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

Multirate Multicast Service Provisioning II: A Tâtonnement Process for Rate Allocation

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 multirate multicast different users in the same multicast group can receive services at different rates depending on their own requirements and the congestion level of the network. In this two-part paper we present a general framework for addressing the optimal rate 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. In Part II we present a market based mechanism and an adjustment process that have the following features. They satisfy the informational constraints imposed by the nature of multirate multicast; and when they are combined with the results of Part I they result in an optimal solution of the corresponding centralized multirate multicast problem.

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 [2], [31] and over ATM networks [10]. Internet protocols for adding and dropping layers for hierarchical encoding type of transmissions are presented in [11] and [14].

In this two-part paper we present a market based mechanism (described by a Tâtonnement process) for multirate multicast. We have already compared our approach with other existing approaches to multicast service provisioning (e.g., [3, 5, 8, 9, 19–22, 24, 25, 27, 32]) in Part I of this paper. In Part I we also pointed out that the Tâtonnement process we present can be viewed as a hierarchical process consisting of two layers: the lower layer and the upper layer (cf. Section 3.2.1). We addressed the problem of the lower layer in Part I of the paper. In Part II we present a market based mechanism and describe an adjustment process that have the following features. They satisfy the informational constraints imposed by the nature of multirate multicast; and when they are combined with the results of Part I, they result in an optimal solution to the corresponding centralized multirate multicast problem.

Correspondence to: tudor@caltech.edu

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 describe and analyze a competitive market economy which leads to a decentralized rate allocation that achieves a solution of the centralized multi-rate multicast problem. Numerical results on the convergence of the algorithms are given in Section 4. We discuss and critique our results in Section 5, and we conclude the paper in Section 6.

2 The Multicast Problem

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

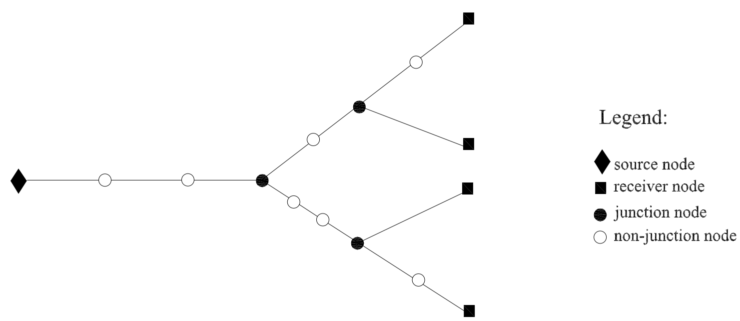

Fig. 1 A multicast tree.

We now present some terminology used for the multicast groups that is similar to terminology developed in [8, 9]. 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 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 *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. [3, 8, 9], with notable exceptions [22, 23]. 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 can not 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 [28, p.823] guarantees the existence of a solution of **Max 1**.

3 A Market Based Realization of the Solution of Problem Max 1

In this section we present a market based mechanism which achieves a solution to Problem **Max 1**, and satisfies the informational constraints imposed by the nature of the network multicast problem (these informational constraints are described at the beginning of Section 3.1).

We proceed as follows: We first describe a competitive market economy consisting of two types of agents: network and users. Then, within the context of this market we specify a procedure used by the network which leads to an allocation that achieves a solution to Problem **Max 1**.

3.1 Description of the Market

The market economy adopted in this section is composed of two types of agents: network (or network manager) and users. The network communicates directly with each user, and the users do not communicate with one another. The messages exchanged by the market agents are service prices and service demands. The network manager is assumed to know the topology of the network and the resources available to the network, but has no a priori information about the number of users that will request services and the preferences (utility function) of each user. The users are assumed to know their own preferences (utility function) but have no information about the number and preferences of other users requesting services, or the topology and the resources available to the network. Further, as mentioned in Section 1, the users are unaware of the method of delivery of services (i.e. they do not know whether service provisioning is unicast or multicast).

The assumption that the network manager has complete knowledge of the network topology and resources is not an unrealistic one. For example, a corporate intranet or VPN (virtual private network) may have a single provider of resources and services, who is likely to have such knowledge about the network, and who will assume the roll of network management in collecting aggregate excess demand on links and adjusting link prices. In particular, some resource/service providers use very sophisticated network management tools to monitor in real time the proper functions of a network (e.g., events such as congestion, fault, server up and downs), and to issue appropriate response/commands. Such monitoring requires complete knowledge of the network (e.g., topology, resources, router/link capacities), as well as separate network management protocols to pass information to and form the management site. These tools can easily be used to acquire information on aggregate excess demands and to adjust link prices.

