Mathematical Methods of Operations Research manuscript No.
(will be inserted by the editor)

Multirate Multicast Service Provisioning I: An Algorithm for Optimal Price Splitting Along Multicast Trees

Tudor Mihai Stoenescu[1], **Mingyan Liu**[2], **Demosthenis Teneketzis**[3]

[1] California Institute of Technology, M/C 136-93, Pasadena, California 91125 e-mail: tudor@caltech.edu
[2] University of Michigan, 1301 Beal Ave. Ann Arbor Mi. 48109-2122, USA e-mail: mingyan@eecs.umich.edu
[3] University of Michigan, 1301 Beal Ave. Ann Arbor Mi. 48109-2122, USA e-mail: teneket@eecs.umich.edu

Key words Multirate Multicast, Rate Allocation, Pricing Mechanism, Price Splitting

The date of receipt and acceptance will be inserted by the editor

Abstract In this two-part paper we present a general framework for addressing the optimal rare control problem in multirate multicast where the objective is the maximization of a social welfare function expressed by the sum of the users' utility functions. Specifically, we propose a market-based mechanism that satisfies the informational constraints imposed by the decentralization of information in multirate multicast service provisioning, and achieves an optimal solution to the corresponding centralized optimization problem. In Part I we discover properties of an optimal solution to the centralized problem. Based on these properties, we develop a distributed algorithm that determines how link prices are split among users whose connections along a multicast tree share the same link.

1 Introduction

Multicasting provides an efficient method of transmitting data in real time applications from one source to many users. The source sends one copy of a message to its users and this copy is replicated only at the branching points of a multicast tree. Real time examples of such multicast applications are audio/video broadcasting, teleconferencing, distributed databases, financial information, electronic newspapers, weather maps and experimental data.

Conventional multicast studies the problem in which the rate received by all the users of the same multicast group is constant. The inherent problem with such a formulation is that a constant rate will overwhelm the slow receivers while starving the fast ones. Multi-rate transmissions can be used to address this problem by allowing a receiver to obtain data at a rate that satisfies its requirements. One way of achieving this is through hierarchical encoding of the transmission, in which a signal is encoded into multiple layers that can be incrementally combined to improve quality. These hierarchical encoding type of transmission schemes have been investigated both for audio and video transmissions over the Internet [4], [41] and over ATM networks [20]. Internet protocols for adding and dropping layers for hierarchical encoding type of transmissions are presented in [23] and [27].

Within the context of single rate and multi-rate multicast problems, studies have addressed issues of bandwidth/rate allocation [5, 10, 13, 15, 29–32, 34, 35, 37, 42], routing [6, 7, 28, 36, 43] and reliability [8, 11, 16]. Most of the literature on rate allocation is done via the notion of fairness [5, 10, 29–32, 34, 35, 37, 42], specifically, max-min fairness [3] and proportional fairness [18]. In particular, [35] develops a unified framework for diverse fairness objectives via the notion of fair allocation of utilities. A more general approach to rate allocation is via utility maximization. Utility maximizing is more general because rate allocation with the fairness property is utility maximizing when the utility has a special form [5, 26, 35, 37]. Although utility maximization has been extensively studied within the context of unicast rate allocation to achieve congestion control [1, 14, 17, 19, 21, 22, 24, 39, 40], relatively fewer studies

Correspondence to: tudor@caltech.edu

approached the multi-rate multicast allocation problem via a general utility maximization formulation, with the notable exceptions [5, 13, 15]. Specifically, [13] introduces a solution based on dual methods and [15] derives a primal algorithm based on non-differential optimization methods, both assuming that the users' utility functions are twice continuously differentiable and their second derivative bounded. Algorithms introduced in [5, 13, 15] have a "flat architecture" in that the optimal rate allocation messages are exchanged among nodes on the multicast tree.

In this two-part paper we present a market-based approach/mechanism to multi-rate multicast service provisioning. This approach relies upon: (i) the assumption that users are unaware of the method of service delivery (e.g., they do not know whether service provisioning is unicast or multicast and thus it is reasonable to assume that they are price-takers); and (ii) the properties of the optimal solution to the corresponding centralized resource allocation problem. The proposed market-based mechanism satisfies the informational constraints imposed by the nature of the multicast problem, results in a hierarchical architecture for resource allocation (as opposed to the flat architecture used in [5, 13, 15]), and employs a Tâtonnement process that is different from those of [5, 13, 15].

The market mechanism consists of two types of agents: (1) the network (auctioneer and service provider); and (2) the users. The network (auctioneer) announces prices per unit of rate at each link of the network. The service provider determines the service price per unit of rate for each user and announces these prices to the users. The users respond with a demand. Based on the excess demand, the auctioneer revises the link prices per unit of rate and the process repeats. This market mechanism looks at first glance like market mechanisms for unicast service provisioning (e.g., [39, 40]). The additional difficulty that arises in multirate multicast service provisioning (as compared to unicast) is in the determination of service prices given a set of fixed link prices. Since only one copy of the information is sent along links used by more than one user, the key issue in determining service prices, is to specify how a link's price is split among the users that share the link. The mechanism proposed in Part I of this two-part paper resolves this issue.

The Tâtonnement process we present in this two-part paper can be viewed as a hierarchical process consisting of two layers: a lower layer and an upper layer. In the lower layer of the process we solve the service provider's problem. This is, we present an algorithm which for a fixed set of link prices, determines the price per unit of service for each user. The algorithm also specifies the price-sharing of links that are used by more than one user. In the upper layer of the hierarchy we present a link price adjustment process (for the auctioneer) that is based on excess demand at each link. The results of the upper and lower layers combined together lead to an allocation which is an optimal solution of the centralized multirate multicast problem.

The major contribution of Part I of this paper is the solution to the service provider's problem, mentioned above, and the determination of optimal link price-sharing among the users of a link. The methodology for determining optimal link price-sharing is different form the work cited earlier; it is also different from that of [9, 12] which is based on cooperative game theory.

Under the assumption that the users are unaware of the method of delivery of services, the formulation of the multirate multicast problem presented in this paper and in [5, 13, 15] is a reasonable one. If the users are aware that service delivery is done by multicast, then the price-taking assumption is hard to justify and a more appropriate formulation of the multirate multicast problem may be one where the links shared by more than one user are treated as public goods [25, Chapter 11]. Such a formulation could prevent the users from hiding their true characteristics (e.g. their true utility functions) so as to benefit, and could avoid the free-rider problem [25, Chapter 11] which in the context of the multirate multicast problem manifests itself as follows: the price of a link that is used by more than one user is shared only by those users that demand the maximal rate.

The remainder of this paper is organized as follows. In Section 2 we formally present the centralized multi-rate multicast problem. In Section 3 we develop properties of the optimal solution of this problem. In particular these properties determine the optimal price sharing along each multicast tree. Using the properties developed in Section 3, we present in Section 4 an iterative algorithm which computes the optimal price per unit or service given a fixed price per unit of rate on each link. We conclude the paper in Section 5.

2 The Multicast Problem

In this section we present the mathematical formulation of a network multirate multicast problem.

2.1 The model, terminology and notation

Consider a network consisting of a set of L unidirectional links, each link $l \in L$ having finite capacity c_l. The network is used by a set M of multicast groups. Each multicast group $m \in M$ is specified by $\{s_m, R_m, L_m\}$, where s_m is the unique source node, R_m is the set of receiver nodes, and L_m is the set of links used by the group. Since each multicast group is a *tree*, we are going to use the terms multicast group and multicast tree interchangeably.

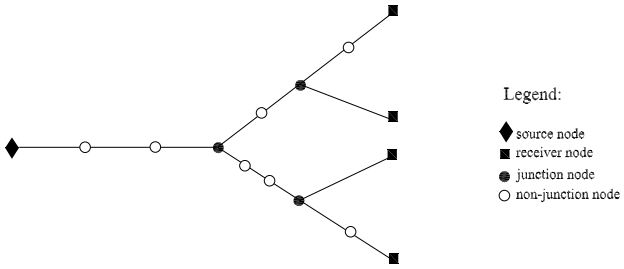

Fig. 1 A picture of a multicast tree.

We now present some terminology used for the multicast groups that is similar to terminology developed in [13, 15]. We start by looking at the nodes that are part of an arbitrary multicast group m. There are four types of nodes in this group: the source node s_m, receiver nodes $r \in R_m$, *junction nodes* and *non-junction nodes*. The junction nodes are the nodes that are connected to more than two links of L_m, i.e. they are connected to a link which will lead to the source and to two or more other links which will lead to some subset of R_m. We denote the set of all the junction nodes of multicast group m by $\hat{R}_m$, and we let $\tilde{R}_m \triangleq \hat{R}_m \cup R_m$. The non-junction nodes are all the nodes, excluding the source node, that are connected to exactly two links of L_m.

From this moment on we are going to assume that for every receiver node in m there is a unique link $l \in L_m$ connected to it, i.e. the receiver nodes are terminal nodes with an unique incoming link. For our formulation there is no loss in generality by making this assumption, since if $r \in R_m$ is a receiver node but not a terminal node, we can always replace the receiver node by a new terminal node r', which is connected to r by an infinite capacity link.

We denote by $R \triangleq \cup_{m \in M} R_m$ the set of all receiver nodes over all the multicast groups, and by $R_{l,m}$ the set of all the receivers of multicast group $m \in M$ using link $l \in L$.

We define a *branch* to be the set of links that are between a source/junction node and its immediate downstream junction/receiver node. Note that the set of branches of $m \in M$ forms a partition of L_m. Also note that each branch j can be associated with its "downstream" junction/receiver node, which will be denoted by $\mathfrak{r}(j)$. Denote the set of branches associated with receiver nodes by J_m, and the set of branches associated with all junction nodes by $\hat{J}_m$. Let $\tilde{J}_m \triangleq J_m \cup \hat{J}_m$ be the set of all branches over multicast group $m \in M$.

The *parent* of a receiver/junction node $r \in \tilde{R}_m$ refers to the closest junction/source node in the "upstream" path toward the source. Similarly the parent of a branch $j \in \tilde{J}_m$, if it exists, is the closest branch in the "upstream" path toward the source. We denote the parent of node $r \in \tilde{R}_m$ by $\Pi_m(r)$ and the parent of branch $j \in \tilde{J}_m$ by $\pi_m(j)$. The *children* of a junction/source node $r \in \hat{R}_m \cup \{s_m\}$ are the set of receiver/junction nodes which have r as their parent and it will be denoted by $\mathfrak{Ch}_m(r)$.

2.2 The Optimization Problem

We assume that we have a unique user connected to each receiver node $r \in R$. For each user we have a utility function $U_r(x_r)$, where x_r is the rate at which r receives data. This utility function can

be interpreted both from the point of perceived quality of service received and the amount paid in order to receive the service. Since there is a unique user connected to each receiver node, we will use the same notation when we talk about the receiver nodes or the users connected to these nodes.

We make the following assumptions:

Assumption 1 *The utility functions $U_r(x_r)$ are strictly concave, differentiable and increasing.*

Assumption 2 *The rate x_r is assumed to be a continuous variable.*

Assumption 3 *Rate allocations are done along fixed multicast trees with fixed number of users.*

Assumption 1 reflects the fact that users have diminishing returns on the goods consumed. Assumption 2 is an approximation to the actual problem. This approximation is made in most multirate multicast problems in the literature, e.g. [5, 13, 15], with notable exceptions [32, 33]. Based on these assumptions we formulate the following static network multicast problem for the model of Section 2.1

$$\max_{x_r, r\in R} \sum_{r\in R} U_r(x_r) \qquad \textbf{Max 1}$$

such that:

$$\sum_{m\in M} \max_{r\in R_{l,m}} x_r \le c_l, \qquad \forall\, l \in L \tag{2.1}$$

$$x_r \ge 0, \qquad \forall\, r \in R \tag{2.2}$$

Constraint (2.1) is also known as the *capacity constraint*. For this constraint to be satisfied, on each link, the totality of the rates used by each multicast tree cannot exceed the link capacity. The capacity constraint insures that for all the multicast trees, the rate on each branch of a tree is less than or equal to the rate on its parent branch.

