would also contradict the assumption that $t \leq b^+(s)$.

To see that $s \oplus t$ is also an optimal solution, note that we can assume that $|V(s)| > 0$, since otherwise the claim is trivial. This implies that $b^-(s) > 0$, and $b^+(s) > 0$. Thus, there are real numbers $-b^-(s) \leq t_1 < 0$ and $0 < t_2 \leq b^+(s)$ such that both $s \oplus t_1$ and $s \oplus t_2$ are feasible solutions. Let $J(s)$ be the value of the objective function for the solution s. Since the objective function is linear, it must be the case that $J(s \oplus t) = J(s) + \alpha t$, for some constant α. If $\alpha > 0$, then $J(s \oplus t_1) > J(s)$. If $\alpha < 0$, then $J(s \oplus t_2) > J(s)$. Since $J(s)$ is by definition optimal, we must have that $\alpha = 0$, and that $J(s \oplus t_1)$ and $J(s \oplus t_2)$ are also optimal.

To complete the inductive proof of the theorem, assume that s is an optimal solution to the linear programming relaxation such that $|V(s)| = k + 1$. By Claim 9, the solution $s \oplus b^+(s)$ is also an optimal feasible solution to the linear programming relaxation. It must also be the case that $|V(s \oplus b^+(s))| \leq k$. Thus, by the inductive hypothesis, $s \oplus b^+(s)$ can be converted into an optimal solution to the integer program by setting every variable $x \in V(s \oplus b^+(s))$ to $b^+(x)$. However, the resulting solution is the same solution that is obtained by starting with s, and setting every variable $x \in V(s)$ to $b^+(x)$. This completes the induction.

COROLLARY 5.1. *There is a polynomial time algorithm that maximizes network welfare in the layered paradigm of the multiple tree case.*

Proof. We show that this problem can be expressed as a totally unimodular integer program. For each edge e and each layer i that can travel on edge e, we have a variable x_{ei}, such that $0 \leq x_{ei} \leq 1$. If $x_{ei} = 1$, this represents that layer i traverses edge e; otherwise $x_{ei} = 0$. To ensure that we have a valid multicast tree, for any edges e and e' such that in the tree for layer i, edge e connects a node n to its parent, and e' connects n to a child, we include the constraint $x_{e'i} \leq x_{ei}$. To ensure that we have a valid layered flow, for any pair of edges e and e' such that e is the last edge that brings a layer i to a

receiver in the multicast tree, and e' is the last edge that brings a layer $i+1$ to that same receiver, we have the constraint $x_{e'(i+1)} \leq x_{ei}$. Any feasible solution to this integer program defines a valid set of layered multicast trees. Our objective function is to maximize

$$\sum_i \sum_{e \in leaf(i)} x_{ei} \cdot p_{e,i} - \sum_{e,i} x_{ei} \cdot c_{e,i}$$

where $c(e,i)$ is the cost of sending layer i on edge e, $leaf(i)$ is the set of edges that bring layer i to its receivers, and $p_{e,i}$ is the profit provided by bringing layer i to the receiver served by e.

Note that this integer program has a constraint matrix A such that all entries are ± 1, each row of A has at most two entries, and if a row contains two entries one is +1 and the other is -1. Thus, the determinant of any submatrix of A has either no nonzero terms, a single nonzero term equal to ± 1, or one term equal to 1 and one term equal to -1. Therefore, A is in fact totally unimodular.

References

[1] M. Adler and D. Rubenstein. Pricing Multicasting in More Practical Network Models. Technical report, University of Massachusetts UM-CS-2001-016, April 2001.

[2] Y. Bartal, J. Byers, and D. Raz. Global Optimization Using Local Information with Applications to Flow Control. In 38th IEEE Symp. on Foundations of Computer Science, pages 303–312, 1997.

[3] J. Byers, M. Luby, and M. Mitzenmacher. Fine-Grained Layered Multicast. In *Proceedings of IEEE INFOCOM'01*, Anchorage, AK, April 2001.

[4] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A Digital Fountain Approach to Reliable Distribution of Bulk Data. In *Proceedings of SIGCOMM'98*, Vancouver, Canada, September 1998.