For conceptual clarity we decompose the network manager into two distinct entities: service provider and auctioneer. The market features and the relation between the market agents are as follows: The resource traded at each link is the available communication rate (i.e. bandwidth or capacity). The rate at each link is available to the service provider as raw material. The rate price at link $l \in L$ will be denoted by λ_l. The service provider sets up services and the corresponding prices for each unit of these services and then sells these services to the users. Based on the service prices announced by the service provider the users demand a certain amount of service from the network in order to maximize their utility functions. Based on the user demands the auctioneer updates the price per unit of rate on each link.

We make the assumption that the service provider and users are price takers. They act as if their behavior has no effect on the equilibrium prices reached by the market allocation process. This assumption is justified by the fact that the users are unaware of the type of service received and they do not know the number of users requesting service from the network. The price taking assumption and the fact that we try to maximize the users utilities imply that: (i) the service provider will not attempt to make a profit; and (ii) the service prices are directly derived from resource prices. A further discussion of the price taking assumption appears in [30, Section 5].

3.1.1 Service provider The service provider receives from the auctioneer a rate price λ_l for each link l of the network. Based on these link prices the task of the service provider is to compute for each user $r \in R$ the price per unit of service $p(r, \lambda)$. A major challenge in solving multirate multicast problems through pricing is the determination of the set of users' price $\{p(r, \lambda)\}$ per unit of service from the set of link prices λ. This issue was addressed in Part I of this two-part paper.

In Part I we presented a distributed algorithm which, for a fixed set of link prices λ, computes price shares $\gamma_m = \{\gamma_{r,l,m} | r \in R_m, l \in L_m\}$ and service prices $p(r, \gamma(\lambda))$ that satisfy the following.

Property 1

1.
$$\lambda_l = \sum_{r \in R_m} \gamma_{r,l,m}, \quad \forall\, l \in L_m \;; \tag{3.1}$$

2.
$$p(r, \lambda) \triangleq p(r, \gamma_m(\lambda)) = \sum_{l \in L} \gamma_{r,l,m}, \quad \forall\, r \in R_m \;; \tag{3.2}$$

3. Let $x_r(p(r, \lambda)) \triangleq \text{argmax}_{x>0}\{U_r(x) - p(r, \lambda) \times x\}$ be the demand requested by user r given the price per unit of service $p(r, \lambda)$. Then,

$$\sum_{r \in R_m} x_r(p(r, \lambda)) \times p(r, \lambda) = \sum_{l \in L_m} \lambda_l \times \max_{r \in R_{l,m}} x_r(p(r, \lambda)) \tag{3.3}$$

and

$$\sum_{r \in R_m} U_r(x_r(p(r, \lambda))) \geq \sum_{r \in R_m} U_r(x_r(p_r*)) \tag{3.4}$$

for all p_r* satisfying

$$\sum_{r \in R_m} x_r(p_r*) \times p_r* \geq \sum_{l \in L_m} \lambda_l \times \max_{r \in R_{l,m}} x_r(p_r*). \tag{3.5}$$

3.1.2 Users Users are price takers and request service from the service providers. For each user $r \in R_m$ of the multicast tree $m \in M$ the service provider announces a service price $p(r, \lambda)$. Based on its service price, each user determines its desired service rate by solving:

$$x_r(p(r,\lambda)) \triangleq \underset{x}{\operatorname{argmax}}\{U_r(x) - p(r,\lambda) \times x\} \tag{3.6}$$

3.1.3 Auctioneer The role of the auctioneer is to regulate the prices of the resources, based on the aggregate excess demand vector $z(\lambda)$,

$$z_l(\lambda) \triangleq \sum_{m \in M} \max_{r \in R_{l,m}} x_r(p(r,\lambda)) - c_l \tag{3.7}$$

at every link $l \in L$.

3.2 The Market Mechanism

3.2.1 The Mechanism for General Concave Utility Functions We present a market mechanism, described by an algorithm, called Algorithm (⊛), that describes how the market works. The algorithm proceeds iteratively as follows:

Step 1: The multicast trees are fixed.
Step 2: The auctioneer announces prices λ per unit of rate at each link of the network.
Step 3: The service provider receives the link prices λ announced by the auctioneer. Given the link prices, the service provider communicates with the users via an iterative process in order to determine the optimal service price. During the iterative process the service provider and the users exchange prices per unit of service p and service demands $x(p)$, with $x(p)$ satisfying (3.6). The iterative process used in this paper is described by the algorithm presented in Part I of the paper.