Noting that the constraints (2.1) and (2.2) make the set of feasible solutions (x's) compact, and since $U_r's$ are assumed to be continuous, Weierstrass's Theorem [38, p.823] guarantees the existence of a solution to **Max 1**.

3 Properties of Optimal Solutions of Problem Max 1

In this section we derive properties of an optimal solution of Problem **Max 1**. These properties provide guidelines for the development of a Market-based decentralized algorithm that satisfies the informational constraints imposed by the nature of the multicast problem (described in Part II of the paper) and achieves the solution to Problem **Max 1**. The significance of each of the results developed in this section will be discussed at the end of the section.

We proceed by considering the following problem:

$$\max_{x_r, r\in R} \sum_{r\in R} U_r(x_r) \qquad \textbf{Max 2}$$

such that:

$$\sum_{m\in M} x_{r_{l,m}} \le c_l, \qquad \forall\, l \in L,\ \forall\, r_{l,m} \in R_{l,m}. \tag{3.1}$$

$$x_{r_{l,m}} \ge 0, \qquad \forall\, l \in L,\ \forall\, r_{l,m} \in R_{l,m}. \tag{3.2}$$

where $r_{l,m}$ denotes a receiver on the m^{th} multicast tree that employs link l.

Since the set of equations (3.1) is equivalent to the set of equations (2.1), Problem **Max 1** is equivalent to Problem **Max 2**. So any optimal solution of Problem **Max 2** is also an optimal solution for Problem **Max 1**.

Let $|M|$ denote the number of multicast trees in the network. We define the set

$$\Phi(l) \triangleq \{\left(r_{l,1}, \ldots, r_{l,|M|}\right) : r_{l,i} \in R_{l,i}, 1 \le i \le |M|\},$$

to be the set of $|M|$-tuples, each tuple consisting of one receiver from each multicast tree, and every receiver of each tuple is downstream from link $l \in L$ on its respective multicast tree. We note that the

number of elements in $\Phi(l)$ corresponds to the number of constraints for link l in the set of equations (3.1). We denote by r_l an element of $\Phi(l)$, and by $r_{l,m}$ a receiver on the m^{th} multicast tree of r_l. Note that if for some multicast tree $m \in M$ and some link $l \in L$, $R_{l,m} = \emptyset$, i.e. link l is not part of the multicast tree m, then we let the $r_{l,m}$ entry of the r_l tuple be empty, i.e. no receiver from multicast tree m is assigned to any of the r_l tuples. We define the set $\Phi(l,r) \triangleq \{(r_{l,1}, \ldots, r_{l,|M|}) : r \in \{r_{l,1}, \ldots, r_{l,|M|}\}, r_{l,i} \in R_{l,i}, 1 \leq i \leq |M|\}$ to be a subset of $\Phi(l)$ where all the tuples contain receiver r.

Using the above notation we can rewrite equation (3.1) as follows:

$$\sum_{m \in M} x_{r_{l,m}} \leq c_l, \qquad \forall\, l \in L,\ \forall\, r_l \in \Phi(l)$$

Then, the Lagrangian function for Problem **Max 2** can be expressed as follows:

$$\Lambda(x, \gamma) \triangleq \sum_{r \in R} U_r(x_r) - \sum_{l \in L} \sum_{r_l \in \Phi(l)} \gamma_{r_l} \Big(\sum_{m \in M} x_{r_{l,m}} - c_l \Big). \tag{3.3}$$

where $\gamma \triangleq \{\gamma_{r_l} : \gamma_{r_l} \in R_+, r_l \in \Phi(l), l \in L\}$.

Assume that γ and x^* satisfy:

$$\gamma_{r_l} \Big(\sum_{m \in M} x^*_{r_{l,m}} - c_l \Big) = 0, \tag{3.4}$$

$$\Big(\frac{\partial U_r(x_r)}{\partial x_r} - \sum_{l \in \mathcal{L}_r} \sum_{r_l \in \Phi(l,r)} \gamma_{r_l} \Big) \Big|_{x_r = x^*_r} = 0. \tag{3.5}$$

where $\mathcal{L}_r$ is defined to be the set of links connecting $r \in R_m$ to the source s_m.

Note that by construction $\Lambda(x, \gamma) \geq \sum_{r \in R} U_r(x_r)$ for all feasible x, and $\Lambda(x^*, \gamma) = \sum_{r \in R} U_r(x^*_r)$. Hence, if x^* turns out to be a local maximizer of $\Lambda(x, \gamma)$ then it is also a local maximizer of Problem **Max 2**.

Since $U_r(x_r)$ are strictly concave for all $r \in R$, $\Lambda(x, \gamma)$ is strictly concave for all γ. Equation (3.5) along with Theorem [2, Theorem 3.4.3] give us that x^* is a global maximizer of $\Lambda(x, \gamma)$, which implies that x^* is a global maximizer of Problem **Max 2**.

From equation (3.5) it follows that the rate demanded by user r at an optimal solution of Problem **Max 1** can be written as a function of the shadow prices γ. In particular:

$$x^*_r \triangleq x_r \Big(\sum_{l \in \mathcal{L}_r} \sum_{r_l \in \Phi(l,r)} \gamma_{r_l} \Big). \tag{3.6}$$

We define

$$p(r, \gamma) \triangleq \sum_{l \in \mathcal{L}_r} \sum_{r_l \in \Phi(l,r)} \gamma_{r_l} \tag{3.7}$$

to be user r's *service price*, which gives

$$x^*_r = x_r(p(r, \gamma)). \tag{3.8}$$

The underlying intuition behind the above equations is that $p(r, \gamma)$ represents the sum of the shadow prices for the constraints involving user r. Combining equations (3.6) and (3.7) we get equation (3.8), which shows that at an optimal solution of Problem **Max 2** the demand of user r is a function of $p(r, \gamma)$.

In the rest of the section we will prove a series of theorems which describe properties of any user r's service price and demand at an optimal solution of Problem **Max 2**.

Property 1 For any $l \in L$ and any $r_l \in \Phi(l)$, $\gamma_{r_l} > 0$ implies that $x_{r_{l,m}} = \max_{r' \in R_{l,m}} x_{r'}(p(r', \gamma))$ for any $m \in M$.

This result is intuitively expected since only the constraints that are active have a positive Lagrange multiplier. For any link $l \in L$, the active constraints are the ones for which the sum of rates is equal to capacity.

Proof We prove the theorem by contradiction. Assume that there exists $r_l \in \Phi(l)$ such that $\gamma_{r_l} > 0$ and $x_{r_{l,i}}(p(r_{l,i})) \neq \max_{r' \in R_{l,i}} x_{r'}(p(r', \gamma))$ for some $i \in M$. Then there exists an $r \in R_{l,i}$ such that $x_r(p(r,\gamma)) = \max_{r' \in R_{l,i}, \gamma} x_{r'}(p(r', \gamma))$. Since $\gamma_{r_l} > 0$, this implies by equation (3.3) that

$$\sum_{m \in M} x_{r_{l,m}}(p(r_{l,m}, \gamma)) = c_l.$$

Then

$$\sum_{m \in \{M - \{i\}\}} x_{r_{l,m}}(p(r_{l,m}, \gamma)) + x_r(p(r,\gamma)) > c_l \tag{3.9}$$

and since $\left(r_{l,1}, \ldots, r_{l,i-1}, r, r_{l,i+1}, \ldots, r_{l,|M|}\right) \in \Phi(l)$, (3.9) contradicts (3.1).

To proceed further we need to define the concepts "subtree" and of the "splitting tree", as well as related terminology.

Definition 1 *Let $m \in M$ be a multicast tree and $r \in \hat{R}_m$. Let j be the branch satisfying $r = \mathfrak{r}(j)$. The set of links of branch j together with the set of links and nodes downstream of branch j on multicast tree m will be called a* subtree *of m and it will be denoted by $T_{j,m}$. Branch j will be called the* root branch *of this subtree, while r will be called the* root node *of this subtree. A subtree is* proper *if there exists a link which is part of the multicast tree but which is not part of the subtree.*

Note that a subtree is a set of links and nodes. Branches of the multicast tree m, which have the links be elements of $T_{j,m}$, are subsets of $T_{j,m}$.

Definition 2 *Let γ be the Lagrange multipliers of Problem **Max 2** at an optimal solution. For any $m \in M$ and $r \in R_m$, we define receiver r's* splitting tree *to be the set of links and nodes denoted by $\mathcal{T}_r(\gamma)$. $\mathcal{T}_r(\gamma)$ is the largest multicast subtree of which r is a member and on which the rate demanded by r is greater than or equal to the rate on any of the branches of this multicast subtree.*

We make the observation that for any multicast tree $m \in M$ and any receiver $r \in R_m$, the splitting tree of r is a function of the rate demanded by the users of the multicast tree m, and by (3.6) the rate demanded by the users is a function of γ.

The following terminology is related to the concept of a splitting tree, and is needed for future proofs.

Definition 3 *We define the* level *of a receiver node to be 0. We define the* level *of a junction node j to be one plus the level of the highest level junction node downstream of j.*

Definition 4 *We define the* level *of the user i splitting tree (denoted by $\ell(\mathcal{T}_i(\gamma))$), to be the level of the root node of $\mathcal{T}_i(\gamma)$.*

The concept of a splitting tree is needed later in the computation of the optimal receiver prices. In Properties 2, 3 and 4 we show that only the links of a receiver's splitting tree contribute to the receiver's service price.

Let the set of links of $\mathcal{T}_r(\gamma)$ be denoted by $\mathfrak{L}_r(\gamma) \triangleq L \cap \mathcal{T}_r(\gamma)$, and the set of receivers of $\mathcal{T}_r(\gamma)$ be denoted by $\mathfrak{R}_r(\gamma) \triangleq R \cap \mathcal{T}_r(\gamma)$, and let $\overline{\mathfrak{R}}_r(\gamma) \triangleq \{h \in \mathfrak{R}_r(\gamma) : x_h(p(h,\gamma)) = x_r(p(r,\gamma))\}$. We denote by $\overline{\mathfrak{L}}_r(\gamma)$ the set of links on the splitting tree $\mathcal{T}_r(\gamma)$ on which the rate allocated is equal to $x_r(p(r,\gamma))$. We illustrate the above concepts with the following example:

Example 1. Consider Figure 2.

For nodes **8** and **9** we have: the splitting trees $\mathcal{T}_{\mathbf{8}}(\gamma) = \mathcal{T}_{\mathbf{9}}(\gamma) = \{\mathbf{6}, \mathbf{8}, \mathbf{9}, l_6, l_8, l_9\}$, which have level 2 and root node **3**. Also $\mathfrak{L}_{\mathbf{8}}(\gamma) = \overline{\mathfrak{L}}_{\mathbf{8}}(\gamma) = \mathfrak{L}_{\mathbf{9}}(\gamma) = \overline{\mathfrak{L}}_{\mathbf{9}}(\gamma) = \{l_6, l_8, l_9\}$, and $\mathfrak{R}_{\mathbf{8}}(\gamma) = \overline{\mathfrak{R}}_{\mathbf{8}}(\gamma) = \mathfrak{R}_{\mathbf{9}}(\gamma) = \overline{\mathfrak{R}}_{\mathbf{9}}(\gamma) = \{\mathbf{8}, \mathbf{9}\}$.

