Pricing Multicasting in More Practical Network Models

Micah Adler* Dan Rubenstein†

April 27, 2001

Abstract

In [9], Feigenbaum, Papadimitriou and Shenker initiate the study of pricing algorithms for multicast transmission. In this paper, we build on this work by studying the effect on the complexity of pricing when two practical considerations 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. As a test case for the different types of pricing that can occur, we consider a pricing mechanism, called *Marginal Cost*, that has seen considerable attention in simpler network models. We demonstrate that the details of how multiple rates are provided has a significant impact on the complexity of pricing. For multiple rates provided via the *layered* paradigm, we provide a distributed algorithm for computing Marginal Cost efficiently in terms of local computation and message complexity. The bit complexity (per edge) of this algorithm depends linearly on the product of the tree height and the number of possible rates. However, we provide two lower bounds on bit complexity, demonstrating that computing Marginal Cost (a) with multiple rates requires a bit complexity that is linear in the number of rates, and (b) with a cost for enabling multicasting requires a bit complexity that is linear in the height of the tree.

A modification of our algorithm for the layered paradigm also applies to the *split session* paradigm of providing multiple rates, but in this case, both the local computation and the bit complexity become exponential in the number of possible rates. However, we also demonstrate that for the split session paradigm, the problem becomes NP-Hard even to approximate if the number of possible rates is part of the input. This indicates that we cannot expect to do much better than the algorithm we provide. Finally, we examine the effect of delivering the information for the different rates from different locations within the network. We show that in this case, the Marginal Cost problem becomes NP-Hard in the split session paradigm even for a constant number of possible rates, but that in the layered paradigm it can be solved in polynomial time by formulating the problem as a linear program that is guaranteed to have an integral optimal solution.

*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.

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 [7]. However, this sharing of link bandwidth significantly complicates the issue of pricing [8]. Merely splitting the cost of delivery among the receivers is not an adequate solution, since some receivers might be charged more than they would be willing to pay. By dropping out of the multicast session, the charge levied upon the remaining receivers would be more than if a lower, acceptable price had been offered to the exiting receiver. An alternative pricing strategy that has received attention recently [9, 14, 20, 21] is to have each receiver place a bid for the content. After considering all bids, the network determines the set of receivers that receive 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*.

In order for a pricing mechanism to be used in a network, there must be an algorithm for realizing the mechanism. For such an algorithm to be used in a distributed environment such as the Internet, it should be efficient 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 first addressed by Feigenbaum, Papadimitriou and Shenker [9]. They consider two pricing mechanisms: *Marginal Cost* and *Shapley Value*, and provide efficient algorithms for the Marginal Cost mechanism, as well as algorithms and lower bounds on the efficiency of algorithms for the Shapley Value mechanism.

In this paper, we study the effect on the complexity of realizing pricing mechanisms of incorporating two practical considerations into the network model. In particular, the model of [9], similar to that used in other related work [14, 20], makes two simplifying assumptions: (1) they assume that every recipient either receives a "full bandwidth" version of the multicast session or nothing, and (2) they assume that multicasting is possible at every node of the network at no cost. We here consider a network model that removes these assumptions. When multiple possible transmission rates and a cost for enabling a node are included in the model, the problem of realizing pricing mechanisms becomes more complicated, and leads to a number of new and interesting combinatorial optimization problems. Furthermore, we see that the details of how these features are incorporated into the model has a significant effect on the complexity of realizing pricing mechanisms.

To remove assumption (1), we consider a network model that permits transmission of session data at a set of ℓ pre-determined rates $\rho_1 \leq \rho_2 \leq \cdots \leq \rho_\ell$. Such an approach is crucial if the multicast session is going to be received by users with significantly different bandwidth connections. The two main techniques for providing multiple rates in practice are having a separate multicast *group* for each possible rate [5], and having a 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 receive at rate ρ_j, a receiver is sent layers $1 \ldots j$ [19, 25, 3, 2]. We refer to the first technique as the *split session* paradigm, and the second as the *layered* paradigm. This paper is the first to consider algorithms for realizing pricing mechanisms for multicasting when there are multiple possible rates of transmission. In fact, to the best of our knowledge, this is the first work to even define pricing mechanisms for multicast sessions with multiple rates. Auction mechanisms for multiple goods where there is *no* cost for each additional copy of a good delivered have been studied in [12]. We assume that 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 what rate is obtained by each receiver.