  Step 3.1: During the iterative process between the service provider and users, the auctioneer checks if the sign of the excess demand $z(\lambda)$ on any link is positive. If this is the case then the auctioneer interrupts the iterative process and proceeds to Step 4. If $z(\lambda) \leq 0$ at all links, the process terminates.

Step 4: The auctioneer updates the link prices λ and announces them to the service provider. The process loops back to Step 3.

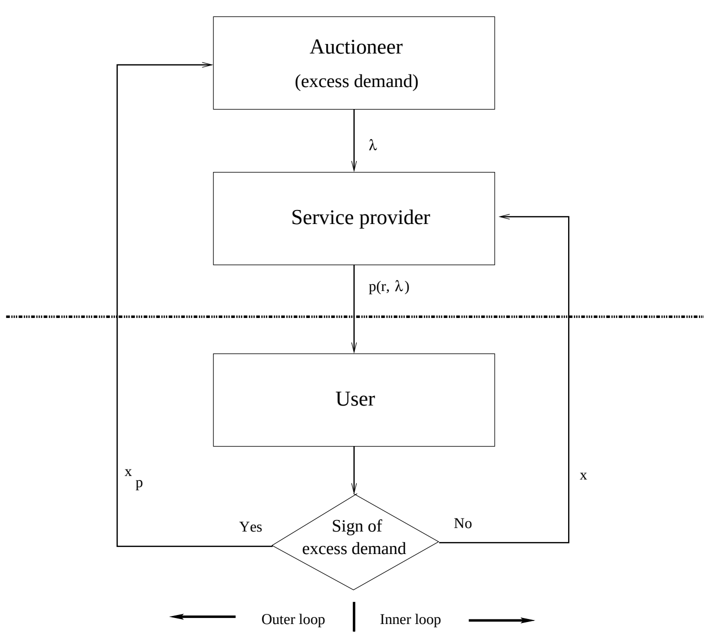

Fig. 2 Market mechanism.

The above steps are pictorially shown in Figure 2. The figure illustrates the fact that the algorithm contains two loops: an outer loop and an inner loop. Thus the above market based mechanism can be viewed as a two-level hierarchical process. The inner loop is the lower level of the hierarchy, and the outer loop is the upper level of the hierarchy. The inner loop describes the iterative process used by the service provider to determine user service prices (hence user demands) for fixed link prices set by the auctioneer. The outer loop determines the iterative process used by the auctioneer to determine link prices based on excess demand. The iterative process of the inner loop is guided by the results developed in Part I of the paper. The iterative process of the outer loop is described by Scarf's Algorithm [26]. A detailed description of this algorithm is presented in [29, Appendices A,B]. It may be possible to use algorithms other then Scarf's at the outer loop, however, to prove convergence of such algorithms we may need to impose additional constraints on the users' utility functions (e.g. second order differentiability of the utility functions). In Section 3.3 we show this mechanism eventually leads to a resource allocation that achieves a solution to Problem **Max 1**. Consequently, the algorithm described in this section "approximates" in a finite number of steps an optimal solution to the original resource allocation problem (Problem **Max 1**) and satisfies the informational constraints imposed by the decentralization of information at the network.

Remark 1 As noted before, the inner loop of the pricing mechanism described above uses the algorithm presented in Part I of this paper. If the set of prices $\{\lambda\}$ per unit of rate on the links is not optimal then there exist links for which the sign of the excess demand function is determined in finite time (i.e., the execution of the inner loop will terminate in finite time). This in turn ensures that each stage of the outer loop for which the prices per unit of rate are not optimal is finite. When the price per unit of rate on each link is optimal then Algorithm ($\circledast$) will result in optimal prices per unit of service for each user, and this, in turn, will result in an optimal resource allocation (i.e., an optimal solution to Problem **Max 1**.

3.2.2 The Mechanism for Parameterized Concave Utility Functions The case where the users' utility functions come from a class which is parameterized by a finite number of parameters deserves special attention. In this case the service provider can determine in the first iteration of the outer loop the users' utility functions. This can be accomplished in a finite number of iterations of the inner loop. After the first iteration of the outer loop the rate allocation problem becomes a centralized decision problem (a nonlinear mathematical programming problem) which can be solved by standard techniques [1, 12, 17, 18]. In Appendix A we present examples of parameterized families of utility functions. For each family we determine the number of iterations required in the execution of the inner loop in order to completely specify the users' utility functions.