For node **7** we have: the splitting tree $\mathcal{T}_{\mathbf{7}}(\gamma) = \{\mathbf{7}, l_7\}$ which have level 2 and root node **3**. Also $\mathfrak{L}_{\mathbf{7}}(\gamma) = \overline{\mathfrak{L}}_{\mathbf{7}}(\gamma) = \{l_7\}$, and $\mathfrak{R}_{\mathbf{7}}(\gamma) = \overline{\mathfrak{R}}_{\mathbf{7}}(\gamma) = \{\mathbf{7}\}$.

For node **5** we have: the splitting tree $\mathcal{T}_{\mathbf{5}}(\gamma) = \{\mathbf{3}, \mathbf{5}, \mathbf{6}, \mathbf{7}, \mathbf{8}, \mathbf{9}, l_3, l_5, l_6, l_7, l_8, l_9\}$ which have level 3 and root node **1**. Also $\mathfrak{L}_{\mathbf{5}}(\gamma) = \{l_3, l_5, l_6, l_7, l_8, l_9\}$, $\overline{\mathfrak{L}}_{\mathbf{5}}(\gamma) = \{l_3, l_5\}$, $\mathfrak{R}_{\mathbf{5}}(\gamma) = \{\mathbf{5}, \mathbf{7}, \mathbf{8}, \mathbf{9}\}$, and $\overline{\mathfrak{R}}_{\mathbf{5}}(\gamma) = \{\mathbf{5}\}$.

For node **4** we have: the splitting tree $\mathcal{T}_{\mathbf{4}}(\gamma) = \{\mathbf{4}, l_4\}$ which have level 3 and root node **1**. Also $\mathfrak{L}_{\mathbf{4}}(\gamma) = \overline{\mathfrak{L}}_{\mathbf{4}}(\gamma) = \{l_4\}$, and $\mathfrak{R}_{\mathbf{4}}(\gamma) = \overline{\mathfrak{R}}_{\mathbf{4}}(\gamma) = \{\mathbf{4}\}$.

For node **2** we have: the splitting tree $\mathcal{T}_{\mathbf{2}}(\gamma) = \{\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}, \mathbf{6}, \mathbf{7}, \mathbf{8}, \mathbf{9}, l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8, l_9\}$. Also $\mathfrak{L}_{\mathbf{2}}(\gamma) = \{l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8, l_9\}$, $\overline{\mathfrak{L}}_{\mathbf{2}}(\gamma) = \{l_1, l_2\}$, $\mathfrak{R}_{\mathbf{2}}(\gamma) = \{\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}, \mathbf{6}, \mathbf{7}, \mathbf{8}, \mathbf{9}\}$, and $\overline{\mathfrak{R}}_{\mathbf{2}}(\gamma) = \{\mathbf{2}\}$.

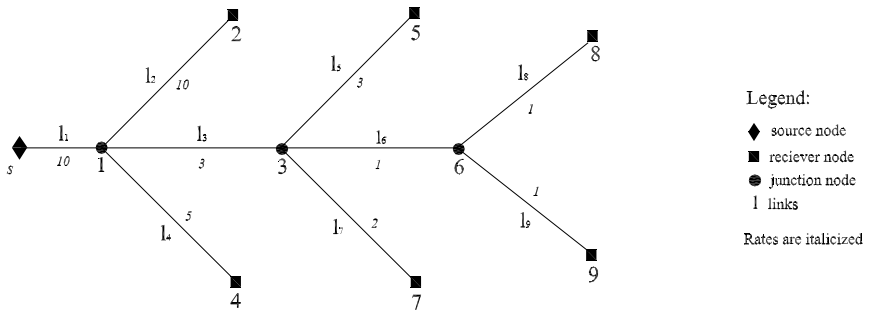

Fig. 2 A picture of a multicast tree.

□

We define

$$\lambda_l \triangleq \sum_{r_l \in \Phi(l)} \gamma_{r_l} \qquad \forall\, l \in L. \tag{3.10}$$

The intuition behind this definition is that at the optimal solution of Problem **Max 2**, λ_l represents the price per unit of rate at link $l \in L$ and it equals the sum of shadow prices of all the users using link l.

Property 2 For $m \in M$ and any $r \in R_m$,

$$\sum_{l \in \mathfrak{L}_r(\gamma)} \lambda_l = \sum_{h \in \mathfrak{R}_r(\gamma)} p(h, \gamma). \tag{3.11}$$

In words, this theorem says that the sum of service prices over all the users of a splitting tree is equal to the sum of prices per unit of rate over all the links of the splitting tree.

Proof We have the following set of equalities:

$$\sum_{l \in \mathfrak{L}_r(\gamma)} \lambda_l = \sum_{l \in \mathfrak{L}_r(\gamma)} \sum_{r_l \in \Phi(l)} \gamma_{r_l} \tag{3.12}$$

$$= \sum_{h \in \mathfrak{R}_r(\gamma)} \sum_{l \in \mathcal{L}_h} \sum_{r_l \in \Phi(l,h)} \gamma_{r_l} \tag{3.13}$$

$$= \sum_{h \in \mathfrak{R}_r(\gamma)} p(h, \gamma). \tag{3.14}$$

We first note that equation (3.12) follows from (3.10) while equation (3.14) follows from (3.7). To establish equation (3.13) we note that the left hand side is equal to the sum of the shadow prices on the links of the splitting tree of r while the right hand side is equal to the sum of the shadow prices on the links of the splitting tree or r plus shadow prices upstream of the splitting tree of r which by Property 1 have value equal to zero.

Property 3 For any $m \in M$ and any $r \in R_m$,

$$\sum_{l \in \overline{\mathfrak{L}}_r(\gamma)} \lambda_l = \sum_{h \in \overline{\mathfrak{R}}_r(\gamma)} p(h, \gamma). \tag{3.15}$$

This property establishes the fact that on any splitting tree, the service prices of the receivers demanding the maximal service are determined from the link prices of the links with the maximal rate demand.

Proof Using Property 2 we get the following equalities:

$$\begin{aligned}\sum_{l\in\overline{\mathfrak{L}}_r(\gamma)}\lambda_l+\sum_{l'\in\{\mathfrak{L}_r(\gamma)-\overline{\mathfrak{L}}_r(\gamma)\}}\lambda_{l'}&=\sum_{l\in\mathfrak{L}_r(\gamma)}\lambda_l\\&=\sum_{h\in\mathfrak{R}_r(\gamma)}p(h,\gamma)\\&=\sum_{h\in\overline{\mathfrak{R}}_r(\gamma)}p(h,\gamma)+\sum_{h'\in\{\mathfrak{R}_r(\gamma)-\overline{\mathfrak{R}}_r(\gamma)\}}p(h',\gamma)\end{aligned}\tag{3.16}$$

First we note that:

$$\begin{aligned}\sum_{h\in\{\mathfrak{R}_r(\gamma)-\overline{\mathfrak{R}}_r(\gamma)\}}p(h,\gamma)&=\sum_{h\in\{\mathfrak{R}_r(\gamma)-\overline{\mathfrak{R}}_r(\gamma)\}}\sum_{l\in\mathcal{L}_h}\sum_{r_l\in\Phi(l,h)}\gamma_{r_l}\\&\le\sum_{l\in\{\mathfrak{L}_r(\gamma)-\overline{\mathfrak{L}}_r(\gamma)\}}\sum_{r_l\in\Phi(l)}\gamma_{r_l}\\&=\sum_{l\in\{\mathfrak{L}_r(\gamma)-\overline{\mathfrak{L}}_r(\gamma)\}}\lambda_l.\end{aligned}\tag{3.17}$$

Since all the Lagrange multipliers for any $h\in\{\mathfrak{R}_r(\gamma)-\overline{\mathfrak{R}}_r(\gamma)\}$ are 0 on all the links $l\notin\{\mathfrak{L}_r(\gamma)-\overline{\mathfrak{L}}_r(\gamma)\}$, the inequality in equation (3.17) comes from the fact that the left hand side is a sum of Lagrange multipliers over the links that do not demand the maximum rate, while the right hand side is the sum of all the Lagrange multipliers over the links that do not demand maximum rate.

We also note that:

$$\begin{aligned}\sum_{h\in\overline{\mathfrak{R}}_r(\gamma)}p(h,\gamma)&=\sum_{h\in\overline{\mathfrak{R}}_r(\gamma)}\sum_{l\in\mathcal{L}_h}\sum_{r_l\in\Phi(l,h)}\gamma_{r_l}\\&\le\sum_{l\in\overline{\mathfrak{L}}_r(\gamma)}\sum_{r_l\in\Phi(l)}\gamma_{r_l}\\&=\sum_{l\in\overline{\mathfrak{L}}_r(\gamma)}\lambda_l\end{aligned}\tag{3.18}$$

Since for any $h\in\overline{\mathfrak{R}}_r(\gamma)$ all the Lagrange multipliers are over links in $\overline{\mathfrak{L}}_r(\gamma)$, the inequality in equation (3.18) comes from the fact that the left hand side is a sum of Lagrange multipliers over the links that demand the maximum rate, while the right hand side is the sum of all the Lagrange multipliers over the links that demand maximum rate.

Combining equations (3.16), (3.17) and (3.18) we get $\sum_{l\in\overline{\mathfrak{L}}_r(\gamma)}\lambda_l=\sum_{h\in\overline{\mathfrak{R}}_r(\gamma)}p(h,\gamma)$.

The following property shows that the sum of the service prices of the receivers on a particular subtree $T_{j,m}$, $m\in M$ and $j\in\tilde{J}_m$, is equal to the sum of all the link prices on $T_{j,m}$ plus the price incurred upstream from $T_{j,m}$ by the receivers with maximal demand.

Property 4 For any $m\in M$ and $j\in\tilde{J}_m$ there exists $r\in R_m\cap T_{j,m}$ such that $j\subset\mathcal{T}_r(\lambda)$, then

$$\sum_{h\in R_m\cap T_{j,m}}p(h,\gamma)=\sum_{l\in T_{j,m}\cap L}\lambda_l+\sum_{h\in\overline{\mathfrak{R}}_r(\lambda)\cap T_{j,m}}\sum_{l'\in\{\mathcal{T}_r(\gamma)\cap\mathcal{L}_r\}-\{T_{j,m}\cap L\}}\sum_{k\in\Phi(l',h)}\gamma_k\tag{3.19}$$

Proof We have:

$$\sum_{h\in R_m\cap T_{j,m}}p(h,\gamma)=\sum_{h\in R_m\cap T_{j,m}}\sum_{l\in\mathcal{L}_h}\sum_{r_l\in\Phi(l,h)}\gamma_{r_h}\tag{3.20}$$

$$\sum_{h\in R_m\cap T_{j,m}}\sum_{l\in\mathcal{L}_h}\sum_{r_l\in\Phi(l,h)}\gamma_{r_h} = \sum_{h\in R_m\cap T_{j,m}}\sum_{l\in T_{j,m}\cap L}\sum_{r_l\in\Phi(l,h)}\gamma_{r_l}$$
$$+\sum_{h\in\overline{\mathfrak{R}}_r(\lambda)\cap T_{j,m}}\sum_{l'\in\{\mathcal{T}_r(\gamma)\cap\mathcal{L}_r\}-\{T_{j,m}\cap L\}}\sum_{k\in\Phi(l',h)}\gamma_k \tag{3.21}$$
$$=\sum_{l\in T_{j,m}\cap L}\lambda_l+\sum_{h\in\overline{\mathfrak{R}}_r(\lambda)\cap T_{j,m}}\sum_{l'\in\{\mathcal{T}_r(\gamma)\cap\mathcal{L}_r\}-\{T_{j,m}\cap L\}}\sum_{k\in\Phi(l',h)}\gamma_k \tag{3.22}$$

where (3.20) follows from (3.7), (3.21) follows from the fact that all the constraints for the links in $T_{j,m}$ contain a user from $R_m\cap T_{j,m}$ and the fact that only the users in $\overline{\mathfrak{R}}_r(\lambda)$ have non-zero Lagrange multiplier for the constraints on the links upstream from branch j, while (3.22) follows from (3.10).