To remove assumption (2), we consider a network model where there is a cost for making a node of the network capable of multicasting. If a node is enabled, it can forward any number of copies of a session it is sent to other enabled nodes or receivers, possibly using nodes that are not enabled as intermediaries. On the other hand, if a node is not enabled, it can only forward the same number of copies of a sessions as it is sent, and thus the cost of enabling a node may be offset by the reduced cost of only delivering one copy of the session to that node. The cost for enabling multicasting may vary from node to node. This model incorporates recent approaches to multicasting such as network overlays [15] and application layer multicasting [16, 24, 4, 6]. This paper is the first to consider pricing when there is a cost for enabling multicasting. In this model, the network must still determine the set of participants in the multicast session, as defined by the pricing mechanism, but which set the network determines is now influenced by the cost of enabling multicasting at each node. In addition, the network must choose which nodes of the network to enable.

As in [9], many of our results assume that there is a single directed multicast tree that defines the routes used by all transmissions; this tree is not affected by which receivers are accepted, or which nodes are multicast enabled. [9] 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 pricing mechanisms (the problem becomes a version of the Prize Collecting Steiner Tree problem - see also [17, 11]). While this hardness result 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 which we consider.

1.1 Summary of results

In this paper, we focus on the Marginal Cost mechanism. 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. Furthermore, some of our techniques also apply to the Shapley mechanism, but this is not addressed in this paper, since a lower bound from [9] demonstrates that realizing this mechanism is computationally expensive even in the simpler network model considered there. We start by introducing a straightforward generalization of the Marginal Cost mechanism to scenarios where there are multiple possible rates, and prove that this generalization has the same properties as the mechanism for single rate scenarios. We define this mechanism in Section 2, but we here note that any algorithm for this mechanism also applies to the simple mechanism that chooses the network configuration that maximizes the network profit, and 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*.

We then consider algorithms for realizing Marginal Cost in the layered paradigm. We provide a distributed algorithm that realizes the Marginal Cost mechanism in such a network, even when each node has a cost for enabling multicast. This algorithm 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 required to represent the bids and costs of the network.

The algorithm in [9] for the same problem (in the simpler model) achieves the same result using

only $O(K)$ bits of communication per edge. However, we also provide two lower bounds on the communication required to maximize network welfare: We demonstrate that the linear dependence on h is in fact necessary when considering networks with a cost for enabling multicast, and that the linear dependence on ℓ is necessary when considering multiple layers. In particular, we show that there exist networks such that for any protocol, there is an input such that the average, over all edges of the network, of the number of bits communicated is $\Omega(hK)$ (for reasonable assumptions on the value of K). This lower bound is information theoretic, and holds even when there is only a single possible rate of transmission. For the case of layered transmissions, we demonstrate that for **every** network without receivers directly adjacent to the source, there is an input such that the average number of bits communicated per edge is $\Omega(\ell K)$ (again, for reasonable assumptions on the value of K). This lower bound is also information theoretic, and still holds when there is no cost for enabling multicasting at any node.

We then turn to the question of realizing the Marginal Cost mechanism for the split session paradigm. We demonstrate that our algorithm for the layered case can easily be adapted to provide a solution for this case, even when there is a cost for enabling multicasting. However, both the communication and the computation required of the adapted algorithm become 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 grows with the input size. This result holds even in networks without a cost for enabling multicasting costs, and even if the allowed rates are (for example) powers of 2, and the cost for a group to use an edge is proportional to the bandwidth of the group. 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.

Finally, we turn to the question of removing the assumption that there is a single fixed multicast tree. In particular, 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 this 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. To demonstrate this, we show that this problem can be expressed as an integer program with a linear programming relaxation that is guaranteed to have an integral optimal solution.

The remainder of the paper is organized as follows. In Section 2, we provide more details on the network model, and describe the Marginal Cost pricing mechanisms we consider. In Section 3, we describe our Marginal Cost algorithm for the layered paradigm with node cost, and show how to adapt this algorithm to the split session paradigm. In Section 4, we provide our lower bounds on communication requirements for Marginal Cost computation, as well as the hardness results for the split session paradigm where the number of rates grows with the input. In Section 5, we describe our hardness results for the case where the multicast trees vary for different groups, as well as our Linear Programming algorithm for the case of layered multicasting.

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. We say that a node $n \in N$ is *enabled* when it is configured to multicast for the session: i.e., any flow of information entering that node can be forwarded on multiple outgoing edges. If a node is not enabled, then each incoming flow can only be forwarded on a single outgoing edge. There is a cost $\mathbf{c}_n$ for enabling node n. We can also model nodes that cannot be enabled for multicasting by setting their cost to ∞. We here assume 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. There is also a cost 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 $\mathbf{c}_e$) is the cost at e of providing the jth rate. In the layered paradigm, $[\mathbf{c}_e]_j$ is the cost of providing the jth layer.