3.3 The Main Result

The main result of this paper is summarized by the following theorem:

Theorem 1 *The market mechanism described in Section 3.2 along with the algorithm developed in Part I of this paper converges to an optimal solution of Problem **Max 1**.*

We prove the main result in the rest of the paper.

3.4 Analysis of the Market and Proof of the Main Result

The proof of Theorem 1 proceeds in several steps. First we present preliminary technical results that are related with the behavior of the iterative process describing the outer loop of the algorithm. Then, we use these results to conclude the proof of Theorem 1

We proceed with the details of the analysis. Scarf's Algorithm (that describes the outer loop) works in the price simplex. Therefore, we start by defining the following $|L| + 1$ dimensional simplex.

$$S \triangleq \{q \in \mathbb{R}_+^{|L|+1} : \sum_{m=0}^{|L|} q_m = 1\} \tag{3.8}$$

where $|L|$ is the cardinality of L. For each $q \in S$ with $q_0 > 0$ we define the *price vector* $\lambda(q)$:

$$\lambda(q) = \{\lambda_l(q)\}_{l\in L} \triangleq \left\{ \frac{q_1}{q_0}, \frac{q_2}{q_0}, \ldots, \frac{q_{|L|}}{q_0} \right\} \tag{3.9}$$

The goal is to find an optimal price for each of the resources in the network, that is, a price vector λ which leads to a solution for **Max 1**. Note that the mapping defined in equation (3.9) is a continuous bijection, so it is enough to find an appropriate $q \in S$ such that $\lambda(q)$ is a solution of **Max 1**.

To achieve this goal we need to introduce the following concepts:

(i) The subsets $P^D \triangleq \{q \in S : q = (\frac{n_0}{D}, \frac{n_1}{D}, \ldots, \frac{n_{|L|}}{D}), n_i \in \mathbb{N}\}$, where $D \in \mathbb{N}$ represents "how close" our solution is to the solution of **Max 1**. (In Sections 5 we discuss what we exactly mean by "how close", and its implication to the problem.)

(ii) The notions of a *side* of the simplex and of a *primitive set* that are defined as follows:

Definition 1 *A side of S, denoted by $\mathfrak{s}^m$, is defined by $\mathfrak{s}^m \triangleq \{q \in S : q_m = 0\}$, for $m \in \{0, 1, \ldots, |L|\}$.*

Definition 2 *Let $\mathbb{S} \triangleq \{\mathfrak{s}^0, \ldots, \mathfrak{s}^{|L|}\}$ be the set of sides of S. Define $S^D \triangleq P^D \cup \mathbb{S}$.*

Definition 3 *$Q^D = (\mathfrak{s}^{i_1}, \ldots, \mathfrak{s}^{i_n}, q^{j_0}, \ldots, q^{j_{|L|-n}})$, in S^D is called a* primitive set *if $q^{j_0}, \ldots, q^{j_{|L|-n}} \in P^D$, $\mathfrak{s}^{i_1}, \ldots, \mathfrak{s}^{i_n} \in \mathbb{S}$, and no $q \in P^D$ is interior to the simplex generated by the vectors of Q^D, i.e. $\{x \in S : x_{i_1}, \ldots, x_{i_n} \geq 0, x_m \geq min\{q_m^{j_0}, \ldots, q_m^{j_{|L|-n}}\}, \forall m \neq \{i_1, \ldots, i_n\}\}$.*

(iii) The following subsets of S:

$$\mathfrak{C}_0 \triangleq \left\{ q \in S : q_0 = 0 \ \text{ or } \ z_l(\lambda(q)) \leq 0, \forall\, l \in L \right\} , \tag{3.10}$$

$$\mathfrak{C}_l \triangleq \left\{ q \in S : q_0 > 0 \text{ and } \{q_l = 0 \ \text{ or } \ z_l(\lambda(q)) \geq 0\}, l \in L \right\} , \tag{3.11}$$

where,

$$z_l(\lambda) = \sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda)) - c_l \ , \tag{3.12}$$

with $l \in L$, and $x_r(p(r,\lambda))$ being determined as in Section 3.1.