The following two properties give us properties of the optimal service prices.

Property 5

1. Given any multicast tree $m\in M$ and for any link $l\in L_m$, λ_l can be split into shares $\varphi_{l,r}\triangleq\sum_{r_l\in\Phi(l,r)}\gamma_{r_l}$ among all $r\in R_{l,m}$, where the rate demanded by r is equal to the rate on link l.
2. For any $r\in R_m$, the optimal service price is equal to $\sum_{l\in\mathcal{L}_r}\varphi_{l,r}$.

Property 5 says that at an optimal solution of Problem **Max 1**, the price per unit of rate of each user $r\in R$ is equal to the sum of shares of the link prices $\{\lambda_l\}_{l\in L}$ over the links used by this user. The share of the link price on each link $l\in L$ for user r is equal to the sum of Lagrange multipliers at link l which involve user r.

Proof Part 1 follows from equation (3.10) and Property 1. Part 2 follows from (3.7).

Property 6 Given fixed λ, the computation according to Property 5 of the optimal service prices on any multicast tree $m\in M$ is independent of the demands requested by the users of other multicast trees.

Proof By assumption, a service to a user r is provided along a single multicast tree, say m. From Equation (3.8) the optimal service rate for the user is a function of the price $p(r,\gamma)$ defined by (3.7). By Property 5, $p(r,\gamma)$ is determined solely by the price shares user r pays along the links of m.

In conclusion, the main contribution of this section is the presentation of an extensive set of properties which relate the prices per unit of rate on each link and the price per unit of service at an optimal solution of Problem **Max 1**. The notion of splitting tree proved to be crucial in deriving this properties. The results of this section were derived under the assumption that the information is centralized. Within the context of the multicast problem these results are vital in the development of resource allocation algorithms that satisfy the informational constraints imposes by the nature of the multicast problem, and converge to the solution of Problem **Max 1**. Specifically, Properties 5 and 6 motivate the development of an algorithm (cf. Section 4) that computes the optimal service price for each user along any multicast tree given a fixed set of prices per unit of rate at each link. Furthermore, Properties 1-4 are useful in proving properties of link price updates; these properties play a key role in the proof of convergence of the market mechanism proposed in Part II of this work.

We would also like to remark that all the properties developed in this section can be derived without assuming that user utility functions are differentiable. This can be achieved by replacing the derivative of U_r in equation (3.5) with the subgradient of U_r.

4 A Price Splitting Algorithm

In Section 3 we showed that at an optimal solution of Problem **Max 1** the shadow prices generate a set of optimal link prices and service prices, which are related by the result of Property 5. In this section we present an algorithm, which we call "price splitting algorithm", that computes along each multicast tree the price per unit of service for each user given the set of prices per unit of rate at each link.

For the rest of the section we fix $\lambda := \{\lambda_l : l \in L\}$ to be the set of prices per unit of rate on the links[1]. According to Property 5, for each given multicast group it is optimal[2] to split the cost incurred on each branch among the users using the maximum rate on that branch. Since the link prices seen by different multicast trees are going to be the same, and because of Property 6, the algorithm on each multicast group can run independently of the other multicast groups. Thus, we will describe how the algorithm works on a single multicast group $m \in M$.

We proceed as follows: In Section 4.1 we illustrate the idea of how to split the price on a branch which is common to two users. In Section 4.2 we give a detailed example of how the algorithm works in a tree of level grater than one, and explain how the cost incurred on one branch can be split among multiple children. In Section 4.3 we present a high level description of the algorithm for a general network. The formal description of the algorithm and the proof of its convergence are presented in Appendices A and B, respectively.

In the following examples, the message passing between the users and the network is done over the multicast tree by using intermediate nodes as relays. Note that this does not have to be the case. As long as users have a way of communicating with the network, the same algorithm can be used by the network in order to compute the service prices.

4.1 Example of two users with one link in common

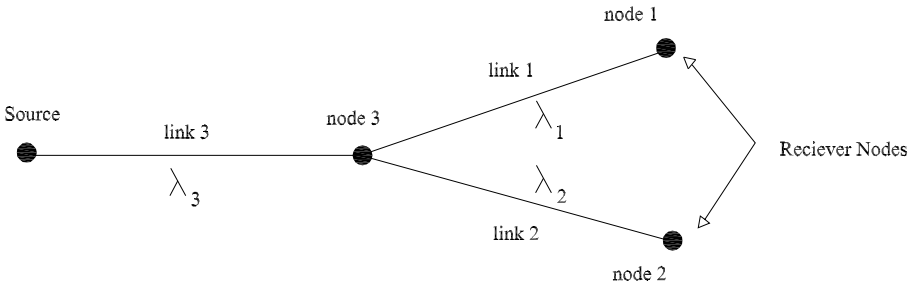

Fig. 3 Two users sharing link 3.

Consider a tree consisting of 3 links as in Figure 3. The prices per unit of rate on links 1,2,3 are $\lambda_1, \lambda_2, \lambda_3$ respectively. The optimal sharing of the price λ_3 is determined by the following algorithm:

1. Set $\alpha^{(1)} = 1/2, n = 1$ and assume w.l.o.g. that $x_1(\lambda_1) \geq x_2(\lambda_2)$.
2. If $x_1(\lambda_1 + \lambda_3) \geq x_2(\lambda_2)$ stop; the prices per unit of rate for user 1 and 2 are $\lambda_1 + \lambda_3$, λ_2, respectively. If $x_1(\lambda_1 + \lambda_3) \leq x_2(\lambda_2)$ then proceed with the following recursion.
3. Compute $x_1(\lambda_1 + \alpha^{(n)} \times \lambda_3)$ and compare it to $x_2(\lambda_2 + (1 - \alpha^{(n)}) \times \lambda_3)$.
4. (a) If $x_1(\lambda_1 + \alpha^{(n)}\lambda_3) = x_2(\lambda_2 + (1 - \alpha^{(n)})\lambda_3)$ stop; $\alpha^{(n)}\lambda_3$ and $(1 - \alpha^{(n)})\lambda_3$ provide the optimal price sharing of the price λ_3 for users 1 and 2, respectively.
   (b) If $x_1(\lambda_1 + \alpha^{(n)}\lambda_3) > x_2(\lambda_2 + (1 - \alpha^{(n)})\lambda_3)$ set $\alpha^{(n+1)} = \alpha^{(n)} + \frac{1}{2^{n+1}}$.
   (c) If $x_1(\lambda_1 + \alpha^{(n)}\lambda_3) < x_2(\lambda_2 + (1 - \alpha^{(n)})\lambda_3)$ set $\alpha^{(n+1)} = \alpha^{(n)} - \frac{1}{2^{n+1}}$.
5. Increment n and return to step 3.

[1] Note that we do not assume that λ is generated from a set of shadow prices which corresponds to an optimal solution of Problem **Max 2**.

[2] Given a set of fixed link prices λ, by an *optimal splitting* of λ we want to find a set of price shares γ's and service prices $p(r, \gamma)$ which generate user demands $x_r(p(r, \gamma)) \triangleq \text{argmax}_x\, U_r(x) - p(r, \gamma) \times x$ having the following properties: (i) For any $m \in M$, $\sum_{l \in L_m} \max_{r \in R_{l,m}} x_r \times \lambda_l = \sum_{r \in R_m} x_r(p(r, \gamma)) \times p(r, \gamma)$; and (ii) the sum of user's utility functions given $x_r(p(r, \gamma))$ is maximized. We note that although the analysis in Section 3 was done at an optimal solution of Problem **Max 2**, the proofs of Properties 2 - 6 hold for optimal splitting of arbitrary link prices.

We prove that this algorithm determines the optimal sharing of the price λ_3. Note that there are two cases:

Case 1: If $x_1(\lambda_1) \geq x_2(\lambda_2)$ and $x_1(\lambda_1 + \lambda_3) \geq x_2(\lambda_2)$ then the optimal price sharing is the following: user 1 pays for both the links it uses, i.e. links 1 and 3; user 2 is charged only the price of link 2.

Case 2: If $x_1(\lambda_1) \geq x_2(\lambda_2)$ but $x_1(\lambda_1 + \lambda_3) \leq x_2(\lambda_2)$ then users 1 and 2 have to split the price of link 3. We show that the algorithm determines as $n \to \infty$, the share of λ_3 that receiver 1 will have to pay.

The result for Case 1 follows directly from Property 5.

For Case 2 we first note that both x_1 and x_2 are assumed to be continuous, strictly monotonically decreasing functions (i.e. the demand for rate decreases monotonically in price). This implies that $g(\alpha) \triangleq x_1(\lambda_1 + \alpha\lambda_3) - x_2(\lambda_2 + (1-\alpha)\lambda_3)$ is a continuous, strictly monotonically decreasing function of α. By the Intermediate Value Theorem, since $g(0) > 0$ and $g(1) < 0$, there exists a $0 < \alpha < 1$ for which user's 1 demand equal user's 2 demand. Also by the strict monotonicity property of g, α is unique. We prove that by the construction of the algorithm the sequence $\{\alpha^{(n)}\}_{n\in\mathbb{N}}$ converges to α as $n \to \infty$.

The proof of this result follows by contradiction. Assume that $\alpha^{(n)} \xrightarrow[n\to\infty]{} \beta \neq \alpha$. Note that at each iteration the algorithm determines the value of the k^{th} digit of β written in binary expansion. Let k be the first digit (in the binary expansion) for which α and β differ. If the k^{th} digit of β is 1, then the k^{th} digit of α is 0, implying that $g(\alpha^{(k)}) < 0$. In this case, at k^{th} iteration of the algorithm, $\alpha^{(k)}$ is assigned value 0 for the k^{th} digit, which implies that the k^{th} digit of β is 0, a contradiction. On the other hand, assume that the k^{th} digit of β is 0, then the k^{th} digit of α is 1, implying that $g(\alpha^{(k)}) > 0$. In this case, at the k^{th} iteration of the algorithm, $\alpha^{(k)}$ is assigned value 1 for the k^{th} digit, which implies that the k^{th} digit of β is 1, a contradiction. So α and β are equal since their binary expansion is the same. □

Remark 1 The sequence of $\{\alpha^{(n)}\}$ generated by the above process is a convergent dyadic sequence with some limit $\dot{\alpha}$. $\dot{\alpha}$ has the property that $x_1(\lambda_1 + \dot{\alpha}\lambda_3) = x_2(\lambda_2 + (1-\dot{\alpha})\lambda_3)$, and the distance between $\alpha^{(n)}$ and $\dot{\alpha}$ is at most $\frac{1}{2^n}$.

4.2 A two level example

In this section we present via an example of a generalization of the algorithm developed in Section 4.1. In the example presented below each junction node has two children nodes. After the completion of the example we describe a procedure which reduces the case where some junction nodes have more then two children nodes to the one where each junction node has precisely two children nodes.

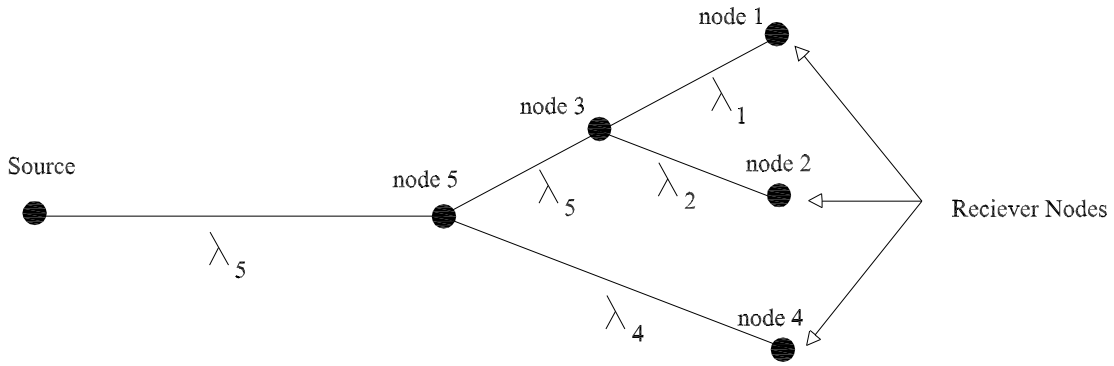

Fig. 4 A more complex example of a multi-rate multicast tree.