For most of this paper, we make the assumption that the multicast session uses a single source, and 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 [7], 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 multicast networks constructed using overlays or application layer multicasting. However, the hardness result from [9] applies to a network where the routing depends on which nodes are enabled, even if the cost of enabling those nodes is zero. Thus, we cannot expect to find efficient algorithms for networks with costs for enabling nodes without some restriction on the choice of routes. We consider the model of this paper an important step towards understanding algorithms for pricing in such networks.

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. For the optimization problems we consider, the input is distributed as follows: each node n is informed of $\mathbf{c}_n$, $\mathbf{c}_e$ 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, as well as 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 amount it must pay to receive this service. For example, as long as the

network charges the receiver less than its utility, the receiver's satisfaction is positive, and it is in the receiver's interest to join the session. The simple 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 satisfaction at the cost of network profits. What is needed is a *strategy-proof* mechanism in which a receiver maximizes its satisfaction by bidding its true utility.[1] 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 [21] 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: the amount payed 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 [23], although that mechanism does not satisfy Efficiency.

All of the previous work on Marginal Cost assumes that the service being priced is binary: each receiver either receives the service at a given rate or does not. However, the definition can be easily extended to multiple rates. In particular, consider any network G and set of receivers, R that wish to join a session under any generic paradigm, $pdgm$, in which 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 welfare. The price charged to a receiver i that has an accepted bid of b_i^k is defined to be $\mathcal{M}_{pdgm}(i) = b_i^k - (P^*_{pdgm}(R) - P^*_{pdgm}(R \backslash \{i\}))$, where $P^*_{pdgm}(X)$ 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 marginal contribution to the network welfare of receiver i participating in the multicast session. The fact that Marginal Cost for a binary service is strategy-proof does not imply that Marginal Cost for multiple goods is also strategy-proof. Thus, we next prove the following theorem.

Theorem 1 *The Marginal Cost mechanism for multiple goods is strategy-proof, efficient, NPT, VP, and CS.*

Proof: The proofs that these properties are satisfied under this definition of marginal cost are trivial for all cases except for strategy-proofness. For strategy-proofness, we must show that a receiver maximizes its satisfaction by bidding its true utility for the service associated with each vector component. Our proof considers two sets of bids that a receiver can place. $B_1 = \langle b_1^1, b_1^2, \cdots, b_1^n \rangle$ is a bid in which a receiver bids the utility that it would receive if the transmission associated with the bid were accepted. $B_2 = \langle b_2^1, b_2^2, \cdots, b_2^n \rangle$ is a bid in which the receiver need not bid its utility. We will show that the receiver's satisfaction is no less for B_1 than for any possible B_2.

A *configuration* of the network, C, fixes the set of admitted receivers and the service received by each receiver. For instance, within the layered paradigm, a configuration would fix the set of layers

[1] We assume that receivers do not conspire with one another.

or the group from which each receiver receives data, as well as the set of nodes that would be enabled for multicasting. Note that under a fixed configuration, C, a receiver's accepted bid is the one that corresponds to the service it is offered within C. No bid is accepted if the receiver does not receive service.

Let $C_i, i = 1, 2$ be the configuration that the network would use to optimize profits (aggregating accepted bids minus link and node costs) when r places bid B_i. Let C_R be the configuration that the network uses to achieve maximum network welfare over all configurations in which none of r's bids are accepted. Let $P_C(B_i)$ be the network welfare that results from bid B_i under configuration C. Let $A(C) = j$ if r's jth component of the bid is used under configuration C (i.e., for bid B_i, b_i^j is accepted.) We let $A(C) = 0$ when r is rejected. Let $V_C(B)$ be the value of the $A(C)$th component of B, i.e., $V_C(B_i) = b_i^{A(C)}$.

We point out several facts. Let us fix a configuration C.