[5] Y. Chawathe, S. McCanne, and E. Brewer. RMX: Reliable Multicast for Heterogeneous Networks. In *Proceedings of IEEE INFOCOM'00*, Tel-Aviv, Israel, March 2000.

[6] S. Cheung, M. Ammar, and L. Xue. On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution. In *Proceedings of IEEE INFOCOM'96*, San Francisco, CA, March 1996.

[7] Y. Chu, S. Rao, and H. Zhang. A Case for End System Multicast. In *Proceedings of ACM SIGMETRICS'00*, Santa Clara, CA, May 2000.

[8] S. Deering and D. Cheriton. Multicasting routing in datagram internetworks and extended LANs. *ACM Trans. on Computer Systems*, 8(2):85–110, May 1990.

[9] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment Issues for the IP Multicast Service and Architecture. *IEEE Network Magazine*, Jan/Feb 2000.

[10] eMule Team. eMule file-sharing software. http://www.emule-project.net.

[11] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the Cost of Multicast Transmissions. In *Proceedings of the Thirty Second Annual ACM Symposium on Theory of Computing (STOC)*, Portland, Oregon, May 2000.

[12] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[13] M. Goemans and D. Williamson. A General Approximation Technique for Constrained Forest Problems. *SIAM J. Comput.*, 24:296–317, September 1995.

[14] A. Goldberg and J. Hartline. Competitive Auctions for Multiple Digital Goods. Technical report, Technical Report STAR-TR-00,05-02, May 2000.

[15] J. Håstad. Clique is Hard to Approximate Within $n^{1-\epsilon}$. In 37th IEEE Symp. on Foundations of Computer Science, pages 627–636, 1996.

[16] John Hershberger and Subhash Suri. Vickrey Prices and Shortest Paths: What is an Edge Worth? In *IEEE Symposium on Foundations of Computer Science*, pages 252–259, Las Vegas, NV, 2001.

[17] S. Herzog, S. Shenker, and D. Estrin. Sharing the "Cost" of Multicast Trees: An Axiomatic Analysis. *Transactions on Networking*, December 1997.

[18] The Inktomi Overlay Solution for Streaming Media Broadcasts. Inktomi White Paper, 2001.

[19] K. Jain and V. Vazirani. Applications of Approximation Algorithms to Cooperative Games. In *Proceedings of the Thirty ThirdAnnual ACM Symposium on Theory of Computing (STOC)*, 2001.

[20] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of USENIX*, San Diego, CA, October 2000.

[21] D. Johnson, M. Minkoff, and S. Phillips. The Prize Collecting Steiner Tree Problem: Theory and Practice. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, January 2000.

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

[23] S. McCanne, V. Jacobson, and M. Vetterli. Receiver Driven Layered Multicast. In *Proceedings of SIGCOMM'96*, Stanford, CA, August 1996.
[24] D. Mitzel and S. Shenker. Asymptotic Resource Consumption in Multicast Reservation Styles. In *Proceedings of ACM SIGCOMM'94*, London, UK, August 1994.
[25] H. Moulin and S. Shenker. Strategyproof Sharing of Submodular Costs: Budget Balance versus Efficiency. *Economic Theory*, To appear.
[26] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *31st ACM Symp. on Theory of Computing (STOC)*, pages 129–140, San Diego, CA, May 1999.
[27] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1982.
[28] L. Shapley. A Value for n-Person Games. In *Contributions to the Theory of Games*, pages 31–40, 1953.
[29] Sharman Networks. KaZaa file-sharing software. http://www.kazaa.com.
[30] Personal communication with Scott Shenker.
[31] I. Stoica, T. Ng, and H. Zhang. REUNITE: A Recursive Unicast Approach to Multicast. In *Proceedings of IEEE INFOCOM'00*, Tel-Aviv, Israel, March 2000.
[32] L. Vicisano, J. Crowcroft, and L. Rizzo. TCP-like Congestion Control for Layered Multicast Data Transfer. In *Proceedings of INFOCOM'98*, San Francisco, CA, March 1998.
[33] A.C. Yao. Some complexity questions related to distributive computing. In 11^{th} *ACM Symposium on Theory of Computing*, pages 209–213, 1979.
[34] A.C. Yao. Lower bounds by probabilistic arguments. In 24^{th} *IEEE Symposium on Foundations of Computer Science*, pages 420–428, 1983.