Consider a tree consisting of 5 links as in Figure 4. In this case the price splitting algorithm proceeds as follows:

Algorithm of node i=1,2,4:

1. Wait for $\mathfrak{Reset}$ from parent. Upon reception go to 2.
2. For node 1,2,4, upon receiving a $\mathfrak{Reset}$ from parent with a price p, return a demand interval $[x_i(p), x_i(p)]$, where $x_i(p)$ is the demand of node i for price p, $i = 1, 2, 4$. Go to step 1.

Algorithm of node i=3:

Let $D_3^{(n)}$ denote the demand interval of node 3 at iteration n; denote by $p_1^{(n)}$, $p_2^{(n)}$ the prices node 3 sends to nodes 1 and 2, respectively, at iteration n.

The algorithm at node 3 proceeds as follows: Upon receiving a $\mathfrak{Reset}$ from node 5 with price p start the initial stage.

Initial stage.

1. Send a $\mathfrak{Reset}$ to nodes 1 and 2 with prices $p_1^{(0)} = \lambda_1$, $p_2^{(0)} = \lambda_2$ respectively.
2. Node 1 returns the demand interval $[x_1(p_1^{(0)}), x_1(p_1^{(0)})]$.
   Node 2 returns the demand interval $[x_2(p_2^{(0)}), x_2(p_2^{(0)})]$.
3. Compare $x_1(p_1^{(0)})$ with $x_2(p_2^{(0)})$. Assume without loss of generality (w.l.o.g.) that $x_1(p_1^{(0)}) \geq x_2(p_2^{(0)})$. The algorithm proceeds in the same way (with obvious modifications) if the opposite inequality is true.
4. Set $p_1^{(1)} = \lambda_1 + p$, $p_2^{(1)} = p_2^{(0)} = \lambda_2$.
5. Send a $\mathfrak{Reset}$ to nodes 1 and 2 with prices $p_1^{(1)}$, $p_2^{(1)}$ respectively. Wait for the demand intervals from nodes 1 and 2.
   (a) If $x_1(p_1^{(1)}) \geq x_2(p_2^{(1)})$, send the interval $[x_1(p_1^{(1)}), x_1(p_1^{(1)})]$ to node 5 and stop.
   (b) If $x_1(p_1^{(1)}) < x_2(p_2^{(1)})$ set
   $p_1^{(2)} = p_1^{(1)} - \frac{1}{2}p$
   $p_2^{(2)} = p_2^{(1)} + \frac{1}{2}p$,
   and send the interval $[x_1(p_1^{(1)}), x_2(p_2^{(1)})]$ to node 5.

Stage n.

1. Send a $\mathfrak{Reset}$ to nodes 1 and 2 with prices $p_1^{(n)}$, $p_2^{(n)}$ respectively.
2. Node 1 returns the demand interval $[x_1(p_1^{(n)}), x_1(p_1^{(n)})]$.
   Node 2 returns the demand interval $[x_2(p_2^{(n)}), x_2(p_2^{(n)})]$.
3. Compare $x_1(p_1^{(n)})$ with $x_2(p_2^{(n)})$.
   (a) If $x_1(p_1^{(n)}) = x_2(p_2^{(n)})$ send the demand interval $[x_1(p_1^{(n)}), x_1(p_1^{(n)})]$ to node 5 and stop.
   (b) If $x_1(p_1^{(n)}) > x_2(p_2^{(n)})$ set
   $p_1^{(n+1)} = p_1^{(n)} + \frac{1}{2^n}p$
   $p_2^{(n+1)} = p_2^{(n)} - \frac{1}{2^n}p$,
   send the interval $[x_2(p_2^{(n)}), x_1(p_1^{(n)})]$ to node 5, and go to Step 1 of *Stage n+1*.
   (c) If $x_1(p_1^{(n)}) < x_2(p_2^{(n)})$ set
   $p_1^{(n+1)} = p_1^{(n)} - \frac{1}{2^n}p$
   $p_2^{(n+1)} = p_2^{(n)} + \frac{1}{2^n}p$,
   send the interval $[x_1(p_1)^{(n)}, x_2(p_2^{(n)})]$ to node 5, and go to Step 1 of *Stage n+1*.

Algorithm of node i=5:

Let $D_5^{(n)}$ denote the demand interval of node 5 at iteration n; denote by $p_3^{(n)}$, $p_4^{(n)}$ the prices node 5 sends to nodes 3 and 4, respectively at iteration n.

The algorithm at node 5 proceeds as follows:
Upon receiving a $\mathfrak{Reset}$ from source node with price λ_5 start the initial stage.

Initial stage.

1. Send a $\mathfrak{Reset}$ to nodes 3 and 4 with prices $p_3^{(0)} = \lambda_3$, $p_4^{(0)} = \lambda_4$, respectively.

2. Node 3 starts returning[3] demand intervals of form $[\underline{x}_3(p_3^{(0)}), \overline{x}_3(p_3^{(0)})]$[4].
   Node 4 returns the demand interval $[x_4(p_4^{(0)}), x_4(p_4^{(0)})]$.
3. Wait until $x_4(p_4^{(0)}) \notin [\underline{x}_3(p_3^{(0)}), \overline{x}_3(p_3^{(0)})]$ [5].
4. If $\underline{x}_3(p_3^{(0)}) \geq x_4(p_4^{(0)})$ then:
   (a) Set $p_3^{(1)} = \lambda_3 + \lambda_5$, $p_4^{(1)} = p_4^{(0)} = \lambda_4$.
   (b) Send a $\mathfrak{Reset}$ to nodes 3 and 4 with prices $p_3^{(1)}$, $p_4^{(1)}$ respectively. Wait for the demand intervals from nodes 3 and 4.
      i. If $\underline{x}_3(p_3^{(1)}) \geq x_4(p_4^{(1)})$, send the interval $[\underline{x}_3(p_3^{(1)}), \overline{x}_3(p_3^{(1)})]$, and every subsequent demand interval received from node 3, upstream to the source node. Stop the algorithm at this node.
      ii. If $\overline{x}_3(p_3^{(1)}) < x_4(p_4^{(1)})$, set
      $p_3^{(2)} = p_3^{(1)} - \frac{1}{2}\lambda_5$
      $p_4^{(2)} = p_4^{(1)} + \frac{1}{2}\lambda_5$,
      and send the interval $[\underline{x}_3(p_3^{(1)}), x_4(p_4^{(1)})]$ to the source node. Go to stage 2 of the algorithm.
5. If $x_4(p_4^{(0)}) \geq \overline{x}_3(p_3^{(0)})$ then:
   (a) Set $p_3^{(1)} = p_3^{(0)} = \lambda_3$, $p_4^{(1)} = p_4^{(0)} + \lambda_5$.
   (b) Send a $\mathfrak{Reset}$ to nodes 3 and 4 with prices $p_3^{(1)}$, $p_4^{(1)}$ respectively. Wait for the demand intervals from node 3 and 4.
      i. If $x_4(p_4^{(1)}) \geq \overline{x}_3(p_3^{(1)})$, send the interval $[x_4(p_4^{(1)}), x_4(p_4^{(1)})]$ upstream to the source node, and stop the algorithm at this node.
      ii. If $x_4(p_4^{(1)}) < \overline{x}_3(p_3^{(1)})$, set
      $p_3^{(2)} = p_3^{(1)} + \frac{1}{2}\lambda_5$
      $p_4^{(2)} = p_4^{(1)} - \frac{1}{2}\lambda_5$,
      send the interval $[x_4(p_4^{(1)}), \overline{x}_3(p_3^{(1)})]$ to the source node. Go to stage 2 of the algorithm.

Stage n.

1. Send a $\mathfrak{Reset}$ to nodes 3 and 4 with prices $p_3^{(n)}$, $p_4^{(n)}$, respectively.
2. Node 3 starts returning demand intervals of form $[\underline{x}_3(p_3^{(n)}), \overline{x}_3(p_3^{(n)})]$.
   Node 4 returns the demand interval $[x_4(p_4^{(n)}), x_4(p_4^{(n)})]$.
3. Wait until $x_4(p_4^{(n)}) \notin [\underline{x}_3(p_3^{(n)}), \overline{x}_3(p_3^{(n)})]$.
   (a) If $x_4(p_4^{(1)}) = \underline{x}_3(p_3^{(n)}) = \overline{x}_3(p_3^{(1)})$, send the interval $[x_4(p_4^{(1)}), x_4(p_4^{(1)})]$ to the source node and stop the process at this node.
   (b) If $\underline{x}_3(p_3^{(n)}) > x_4(p_4^{(n)})$ set
   $p_3^{(n+1)} = p_3^{(n)} + \frac{1}{2^n}\lambda_5$
   $p_4^{(n+1)} = p_4^{(n)} - \frac{1}{2^n}\lambda_5$,
   send the interval $[x_4(p_4^{(n)}), \overline{x}_3(p_3^{(n)})]$ to the source node, and go to Step 1 of *Stage n+1*.
   (c) If $\overline{x}_3(p_3^{(n)}) < x_4(p_4^{(n)})$ set
   $p_3^{(n+3)} = p_3^{(n)} - \frac{1}{2^n}\lambda_5$
   $p_4^{(n+1)} = p_4^{(n)} + \frac{1}{2^n}\lambda_5$,
   send the interval $[\underline{x}_3(p_3)^{(n)}, x_4(p_4^{(n)})]$ to the source node, and go to Step 1 of *Stage n+1*.

Algorithm of the source node:

1. The source node will send a $\mathfrak{Reset}$ to node 5 with price λ_5. This $\mathfrak{Reset}$ will start the algorithm.

In the example presented in this section every junction node has two children. The case where some junction nodes have more then two children can be reduced to the one where each junction node has

[3] At node 3 there is an optimization process running at the same time with the one at node 5. Node 3 will continuously be sending demand intervals to node 5. These demand intervals have the property that they are nested into one another and decreasing in size.

[4] Note that in this case this interval may not be just a singleton, i.e. $\underline{x}_3(p_3^{(0)}) < \overline{x}_3(p_3^{(0)})$

[5] If the demand of node 4 is in all the demand intervals of node 3, then the optimal demand of node 4 equal to the optimal demand of node 3. In this case we have reached the optimal price split.

exactly two children. The reduction is based on the following observation: For a given set of link prices per unit of rate, links that have infinite capacity and zero price per unit of rate can be added to or removed from the multicast tree without altering the problem of price sharing along the tree. For example, if in the system of Figure 4 the price per unit of rate on link 3 (i.e. λ_3) is zero and link 3's capacity is infinite, then link 3 can be removed, links 1, 2 and 4 can be connected to link 5, and the price per unit of service for each user in the new multicast tree is the same as in the original tree. The reason for this equivalence is the following. By Property 5, the optimal service price for each receiver node is equal to the sum of the shares of the shadow prices of the constraints that are active in Problem **Max 2**. Therefore, by removing all links that have zero price per unit of rate, the service price will remain the same for all users (and this in turn implies that the rate demanded by each user will remain the same).

The above observation has the following general implication. Suppose there is a junction node i that has $k > 2$ children $j_1, j_2, \ldots, j_k$. Introduce *intermediate nodes* i_1 and i_2 such that i, i_1 are linked together by a link that has infinite capacity and zero price per unit of rate, and the same is true for the link connection i and i_2. We call such links *intermediate links*. Link half of the children of i to node i_1 and the other half to node i_2. By repeating the above process and introducing intermediate nodes one can end up with a multicast tree where each junction node has exactly two children. As a result of this construction, only the links connected to the original children $j_1, j_2, \ldots, j_k$ have a non-zero price per unit of rate. In the rest of the section we shall assume that each junction node has exactly two children.