Fact 1 *Let $A(C) = k$ and let $B_x = \langle b_x^1, \cdots, b_x^n \rangle$ and $B_y = \left\langle b_y^1, \cdots, b_y^n \right\rangle$ be any two bids placed by r, with $b_y^j = b_x^j + \mu_j$ where μ_j is an arbitrary real-valued value for all $1 \leq j \leq n$. Then $P_C(B_y) = P_C(B_x) + \mu_k$.*

Fact 2 *If $C = C_R$ then $P_{C_R}(B_1) = P_{C_R}(B_2)$.*

Fact 1 states that within a fixed configuration, C, modifying the accepted bid by a constant value μ alters the network welfare by μ, and modifying an unaccepted bid does not alter the network welfare. We stress that this fact requires that the configuration, C, remain fixed. Fact 2 states that if a configuration does not accept any of r's bids, then the network welfare is not altered under the configuration if r changes its bid.

Lemma 1 *If r is admitted to the session under bids B_1 and B_2, in which r bids its utility for B_1, then $P_{C_1}(B_1) \geq P_{C_2}(B_2) + V_{C_2}(B_1) - V_{C_2}(B_2)$.*

Proof: We consider two cases separately. First, we consider the case where $A(C_1) = A(C_2) = i$. Next, we consider the case where $A(C_1) = i$ and $A(C_2) = j$ with $i \neq j$. W.l.o.g., we can assume that $i = 1$ and $j = 2$.

For the first case, let $b_2^1 = b_1^1 + \mu$, where μ is any real valued number. Because C_1 is the optimal profit configuration for B_1, we have that $P_{C_1}(B_1) \geq P_{C_2}(B_1)$. By Fact 1, $P_{C_2}(B_1) = P_{C_2}(B_2) - \mu = P_{C_2}(B_2) + V_{C_2}(B_1) - V_{C_2}(B_2)$. Hence, $P_{C_1}(B_1) \geq P_{C_2}(B_2) + V_{C_2}(B_1) - V_{C_2}(B_2)$ for this first case.

For the second case, let $b_2^2 = b_1^2 + \eta$. Because C_1 is the optimal profit configuration for B_1, we again have $P_{C_1}(B_1) \geq P_{C_2}(B_1)$. By Fact 1, we have that $P_{C_2}(B_1) = P_{C_2}(B_2) - \eta = P_{C_2}(B_2) + V_{C_2}(B_1) - V_{C_2}(B_2)$. Hence, $P_{C_1}(B_1) \geq P_{C_2}(B_2) + V_{C_2}(B_1) - V_{C_2}(B_2)$ for the second case as well. ■

We are now ready to prove Theorem 1. We again consider two bids, B_1 and B_2, where the receiver bids its utilities in B_1 and bids arbitrarily in B_2. There are four cases to consider. A receiver's bid is accepted in both bids B_1 and B_2, only B_1 has an accepted bid and all of B_2's bids are rejected, all of B_1's bids are rejected and one of B_2's bids is accepted, and all bids are rejected in both B_1 and B_2. We consider each case one at a time.

First, let us consider the case where both bids are rejected. In this case, the receiver's satisfaction is 0 for both bids is 0: it receives no utility, and no price is charged. Hence, the theorem holds for this case.

Next, let us consider the case where C_1 is a configuration that admits r and C_2 is a configuration that rejects r. W.l.o.g., assume $A(C_1) = 1$. Since B_1 is a "truthful" bid, r's utility equals b_1^1 as well. Hence, its satisfaction is $b_1^1 - \mathcal{M}_{pdgm}(r) = b_1^1 - (b_1^1 - P_{C_1}(B_1) + P_{C_R}(B_1))$. Since $P_{C_1}(B_1)$ maximizes profits for bid B_1 over all possible configurations, it must be the case that $P_{C_1}(B_1) \geq P_{C_R}(B_1)$. Hence, the satisfaction is no less than 0, which is its satisfaction in configuration C_2 (brought on by bid B_2). Thus, the theorem holds for this case.

Next, consider the case where one of B_2's bids is accepted. W.l.o.g., assume $A(C_2) = 1$, $b_2^1 = b_1^1 + \mu$, and that $A(C_1) = 0$ (B_1's bids yield a configuration in which r is not admitted). r's satisfaction achieved by bidding B_1 is 0. Since B_1 is the "truthful" bid, it holds the actual receiver utilities. By bidding B_2, r's satisfaction is $b_1^1 - (b_2^1 - P_{C_2}(B_2) + P_{C_R}(B_2)) = b_1^1 - (b_1^1 + \mu) + P_{C_2}(B_2) - P_{C_R}(B_2)$. Application of Fact 1 to $P_{C_2}(B_2)$, algebraic simplification, and application of Fact 2 ($P_{C_R}(B_2) = P_{C_R}(B_1)$) simplifies this expression to $P_{C_2}(B_1) - P_{C_R}(B_1)$. Since none of r's bids were accepted when r bid B_1 and C_R is defined as the maximum network welfare over such configurations, we have that $P_{C_2}(B_1) \leq P_{C_R}(B_1)$. Thus, we have that $P_{C_2}(B_1) - P_{C_R}(B_2) \leq 0$ and the theorem holds for this case.

Last, assume that both bids are accepted. As before, since B_1 is the "truthful" bid, it holds the actual receiver utilities. This means that the utility gained from the acceptance of bid B_1 is $V_{C_1}(B_1)$, and the utility from the acceptance of bid B_2 is $V_{C_2}(B_1)$. It follows that the respective satisfactions from bids B_1 and B_2 are $sat(B_1) = V_{C_1}(B_1) - (V_{C_1}(B_1) - P_{C_1}(B_1) + P_{C_R}(B_1))$ and $sat(B_2) = V_{C_2}(B_1) - (V_{C_2}(B_2) - P_{C_2}(B_2) + P_{C_R}(B_2))$. Noting that $P_{C_R}(B_1) = P_{C_R}(B_2)$, we have $sat(B_1) - sat(B_2) = P_{C_1}(B_1) - P_{C_2}(B_2) + V_{C_2}(B_2) - V_{C_2}(B_1)$. By applying Lemma 1, we see that $sat(B_1) - sat(B_2) \geq 0$, proving the Theorem for this final case. ■

3 Efficient Distributed Algorithms for Marginal Cost

In this section, we present an efficient distributed algorithm for realizing the Marginal Cost mechanism in the layered paradigm. We first provide an algorithm that maximizes network welfare. We then demonstrate that it can be modified 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.

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 $0 \leq j \leq \ell$, $1 \leq k \leq h_n$. $S_{j,k}(n)$ 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, under the following two assumptions:

- $\pi_{r,n}$ is not enabled for all $1 \leq r < k$. Nodes n and $\pi_{n,k}$ may or may not be enabled, but only the cost of enabling n counts against $S_{j,k}(n)$.
- 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 $\pi_{n,k}$ to n.

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 $\mathbf{c}_{(\pi_{n_i,k+1},n_i)} = \mathbf{c}_{(\pi_{n,k},n_i)} + \mathbf{c}_{n,n_i}$, where the addition is a componentwise 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

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

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

Theorem 2 *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 $O(\ell h_n |D(n)|)$, exactly two messages are communicated between node n and its child $n_i \in D(n)$, and the total number of bits required by these messages is $O(\ell h_{n_i} K)$, where K is the maximum number of bits required to represent any value $S_{j,k}(n_i)$ or $\mathbf{c}_{(\pi_{n,k},n)}$.*

number of bits communicated between any node n and its child

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 (simple path) distance from n to a leaf. 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.

For the inductive step of the double induction, consider any k, any layer $j > 0$, and any non-leaf node n. By induction, 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 also that $S_{j-1,k}(n)$ is computed correctly. There are now four cases: 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, $S_{j,k}(n) = S_{j-1,k}(n)$. If the optimal solution transmits all j layers to node n, and n is enabled, then $S_{j,k}(n) = \sum_{n_i \in D(n)} S_{j,1}(n_i) - \mathbf{c}_n - \sum_{j=1}^{r} [\mathbf{c}_{(\pi_{n,k},n)}]_r$. 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 must also 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 we see that $S_{j,k}(n) = \sum_{n_i \in D(n)} S_{j,k+1}(n_i)$. Since the algorithm takes the maximum of these possibilities, it does in fact compute the correct value of $S_{j,k}(n)$. ■

We next point out 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 $S_{\int,k}(n)$, where $\int$ is any subset of the ℓ groups, analogously to $S_{j,k}(n)$, except that the second condition becomes

- If a group s is transmitted from $\pi_{k,n}$ to n, then $s \in \int$.

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

$$S_{\int,k}(n) = \max \left\{ \sum_{n_i \in D(n)} S_{\int,k+1}(n_i), \sum_{n_i \in D(n)} S_{\int,1}(n_i) - \mathbf{c}_n - \sum_{s \in \int} [\mathbf{c}_{(\pi_{n,k},n)}]_s, \text{MAX}_{\int' \in m(\int)} S_{\int',k}(n) \right\},$$

where $m(\int)$ is the set of all subsets of $\int$ containing exactly one less element. We call the resulting algorithm **Max-Split-Welfare**. The proof of the following theorem follows along the lines of the proof of Theorem 2.