4.3 Price-Splitting Algorithm Description

To determine the optimal service price[6] for each user, given a fixed link price, the algorithm requires that each node communicate with its parent and children. The downstream communication (i.e. the communication from the source to the users) will consist of a $\mathfrak{Reset}$ packet/signal to which a price is appended, while the upstream communication (i.e. the communication from the users to the source) will consist of a *demand interval*. In the rest of Section 4.3 we explain *qualitatively* how the algorithm works. A formal description of the price-splitting algorithm appears in Appendix A.

Initiation step: The algorithm begins with the source node sending a $\mathfrak{Reset}$ signal downstream the multicast tree. The price attached to each $\mathfrak{Reset}$ sent on any branch of the multicast tree is the price per unit of rate on that branch. Upon receiving a $\mathfrak{Reset}$ signal each junction node saves the price per unit of rate of the branch corresponding to it, and sends a $\mathfrak{Reset}$ to its children.

Iterative step: When a receiver node receives a $\mathfrak{Reset}$ signal with a price p from its parent, it computes its demand, given price p, and sends this demand upstream to its parent node.

Since each junction node has two children, and each one of them may be demanding a different rate, the junction node sends its demand request to its parent node in the form of a demand interval. The demand interval of a junction node at a particular stage in the process is generated as follows: The lower bound of the demand interval is equal to the minimum of the lower bounds of its childrens' demand intervals, while the upper bound of the demand interval is equal to the maximum of the upper bounds of its childrens' demand intervals. Note that upon the arrival of a new demand interval from one of its children, a junction node's demand interval may change, in which case the new demand interval is transmitted upstream to the parent. For consistency we consider the demand requested by a receiver node as a demand interval formed by a singleton.

The goal of each junction node $i \in \hat{R}_m$ is to decide how to split the upstream cost incurred by the receivers of tree $T_{i,m}$ based on the demand intervals received from its children nodes. This is done by splitting the price per unit of rate attached to the $\mathfrak{Reset}$ signal received by node i, among the children of i. The way that this price is split proceeds as follows.

Step 1: Junction node i receives from its parent node a $\mathfrak{Reset}$ signal with a price p; this price has to be split among the children of $T_{i,m}$.

Step 2: Node i sends a $\mathfrak{Reset}$ to each child node containing the price of the child's branch.

Step 3: Upon receiving demand intervals from each child, node i waits until one of two events occur:

3.1 The demand interval of one child, say j, is larger then the demand interval of the other child.

3.2 The demand intervals of the two children of i are overlapping, and are arbitrarily small.

[6] By the optimal set service prices given a set of link prices we mean the set of service prices generated from the set of link prices which generate demands which maximize the sum of the user utility functions.

Step 4:

4.1 If **3.1** is true, node i sends a $\mathfrak{Reset}$ signal to node j. The price of this $\mathfrak{Reset}$ signal is equal to the price p plus the price of the branch associated with node j.

4.2 If **3.2** is true, node i chooses one of the two nodes at random, say node k, and sends to k a new $\mathfrak{Reset}$ signal. The price of this $\mathfrak{Reset}$ is equal to the price p plus the price of the branch associated with node k.

Step 5:

5.1 If **4.1** is true, node i determines how price p shall be split among its children nodes as follows: Node i receives a new demand interval from node j. As proved in Section B, this demand interval decreases with time in a nested fashion. Node i waits until $j's$ demand interval becomes disjoint from the other child's demand interval. If the demand interval of j is larger then that of the other child of i, j will incur the whole price p, otherwise the two children of i will share the price p.[7]

5.2 If **4.2** is true, node i determines how price p shall be split among its children nodes as follows: Node i receives a new demand interval from node k. As proved in Section B, this demand interval decreases with time in a nested fashion. Node i waits until $k's$ demand interval becomes disjoint from the other child's demand interval. If the demand interval of k is larger then that of the other child of i, k will incur the whole price p, otherwise the two children of i will share the price p.[7]

Step 6: Based on 5, junction node i knows if price p is incurred fully by one of its children or it is to be split in some ratio (that has to be determined) between its two children. If the price is to be split among its children, junction node i determines the optimal price share by a procedure which is formally presented in Appendix A.

In Appendix A we present a formal description of the above price splitting algorithm. We prove the following properties of the algorithm in Appendix B and C respectively:

Theorem 1 *The price splitting algorithm determines asymptotically the optimal service prices given a fixed price per unit of rate on each link.*

Theorem 2 *Let $\lambda := \{\lambda_l, l \in L\}$ be a set of link prices and $x_i(p(\lambda))$, $i = 1, 2, \ldots, N$ be the user demands corresponding to $p(\lambda)$ determined by Theorem 1. Define the excess demand at link l for given λ by:*

$$z_l(\lambda) \triangleq \sum_{m \in M} \max_{r \in R_{l,m}} x_r(p(r, \lambda)) - c_l. \tag{4.1}$$

For any $l \in L$ for which $z_l(\lambda) \neq 0$, the price splitting algorithm determines the sign of the excess demand function $z_l(\lambda)$ in a finite number of iterations.

The results of Theorems 1 and 2 are key in the development of the market based mechanism, presented in Part II of this paper, which solves Problem **Max 1**.

As before, the above price splitting algorithm can be implemented for both a flat and hierarchical architecture. Although we have described the algorithm from the perspective of passing information among the various nodes of the multicast trees, in our formulation the network manager can centrally compute the user service prices by iteratively exchanging information with the users. This is due to the fact that the network manager is assumed to have full information about the network topology, the links forming the multicast trees, and the prices per unit of rate on all the links. This type of implementation of the price splitting algorithm is used in Part II of the paper, where we also present numerical results.

5 Conclusion

In Part I of this two-part paper we presented a distributed algorithm which computes, for a set of fixed link prices per unit of rate, the set of user prices per unit of service that maximizes the sum of the users' utilities. Thus, implicitly the algorithm determines how a link's price is spent among the users that

[7] It may happen that these two demand intervals do not become disjoint in finite time. This is the case when both set of demand intervals converge to the same singleton. To alleviate this problem one can decide to split the price p among the children of i when the demand intervals node's i two children become arbitrarily small.

use the link. This algorithm plays a key role in Part II of the paper where a decentralized market-based mechanism, that achieves welfare-maximizing resource allocations, is developed.

The main contributions of Part I of the paper are: (1) the development of properties of an optimal solution of the centralized analogue of the multirate multicast problem, (2) the specification of a distributed algorithm which, under the price-taking assumption, solves the price-sharing problem that arises in multirate multicast and determines the optimal service prices per unit of rate for each user, given a set of fixed prices per unit of rate on each link.

Appendices

A Formal Setup

In Section 4.2 we introduced the notion of "intermediate nodes". With this notion we showed that a general tree is equivalent to a tree where each node has at most two children.

We now describe the formal setup of the algorithm for a tree where each node has two children. The algorithm deals with three types of nodes: (i) receiver nodes, (ii) junction nodes, (iii) source nodes. We describe the algorithmic procedure for each type of nodes separately.

Algorithm of Receiver Node i***:***

On receiving a $\mathfrak{Reset}$ with price $\mathbf{p}$ from the parent:
Compute the demand $x_i(\mathbf{p})$.
Send the demand interval $D_i = [x_i(\mathbf{p}), x_i(\mathbf{p})]$ to parent.

Algorithm of Junction Node i:

Denote by n the iteration number.
Define $\overline{D}_i \triangleq [\overline{b}_i, \overline{B}_i]$ to be the i^{th} node's current demand interval,
and $D_i \triangleq [b_i, B_i]$ to be the i^{th} node's previous demand interval.
Define $D_i(j) \triangleq [b_j, B_j]$, $j \in \mathfrak{Ch}(i) := \{j_1, j_2\}$, to be the j^{th} node's demand interval that node i keeps for node j .
Define p_j to be the price per unit of rate on the branch connecting nodes i and $j \in \mathfrak{Ch}(i)$.

On receiving a $\mathfrak{Reset}$ with a price $\mathbf{p}$ from parent the algorithm starts
Set demand interval $\overline{D}_i = D_i = [0, \infty]$, and $n = 1$.
For all $j \in \mathfrak{Ch}(i)$ set demand intervals $D_i(j) = [0, \infty]$.
Send $\mathfrak{Reset}$ to all children with price p_j to j^{th} child.
Wait until there exists $j_i \in \mathfrak{Ch}(i)$, say j_1, such that $D_i(j_1) > D_i(j_2)$[89]
Send $\mathfrak{Reset}$ to child j_1 with branch price $(p_{j_1} + \mathbf{p})$.
Wait until $D_i(j_1) \cap D_i(j_2) = \emptyset$.[10]
While waiting for $D_i(j_1) \cap D_i(j_2) = \emptyset$.
Upon receiving a demand interval from a child, set
$\overline{D}_i = [\max\{b_i, \min\{b_{j_1}, b_{j_2}\}\}, \min\{B_i, \max\{B_{j_1}, B_{j_2}\}\}$.
Send $\overline{D}_i$ to parent and set $D_i = \overline{D}_i$.
If $D_i(j_1) > D_i(j_2)$ go to $\circledast$.
Otherwise, the price $\mathbf{p}$ is split between j_1 and j_2.
The optimal splitting of $\mathbf{p}$ is achieved as follows:
Set $\alpha_{j_1} = \alpha_{j_2} = \frac{1}{2}$ and increment n.
Send $\mathfrak{Reset}$ to j_1, j_2 with prices $(p_{j_1} + \alpha_{j_1} \times \mathbf{p})$,$(p_{j_1} + \alpha_{j_2} \times \mathbf{p})$ respectively.
Set $D_i(j_1) = D_i(j_2) = [0, \infty)$.
$\ominus$ Wait for demands from children.
While $D_i(j_1) \cap D_i(j_2) \neq \emptyset$
Upon receiving a demand interval from a child, set
$\overline{D}_i = [\max\{b_i, \min\{b_{j_1}, b_{j_2}\}\}, \min\{B_i, \max\{B_{j_1}, B_{j_2}\}\}$.
Send $\overline{D}_i$ to parent and set $D_i = \overline{D}_i$.
If $D_i(j') > D_i(j'')$, where $j', j'' \in \{j_1, j_2\}$, then:
Set $\overline{D}_i = [\max\{b_i, \min\{b_{j'}, b_{j''}\}\}, \min\{B_i, \max\{B_{j'}, B_{j''}\}\}$.

Send $\overline{D}_i$ to parent and set $D_i = \overline{D}_i$.
Set $D_i(j') = D_i(j'') = [0, \infty)$.
Add $1/2^n$ to $\alpha_{j'}$, subtract $1/2^n$ from $\alpha_{j''}$ and increment n
Send $\mathfrak{Reset}$ to j', j'' with price $(p_{j'} + \alpha_{j'} \times p_i)$, $(p_{j'} + \alpha_{j''} \times p_i)$
respectively, and loop back to $\ominus$.

$\circledast$ The price $\mathbf{p}$ is incurred by node j_1. While waiting for a reset from the parent node, relay all the demand intervals received from node j_1 to parent.

Algorithm of the Source Node s:
On the **start** of the algorithm:
Send a $\mathfrak{Reset}$ to its children, with the price p_j to j^{th} child, $j \in \mathfrak{Ch}(s)$.

In the next section we shall prove that the algorithm converges to the optimal service price for each user.

B Proof of Theorem 1

Prior to proving the convergence of the algorithm to optimal price sharing, we establish a few lemmas.