Theorem 3 *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 $O(\ell 2^{\ell} h_n |D(n)|)$, exactly two messages are communicated between node n and its child $n_i \in D(n)$, and the total number of bits required by these messages is $O(2^{\ell} h_{n_i} K)$, where K is the maximum number of bits required to represent any value $S_{\int,k}(n_i)$ or $\mathbf{c}_{(\pi_{n,k},n)}$.*

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.

The network welfare computed during the upward phase is simply be 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 (1) 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 is enabled, which layers or groups it receives from this enabled parent, and what 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 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 the upward phase. In addition, in the downward phase, each node will compute the 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 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. Each node learns every value of $B_{j,k}(n)$ from its parent during the downward phase. Our algorithm is 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

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

- 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 $2 \leq k \leq h_{n_i}$, node n uses

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

For the case where $k = 1$, node n uses

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

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

Theorem 4 *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 done in polynomial time, each node n sends only one message to each child $n_i \in D(n)$, and the number of bits required of this message is $O(\ell h_{n_i} K)$, where K is the maximum number of bits required to represent any value $B_{j,k}(n_i)$, or the maximum network welfare.*

Proof: Proving the bounds on the amount of computation and communication 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 the 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 obtain without any profit provided by the subtree rooted at n is the maximum profit obtainable in the subtrees rooted at the other children of the source. From this, we subtract the cost of sending the required layers from s 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. The case where $k > 1$ computes $B_{j,k}(n_i)$ when node n is not enabled. When this is the case, there is some $t \geq j$, such that t layers reach n from $\pi_{n,k-1}$. Now, consider all configurations where t layers reach n and $\pi_{n,k-1}$ is the first enabled ancestor of n. Within this set of configurations, the configuration of the subtree rooted at any $n'_i \in D(n)$ has no effect on the optimal configuration of the remainder of the tree. Thus, to convert $B_{j,k-1}(n)$ to the optimal configuration where t layers reach n, $\pi_{n,k-1}$ is the first enabled ancestor of n, and no receiver in the subtree rooted at n_i contributes to the network welfare, we simply add to $B_{j,k-1}$ the maximum welfare possible in the subtree rooted at each $n'_i \in D(n)$, $n'_i \neq n_i$, when the cost of sending any required layers (up to t) from $\pi_{n,k-1}$ is included. This is exactly the value $S_{t,k}(n'_i)$, computed during the up-phase of the algorithm **Max-Layered-Welfare.** Thus, to find the value of $B_{j,k}(n_i)$, we simply have to take the optimal allowed value of t, and then subtract the cost of sending the first j layers from n to n_i.

The case where $k = 1$ computes $B_{j,k}(n_i)$ when node n is enabled. Again, there is some t such that t layers reach node n. Within the set of configurations where n is enabled and exactly t layers reach n, the configuration of any subtree rooted at a child of n has no effect on the optimal configuration of the remainder of the tree. Thus, to find the optimal network welfare where n is enabled and receives t layers, and no receiver in the subtree rooted at n_i contributes to the welfare, we first find the optimal configuration where n is enabled and receives t layers, but none of the receivers in the subtree rooted at n contribute to the welfare. In the optimal such configuration, there is some first ancestor of n, $\pi_{n,u}$, that is enabled (again, the source is always enabled), and thus it is sufficient

to maximize $B_{t,u}(n)$ for all possible u, and for each subtract the cost of enabling n. To this value, we add the optimal welfare for the subtrees rooted at each $n'_i \in D(n)$, $n'_i \neq n_i$, including the cost of sending any required layers from n. The value added for each n'_i is exactly $S_{t,1}(n'_i)$. Again, once we have found the optimal value of t, we subtract the cost of sending the first j layers from n 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

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 here assume 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 5 *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 return 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 $b_1 \dots b_i$ are fixed, but $a_1 \dots a_i$ are not fixed, the only edge and node costs that are not fixed for inputs $f_i(a_1, \dots, a_i, b_1, \dots, b_i)$ lie between edge e_{i+1} and the source. If $a_1 \dots a_i$ are fixed but $b_1 \dots b_i$ are not fixed, the only edge and node costs that are not fixed for inputs $f_i(a_1, \dots, a_i, b_1, \dots, b_i)$ 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, $a_1, \dots a_i$ are each chosen independently and uniformly at random from the integers in $[0 \dots \frac{2^k}{4i^2} - 1]$, and D returns the input $f_i(a_1, \dots, a_i, a_1, \dots, a_i)$. With probability $\frac{1}{2} + \frac{\gamma}{2(1-\gamma)}$, $a_1, \dots a_i, b_1 \dots b_i$ are each chosen independently and uniformly at random, and D returns the input $f_i(a_1, \dots, a_i, b_1, \dots, b_i)$. 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 2 *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 $I = f_i(a_1, \dots, a_i, a_1, \dots, a_i)$ and $I' = f_i(a'_1, \dots, a'_i, a'_1, \dots, a'_i)$ 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 $I'' = f_i(a_1, \dots, a_i, a'_1, \dots, a'_i)$ 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 [26], 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 [27], 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. ■

■

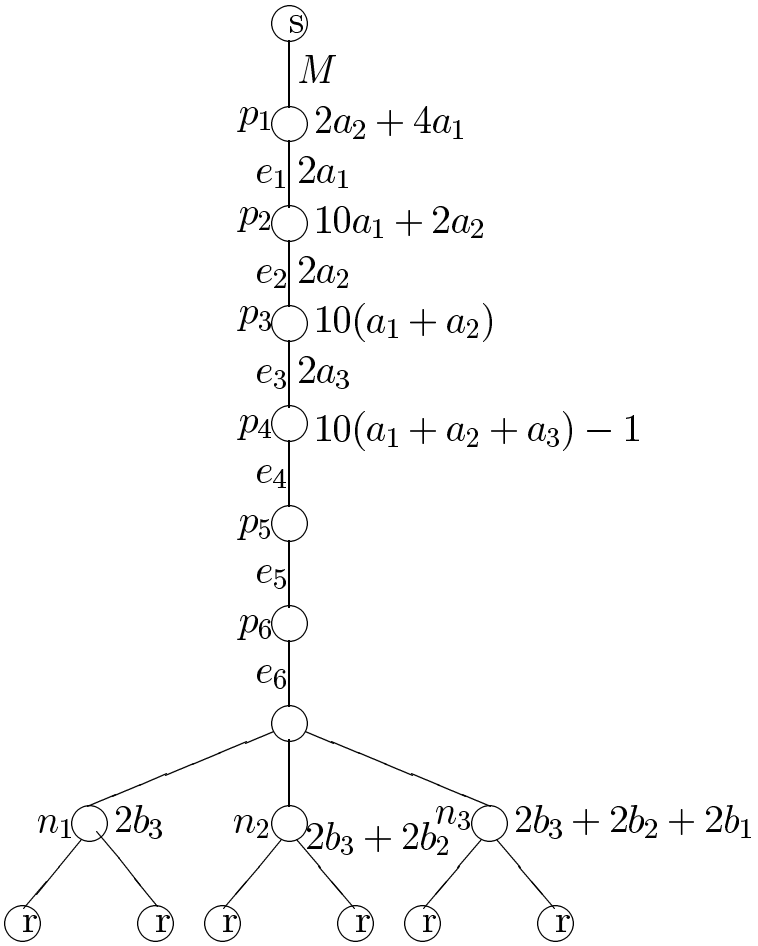

Figure 1: The network N_3. Node (edge) names are depicted to the left of the node (edge, resp.). The cost of enabling multicast at a node is depicted to the right of the node, and the cost of using an edge is depicted to the right of the edge. When the cost of enabling multicast at a node is not depicted,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

4.2 Communication lower bound for the layered paradigm

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 6 *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 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 cell 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. 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 3 *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 profit 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 [26] (see also [18]) 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 [26], 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 profit 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 profit 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 [27], 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 exponential dependence of the communication and computation on ℓ in the algorithm **Max-Split-Welfare**. While the communication requirements could of course be made polynomial by simply sending all information to the source, we here provide evidence that we cannot expect to achieve a more efficient running time for any algorithm that maximizes network welfare in the split session paradigm.

Theorem 7 *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 could be converted into a polynomial time $|V|^{1-\epsilon}$-approximation algorithm for the largest independent set in a graph. Since [13] showed 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. Our reduction is possible 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. We also specify costs in such a manner that for any edge e, there is a constant $c(e)$, such that the cost for any group g to use e is $B(g)c(e)$, where $B(g)$ is the rate of g. Also, the reduction works for a broad range of possible sets of session rates, including (for example) powers of two, although our reduction requires $|V|$ groups, and thus this example would lead to exponential rates.

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: 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 $r(v) \in R$. The cost of a group using an 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 demonstrate that maximizing the network welfare in this network and 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 $r(v)$, plus $deg(v)(B(g(v)) + 2)$, from all $r(e)$. The total cost for group $g(v)$ is $B(g(v))(deg(v) + 2)$, 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 that have a profit of 1 can correspond to vertices that share an edge, and thus

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 to originate from the same source, and as a result can have different trees. 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, most 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.