Lemma 1 *In the algorithm, for any upstream price β_i at junction node i, the value of the optimal rate demand on branch i given β_i is in all the demand intervals at node i.*

Proof The assertion of Lemma 1 follows directly from the construction of the algorithm and from the fact that utilities are strictly concave and monotonically increasing.

Lemma 2 *For any upstream price β_i, in the algorithm of junction node i the demand intervals are nested and the sequence of demand intervals converges to a point. This point is the optimal rate demand given the upstream price β_i.*

Proof The fact that the intervals are nested follows from construction. The fact that the sequence of demand intervals converges to a point can be established by induction.

Basis of induction (level 1 nodes): If i is a node of level 1 then its children are receiver nodes and their demand intervals are singletons. Since the utility functions of the users are monotonically increasing, strictly concave, differentiable, and the price share assigned to each user is a converging dyadic sequence, the demand interval for node i will converge to a singleton.(see Remark 1)

Induction step (level n nodes): Assume that node i has level n and for any node of level $n-1$ the demand intervals converge to a point. This implies that at each iteration of the algorithm for node i the demand intervals of the children become disjoint in finite time[11]. Therefore the following facts are true: **(F1)** successive iterations/updates of the algorithm (for node i) occur in finite time; **(F2)** for each child j of i and for any given upstream price share α_j the demand interval of j converges to a point and contains the optimal demand of j given α_j; and **(F3)** by the construction of the algorithm, for each child node j of i the sequence of upstream price shares α_j resulting from the algorithm's iterations/updates is a diadic sequence that converges to an optimal upstream price share. By strict concavity of the utility functions and facts **F1-F3** the demand intervals for node i will converge to a singleton.

[8] $D_i(j') \geq D_i(j)$ means that $\forall x \in D_i(j'), y \in D_i(j) \implies x \geq y$.

[9] This step of the algorithm determines a node j_1 to see if the price $\mathbf{p}$ will be split or not between the two children of i. If after some time $D_i(j_1) \cap D_i(j_2) \neq \emptyset$ for $j_1, j_2 \in \mathfrak{Ch}(i)$ then one of the children of i is picked to be j_1. If the wrong decision is made, then the algorithm may decide to split the price between j_1, j_2 instead of assigning the whole upstream cost $\mathbf{p}$ to one of them. In this case the algorithm will converge to a 0 price share to the child incurring no upstream cost.

[10] The intervals remain overlapped only if the optimal split of $\mathbf{p}$ has been achieved.

[11] If the demand intervals of the children of i do not become disjoint in finite time it means that the children have the same demand. In this case the optimal price split has been achieved, and since the demand intervals of the children of i are decreasing to a point, by the construction of the algorithm the demand intervals of node i decrease to a point.

Since by Lemma 1 the optimal rate is in all demand intervals, the convergence of the demand intervals to the optimal rate follows.

Lemma 3 *Let i be a junction node upstream of receiver node j. For any $\varepsilon > 0$, if β_i, β_i' are two upstream price shares at node i such that $|\beta_i - \beta_i'| < \varepsilon$, then $|p^*_{j,\beta_i} - p^*_{j,\beta_i'}| < \varepsilon$ where p^*_{j,β_i},$p^*_{j,\beta_i'}$ are the optimal service price of node j given β_i, β_i' respectively.*

Proof Assume without loss of generality that $\beta_i \geq \beta_i'$ and $\beta_i - \beta_i' < \varepsilon$. We prove that for any receiver j' downstream of node i, if we increase the upstream price share of node i from β_i' to β_i then optimal service price of j' will also increase. That is, if p^*_{j',β_i} ($p^*_{j',\beta_i'}$) is the optimal service price of receiver j' given β_i (β_i'), then we will have $p^*_{j',\beta_i} \geq p^*_{j',\beta_i'}$.

Assume by contradiction that for some j' we have that $p^*_{j',\beta_i} < p^*_{j',\beta_i'}$, which implies that for the corresponding demands we will have $x_{j'}(p^*_{j',\beta_i}) > x_{j'}(p^*_{j',\beta_i'})$.

Pick junction node k upstream of node j' in the following way:

- If for the upstream price share β_i' at i the optimal rate demanded by j' is the rate through branch i, then set k to be node i.
- Else let k be the node upstream of j' with the property that the optimal rate on branch k given β_i' is equal to the optimal rate demanded by j' given β_i', and is strictly less than the optimal rate on the parent branch of k under β_i'.

Denote by $\mathbb{L}_{\beta_i}$ the set of links of branch k union the set of links downstream of node k for which the optimal rate given β_i is equal to the optimal rate on branch k given β_i. Denote by $\mathbb{R}_{\beta_i}$ the set of users downstream of node k for which the optimal demand given β_i equals the rate on branch k.

The contradiction is now established by the following inequalities:

$$\sum_{l \in \mathbb{L}_{\beta_i}} \lambda_l \leq \sum_{r \in \mathbb{R}_{\beta_i}} p^*_{r,\beta_i} \tag{B.1}$$

Equation B.1 follows by the algorithm construction, since the price per unit of rate on any link l is split between the users downstream of l demanding the rate of l. Furthermore,

$$\sum_{r \in \mathbb{R}_{\beta_i}} p^*_{r,\beta_i} < \sum_{r \in \mathbb{R}_{\beta_i}} p^*_{r,\beta_i'} \tag{B.2}$$

Equation B.2 is true since for any user $j'' \in \mathbb{R}_{\beta_i}$ the optimal rate of j'' given β_i is greater than or equal to the rate of j' given β_i which is strictly greater than the rate of j' given β_i'. However, since the rate of j' given β_i' is equal to the rate on branch k, then the rate demanded by j'' given β_i' is less than or equal to the rate of j' given β_i'. Finally,

$$\sum_{r \in \mathbb{R}_{\beta_i}} p^*_{r,\beta_i'} \leq \sum_{l \in \mathbb{L}_{\beta_i}} \lambda_l \tag{B.3}$$

Equation B.3 follows since no price upstream of link k is split between the users of $\mathbb{R}_{\beta_i}$, and a share of the prices of the links in $\mathbb{L}_{\beta_i}$ may be assigned to users that are not in $\mathbb{R}_{\beta_i}$.

Combining B.1,B.2 and B.3 we achieve a contradiction to the assumption that $p^*_{j',\beta_i} < p^*_{j',\beta_i'}$. This gives us that the optimal service prices of users downstream of node i are monotonically increasing in the price share upstream of node i.

By the construction of the algorithm, the sum of the service prices downstream of node i is equal to the sum of link prices downstream of node i plus the price share upstream of node i. This implies that the sum of the service prices downstream of i given β_i is within ε of the sum of the service prices downstream of i given β_i'. Since the service prices downstream of node i are monotonic in the price share upstream of node i, then for any j downstream of node i, optimal service prices of j given β_i and β_i' are within ε of each other.

We are now going to prove that for fixed link prices the service prices generated by the price splitting algorithm maximize the users' total utility.

Proof (Proof of Theorem 1)

Fix $\varepsilon > 0$. We show that in a finite number of steps the price seen by any receiver node j is within ε of the optimal price.

Let junction nodes $\{i_1, i_2, \ldots, i_n\}$ be such that: they are all upstream of node j, the parent of node i_1 is the source node, the parent of the j is i_n and the parent of i_k is node i_{k-1}.

We first apply Lemma 2 to node i_1. Then, in finite time the upstream price share at node i_2 is within $\frac{\varepsilon}{2}$ of the optimal upstream price share. By Lemma 3 the optimal service price of j given the upstream price share of node i_2 is, in finite time, within $\frac{\varepsilon}{2}$ of the optimal service price. Applying Lemmas 2 - 3 at node i_2 we determine in finite time an upstream price share of i_3, such that the optimal service price of j given this upstream price share of i_3 is within $\frac{3\varepsilon}{4}$ of the optimal service price. Proceeding in this manner along the tree we have that the in finite time the service price of receiver j is within ε of the optimal service price.

Discussion

The formal description of the algorithm and its analysis were done for the case where each junction node has exactly two children. We showed in Section 4.3 that any multicast tree can be transformed into an equivalent tree where each node has two children. Thus, the result of Theorem 1 is valid for any multicast tree.

C Proof of Theorem 2

We first establish the following result:

Lemma 4 *Let λ be a set of link prices and $m \in M$ be a multicast tree. The price splitting algorithm determines the splitting trees of each user in m in a finite number of iterations.*

Proof Let $\{j_1, j_2, \ldots, j_k\}$ be the set of root branches of the splitting trees of the users in m, excluding the branch which is connected directly to the source node. Let $\{j_i, j_i'\} = \pi_m(j)$, and define ε_i be the difference between the optimal rate on j_i' and j_i. By the definition of a splitting tree, $\varepsilon_i > 0$ for all $i \in \{1, 2, \ldots, k\}$. Define $\varepsilon = \min_{i \in \{1,2,\ldots,k\}} \varepsilon_i$. By Theorem 1 the service prices determined by each iteration of the price splitting algorithm converge to the optimal service prices. Since the utility functions are assumed to be strictly concave, Theorem 1 implies that after a finite number of iterations all the user demands generated by the price splitting algorithm will be within $\frac{\varepsilon}{2}$ of the optimal service demands. Since for any branch j on the multicast tree m the demands on j equal the largest user demand downstream of j, after a finite number of iterations all the rates on all the branches of the multicast tree will be within $\frac{\varepsilon}{2}$ of the optimal rates. This implies that for any $i \in \{1, 2, \ldots, k\}$, after a finite number of iterations of the price splitting algorithm, all the rates demanded on j_i will be smaller then the rates demanded on j_i', which implies that the splitting trees of m are determined after a finite number of iterations.

Using Lemma 4, we proceed to complete the proof of Theorem 2. Assume that for a given set of link prices λ there exists $l \in L$ for which $|z_l(\lambda)| = e > 0$. For any $m \in M$ containing l denote by l_m' the root of the smallest splitting tree of m containing l, and denote by $\mathfrak{M}$ the number of multicast trees which contain l. From Theorem 1 and Lemma 4 we have that for each $m \in M$ after a finite number of iterations the length of the demand interval at l_m' is less then $\frac{e}{\mathfrak{M}}$. Denote such an interval by $[\underline{b}_{l_m',m}, \overline{b}_{l_m',m}]$. We have the following two cases to consider:

Case 1: $z_l(\lambda) = e > 0$

First we establish the following:

Fact C1 $z_l(\lambda) > 0 \iff$ *after a finite number of iterations* $\sum_{m \in \{m' : l \in L_{m'}\}} \underline{b}_{l_m',m} > c_l$.

$(:\Longrightarrow)$ We have:

$$z_l(\lambda) = \sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda)) - c_l \tag{C.1}$$

$$> \sum_{m\in\{m':l\in L_{m'}\}} \underline{b}_{l'_m,m} - c_l \tag{C.2}$$

$$> \sum_{m\in\{m':l\in L_{m'}\}} (\max_{r\in R_{l,m}} x_r(p(r,\lambda)) - \frac{e}{\mathfrak{M}}) - c_l \tag{C.3}$$

$$= z_l(\lambda) - e = 0 \tag{C.4}$$

where equations (C.1) and (C.4) follow from the definition of the excess demand function, equation (C.2) follows from Lemma 1, and equation (C.3) follows from Lemma 1 and the fact that the length of the demand intervals considered is at most $\frac{e}{\mathfrak{M}}$.