5.1 Hardness of the multiple tree case.

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 has computation and communication complexity that is exponential in the number of session, but is efficient for the case of a constant number of session.

Theorem 8 *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 still NP-Complete [10]. 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 those literals can appear in a total of at most 5 clauses, at most 10 colors are ruled out by previous assignments to the other two literals in those 5 clauses. Since there are 12 colors to choose from, there is always at least two colors available to color both literals.

We then construct the welfare maximization problem. We here 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 here assume that the cost to use an 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.

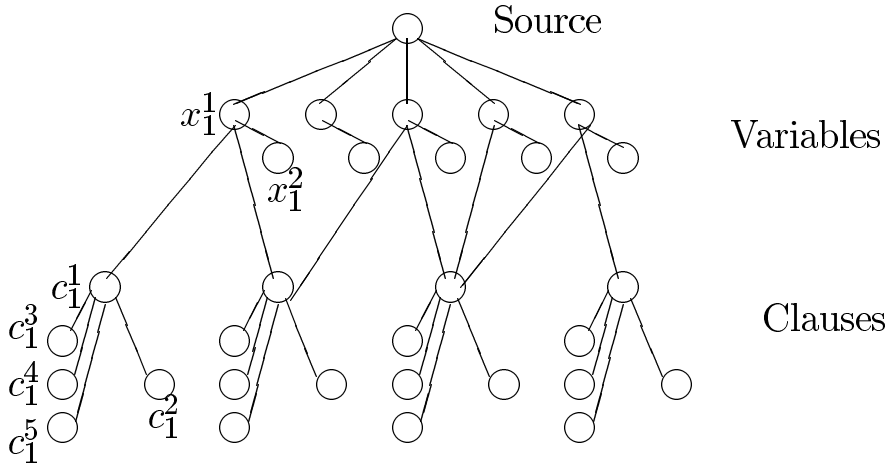

Figure 2: The network used for the reduction.

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 one unit for any group that is successfully delivered to that receiver. The receivers at any c_j^2 each bid for any group. The receivers at any x_i^2 each bid six for any group. There are a total of 12 groups, one for each of the colors. We next describe the trees for each group. For group t, the tree is rooted at the source, and routed from there to every x_i^1 such that 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, the group continues to x_i^2. From x_i^1, that group 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 every pair of literals appearing in the same clause are assigned to a different group, the routing for the group j 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.

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. ■

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 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. ■

■

5.2 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.[2] We next provide a polynomial time algorithm for this problem. The technique we use consists of providing an integer program for the welfare maximization problem such that the optimal solution to the linear programming relaxation of this problem can be converted to an integral solution of equal value. This leads to a centralized polynomial time algorithm that finds the optimal solution. This would 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 [1], but in our scenario, the objective function is distributed across the users, and thus the techniques from [1] do not apply here. We also point out that the result we demonstrate below can also be obtained by showing that the same class of integer programs we describe are always totally unimodular (see for example [22]). However, the more direct proof we

[2]Unless of course P = NP.

provide below provides additional insight into why the result holds. Finally, we 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 9 *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 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 comparison 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

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

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. ■

References

[1] 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.

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

[3] 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.

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

[5] 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.

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

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

[8] 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.

[9] 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.

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

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

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

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

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

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

[16] 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.

[17] 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.

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

[19] S. McCanne, V. Jacobson, and M. Vetterli. Receiver Driven Layered Multicast. In *Proceedings of SIGCOMM'96*, Stanford, CA, August 1996.

[20] D. Mitzel and S. Shenker. Asymptotic Resource Consumption in Multicast Reservation Styles. In *Proceedings of ACM SIGCOMM'94*, London, UK, August 1994.

[21] H. Moulin and S. Shenker. Strategyproof Sharing of Submodular Costs: Budget Balance versus Efficiency. *Economic Theory*, To appear.

[22] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Dover Publications, 1982.

[23] L. Shapley. A Value for n-Person Games. In *Contributions to the Theory of Games*, pages 31–40, 1953.

[24] 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.

[25] 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.

[26] A.C. Yao. Some complexity questions related to distributive computing. In 11^{th} *ACM Symposium on Theory of Computing*, pages 209–213, 1979.

[27] A.C. Yao. Lower bounds by probabilistic arguments. In 24^{th} *IEEE Symposium on Foundations of Computer Science*, pages 420–428, 1983.