The chain of inequalities (C.1)-(C.4) proves that given $l \in L$ if $z_l(\lambda) > 0$ then after a finite number of iterations the sum over all the multicast trees of the lower bounds of the demand intervals on the roots of the splitting trees containing l will be larger than the capacity at l. i.e.

$$\sum_{m\in\{m':l\in L_{m'}\}} \underline{b}_{l'_m,m} > c_l. \tag{C.5}$$

$(:\Longleftarrow)$ To prove the sufficiency part of Fact C1 it is sufficient to show that: if $z_l(\lambda) \leq 0$ then (C.5) is never true. Suppose that $z_l(\lambda) \leq 0$, and let $\{[\underline{b}_{l'_1,1}, \overline{b}_{l'_1,1}], [\underline{b}_{l'_2,2}, \overline{b}_{l'_2,2}], \ldots, [\underline{b}_{l'_{\mathfrak{M}},\mathfrak{M}}, \overline{b}_{l'_{\mathfrak{M}},\mathfrak{M}}]\}$ be any set of demand intervals. Note that by the algorithm construction the optimal rate demand at any link $l \in L$ of the multicast tree $m \in M$ is in all the demand intervals of l'_m, which implies that:

$$\sum_{m\in\{m':l\in L_{m'}\}} \underline{b}_{l'_m,m} \leq \sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda)) \leq c_l. \tag{C.6}$$

Thus we have established Fact C1. Using Fact C1 and the fact that the splitting trees of all users of any multicast tree are determined in a finite number of iterations (Lemma 4), we conclude that the assertion of the Theorem 2 is true when $z_l(\lambda) > 0$.

Case 2: $z_l(\lambda) = -e < 0$

First we establish the following:

Fact C2 $z_l(\lambda) < 0 \iff$ *after a finite number of iterations* $\sum_{m\in\{m':l\in L_{m'}\}} \overline{b}_{l'_m,m} < c_l$.

$(:\Longrightarrow)$ We have:

$$z_l(\lambda) = \sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda)) - c_l \tag{C.7}$$

$$< \sum_{m\in\{m':l\in L_{m'}\}} \overline{b}_{l'_m,m} - c_l \tag{C.8}$$

$$< \sum_{m\in\{m':l\in L_{m'}\}} (\max_{r\in R_{l,m}} x_r(p(r,\lambda)) + \frac{e}{\mathfrak{M}}) - c_l \tag{C.9}$$

$$= z_l(\lambda) + e = 0 \tag{C.10}$$

where equations (C.7) and (C.10) follow from the definition of the excess demand function, equation (C.8) follows from Lemma 1, and equation (C.9) follows from Lemma 1 and the fact that the length of the demand intervals considered is at most $\frac{e}{\mathfrak{M}}$.

The chain of inequalities (C.7)-(C.10) proves that given $l \in L$ if $z_l(\lambda) < 0$ then after a finite number of iterations the sum over all the multicast trees of the of the upper bounds of the demand intervals on the roots of the splitting trees containing l will be larger then the capacity at l. i.e.

$$\sum_{m\in\{m':l\in L_{m'}\}} \overline{b}_{l'_m,m} < c_l. \tag{C.11}$$

($\underline{:\Longleftarrow}$) It is sufficient to show that: if $z_l(\lambda) \geq 0$ then (C.11) is never true. Suppose that $z_l(\lambda) \geq 0$, and let $\{[\underline{b}_{l'_1,1}, \overline{b}_{l'_1,1}], [\underline{b}_{l'_2,2}, \overline{b}_{l'_2,2}], \ldots, [\underline{b}_{l'_{\mathfrak{M}},\mathfrak{M}}, \overline{b}_{l'_{\mathfrak{M}},\mathfrak{M}}]\}$ be any set of demand intervals. Note that by the algorithm construction the optimal rate demand at any link $l \in L$ of the multicast tree $m \in M$ is in all the demand intervals of l'_m, which implies that:

$$\sum_{m \in \{m' : l \in L_{m'}\}} \overline{b}_{l'_m,m} \geq \sum_{m \in M} \max_{r \in R_{l,m}} x_r(p(r,\lambda)) \geq c_l. \tag{C.12}$$

Thus we have established Fact C2. Using Fact C2 and the fact that the splitting trees of all users of any multicast tree are determined in a finite number of iterations (Lemma 4), we conclude that the assertion of the Theorem 2 is true when $z_l(\lambda) < 0$.

□

In the above proof we needed to determine the sign of the excess demand function on a link l from the demand intervals at the root of the smallest splitting tree containing l. The reason for this is that if l is not a root of a splitting tree the demand intervals at l are not nested, they vary in size over time, and may not contain the optimal rate of link l. The root of the smallest splitting tree containing l, call it l', has the same optimal rate as l which is contained in all of its demand intervals (direct consequence of Lemma 1 since the upstream price at l' is equal to the price of the rate on l', and it is constant). Also the demand intervals at l' are nested and they converge to the optimal rate (Lemma 2). Therefore, the demand intervals at l' can be used to determine the sign of the excess demand function at l.

Acknowledgement: This research was supported in part by NSF Grant ECS-9979347 and by ONR Grant N00014-03-1-0232.

References

1. Y. Bartal, J. Byers, and D. Raz. Global optimization using local information with applications to flow control. In *Proceedings of the* 38^{th} *Ann. IEEE Symp. on Fundations of Computer Science (FOCS)*, Miami, FL, October 1997.
2. M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming Theory and Algorithms.* John Wiley, New York, 1993.
3. D.P. Bertsekas and R. Gallagher. *Data Networks.* Prentice Hall, second edition, 1992.
4. T. Bially, B. Gold, and S. Seneff. A technique for adaptive voice flow control in integrated packet networks. *IEEE Transactions on Communications*, 28(3):325–333, March 1980.
5. S. Deb and R. Srikant. Congestion control for fair resource allocation in networks with multicast flows. *IEEE/ACM Transactions on Networking*, 12(2):261–273, 2004.
6. M. J. Donahoo, K. Calvert, and E. W. Zegura. Center selection and migration for wide-area multicast routing. *Journal of High Speed Networks*, 6(2), 1997.
7. M. J. Donahoo and E. W. Zegura. Core migration for dynamic multicast routing. In *In Proceedings of ICCCN*, Washington, DC, 1996.
8. N.G. Duffield, J. Horowitz, D. Towsley, W. Wei, and T. Friedman. Multicast-based loss inference with missing data. *IEEE Journal on Selected Areas in Communications*, 20(4):700–713, May 2002.
9. E.J. Friedman. Optimization based characterizations of cost sharing methods. Departamental Working Paper at Rutgers University, Department of Economics, June 1999.
10. E. Graves, R. Srikant, and D. Towsley. Decentralized computation of weighted max-min fair bandwidth allocation in networks with multicast flows. In *Proceedings Tyrrhenian International Workshop on Digital Communications (IWDC)*, Taormina, Italy, 2001.
11. R. Gupta and J. Walrand. Average bandwidth and delay for reliable multicast. In *Proceedings IFIP Performance*, Istanbul, Turkey, October 1999.
12. S. Herzog, S. Shenker, and D. Estrin. Sharing the "cost" of multicast trees: An axiomatic analysis. *IEEE/ACM Transactions on Networking*, 5(6), December 1997.
13. K. Kar, S. Sarkar, and L. Tassiulas. Optimization based rate control for multirate multicast sessions. In *Proceedings of INFOCOM*, Alaska, 2001.
14. K. Kar, S. Sarkar, and L. Tassiulas. A simple rate control algorithm for maximizing total user utility. In *Proceedings of INFOCOM*, Alaska, 2001.
15. K. Kar, S. Sarkar, and Leandros Tassiulas. A scalable low overhead rate control algorithm for multirate multicast sessions. *IEEE Journal of Selected areas in Communication*, 20(8):1541–1557, October 2002. Special issue in Network Support for Multicast Communications.

16. S. Kasera, G. Hjalmtysson, D. Towsley, and J. Kurose. Scalable reliable multicast using multiple multicast channels. *IEEE/ACM Transactions on Networking*, June 2000.
17. F.P. Kelly. On tariffs, policing and admission control for multiservice networks. *Operations Research Letters*, 15:1–9, 1994.
18. F.P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunication*, 8(1):33–37, 1997.
19. F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Operational Research Society*, 49:237–252, 1998.
20. F. Kishino, K. Manabe, Y. Hayashi, and H.Yasuda. Variable bit-rate coding of video signals for ATM networks. *IEEE Journal on Selected Areas In Communications*, 7(5), June 1989.
21. S. Kunniyur and R. Srikant. End to end congestion control schemes:Utility functions, random losses and ECN marks. In *Proceedings of INFOCOM*, March 2000.
22. R. La and V. Anantharam. Charge-sensitive TCP and rate control on the internet. In *Proceedings of INFOCOM*, Tel Aviv, Israel, March 2000.
23. X. Li, S. Paul, and M. H. Ammar. Layered video multicast with retransmission (LVMR):Evaluation of hierarchical rate control. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, 1998.
24. S. Low and D.E. Lapsley. Optimization flow control I: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6), December 1999.
25. A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, New York, 1995.
26. L. Massoulie and J. Roberts. Bandwidth sharing: Objectives and algorithms. In *Proceedings of IEEE INFOCOM 1999*, New York, NY, March 1999.
27. S. McCanne, V. Jacobson, and M. Vetterli. Receiver driven layered multicast. In *Proceedings of ACM SIGCOMM*, Stanford, CA, 1996.
28. W. B. Park, H. L. Owen, and E. W. Zegura. Sonet/SDH multicast routing algorithms in symmetrical three-stage networks. In *ICC*, Seattle, WA, June 1995.
29. D. Rubenstein, J. Kurose, and D. Towsley. The impact of multicast layering on network fairness. In *Proceedings of ACM SIGCOMM*, Cambridge, MA, 1999.
30. S. Sarkar and L. Tassiulas. Fair allocation of resources in multirate multicast trees. In *Proceedings of Globecom*, 1999.
31. S. Sarkar and L. Tassiulas. Distributed algorithms for computation of fair rates in multirate multicast trees. In *Proceedings of IEEE INFOCOM*, Tel Aviv, Israel, 2000.
32. S. Sarkar and L. Tassiulas. Fair allocation of disrete bandwidth layers in multicast networks. In *Proceedings of INFOCOM*, Tel Aviv, Israel, 2000.
33. S. Sarkar and L. Tassiulas. Layered bandwidth allocation for multicasting of hierarchically encoded sources. 2000.
34. S. Sarkar and L. Tassiulas. Back pressure based multicast scheduling for fair bandwidth allocation. In *Proceedings of INFOCOM*, Alaska, 2001.
35. S. Sarkar and L. Tassiulas. Fair allocation of utilities in multirate multicast networks: A framework for unifying diverse fairness objective. *IEEE Transactions on Automated Control*, 47(6):931–944, 2002.
36. J. K. Shapiro, J. Kurose, D. Towsley, and S. Zabele. Topology discovery service for router-assisted multicast transport. In *Proceedings of OpenArch*, 2002.
37. J. K. Shapiro, D. Towsley, and J. Kurose. Optimization-based congestion control for multicast communications. In *Proceedings of IEEE INFOCOM*, Tel Aviv, Israel, 2000.
38. C.P. Simon and L. Blume. *Mathematics for Economists*. W. W. Norton, New York, 1994.
39. T.M. Stoenescu and D. Teneketzis. A pricing methodology for resource allocation and routing in integrated-service networks with quality of service requirements. *Mathematical Methods of Operations Research (MMOR)*, 56(2), 2002.
40. P. Thomas, D. Tenektezis, and J. K. MacKie-Mason. A market - based approach to optimal resource allocation in integrated - services connection- oriented networks. *Operations Research*, 50(5), July-August 2002.
41. T. Turletti and J.C. Bolot. Issues with multicast video distribution in heterogeneous packet networks. Packet Video Workshop, 1994.
42. H. Y. Tzeng and Siu K, Y. On max-min fair congestion for multicast ABR service in ATM. *IEEE Journal on Selected Areas in Communication*, 15(3), 1997.
43. E. W. Zegura. Routing algorithms in multicast switching topologies. In *Proceedings of Allerton Conference on Communication, Control and Computing*, Monticello, IL, September 1993.