(iv) The concept of a *labeling function* that is defined as follows:

Definition 4 *A labeling function is a function with domain S and range $\{0, 1, \ldots, |L|\}$.*

We define the labeling function ϑ as follows:

$$\vartheta(q) = \begin{cases} i \text{ if } q \in \mathfrak{s}^i \\ 0 \text{ if } q \in \mathfrak{C}_0 \\ j \text{ where } j = min\{l : q \in \mathfrak{C}_l\} \end{cases} \tag{3.13}$$

A key result in our analysis is the following:

Lemma 1 *Starting with S^D, if we give $\mathfrak{s}^i \in \mathbb{S}$ label i for every $i \in \{0, \ldots, |L|\}$ and every $q \in P^D$ a label from $\{0, \ldots, |L|\}$, then exists a primitive set in S^D such that its vectors have distinct labels.*

Proof For a proof of this lemma see [29, Appendix C].

Lemma 1 can be used to prove the following result that is crucial in the proof of the Theorem 1.

Lemma 2 *Let $\{D_i\}_{i\in\mathbb{N}}$ be a sequence such that $\forall i$, $D_i \in \mathbb{N}$ and $D_i \to \infty$ as $i \to \infty$. Let for every i, W^{D_i} denote the primitive set with distinct labels of S^D described by Lemma 1. Then there exists a subsequence $\{D'_j, j \in \mathbb{N}\}$ such that for any $q_{D'_j} \in W^{D'_j}$, $q_{D'_j}$ converges to q, where $\{x_r(p(r,\lambda(q)))\}_{r\in R}$ solves Problem **Max 1**.*

Proof First we note that given any $D \in \mathbb{N}$, the function ϑ satisfies the conditions for the labeling of the vectors S^D described by Lemma 1. This implies that given any $D \in \mathbb{N}$, there is a primitive set W^D with all the vectors that generate it having a distinct labels.

Define

$$\mathfrak{C} \triangleq \mathfrak{C}_0 \bigcap \left(\bigcap_{l\in L} \mathfrak{C}_l \right) .$$

Proposition 1 *The allocation $x_r(p(r,\lambda))$ and the aggregate excess demand $z(\lambda)$ are continuous functions of λ.*

Proof See Appendix B

Proposition 2 *For every $q \in \mathfrak{C}$, $\{x_r(p(r,\lambda(q)))\}_{r\in R}$ solves **Max 1**.*

Proof See Appendix C.

Proposition 3 $\mathfrak{C} \neq \emptyset$.

Proof Denote the element with label i of the primitive set W^D, by $q_{D,i}$. Since S is compact, for every $i \in \{0,1,\ldots,|L|\}$, the sequence $\{q_{D,i}\}_D$ has a cluster point. As $D \to \infty$ the distance between the vertices of W^D goes to 0, so $\|q_{D,i} - q_{D,j}\| \xrightarrow[D\to\infty]{} 0$ for any $i,j \in \{0,1,\ldots,|L|\}$. This means that for any $i \in \{0,1,\ldots,|L|\}$ the sequences $\{q_{D,i}\}_D$ have identical cluster points. Pick any such cluster point and denote it by q. Since $z_l(\lambda)$ is continuous in λ (Proposition 1), and $\lambda(q)$ is continuous in q, this implies that $\lambda(q) \in \mathfrak{C}$.

Proposition 2 together with the proof of Proposition 3 conclude the proof of Lemma 2.

Proof Theorem 1 The assertion of the theorem is a direct consequence of Lemma 2.

4 Numerical results

In this section we present numerical results from using the algorithms of the inner and outer loops. In particular, in Section 4.1 we present results for the price splitting algorithm described in Part I of this paper, for fixed sets of link prices. In Section 4.2 we provide an example combining both the inner and outer loops using Scarf's and Eaves' (K1) algorithms. Eaves' (K1) algorithm is a variation of Scarf's algorithm. The convergence of the inner loop using the price splitting algorithm and the outer loop using Scarf's algorithm were established in Part I and Section 3.4 respectively. We do not have any analytical results on the rate of the convergence of these algorithms. With the results presented in this section we illustrate features of the proposed algorithms within the context of a small set of examples.

We consider a network formed by one multicast tree as shown in Figure 3 where the link capacities are $c = \{c_1, c_2, \ldots, c_{11}\} = \{100, 100, 110, 110, 20, 80, 100, 100, 100, 100, 110\}$ and where the users' utility functions are of the form

$$u_i(x_i) = a_i \log(x_i + 1) \tag{4.1}$$

with $a = \{a_1, a_2, a_3, a_4, a_5, a_6\} = \{10, 20, 31, 25, 15, 45\}$.

4.1 Inner loop using the Price Splitting Algorithm

For the inner loop we considered two arbitrarily chosen sets of link prices $\lambda \triangleq (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6, \lambda_7, \lambda_8, \lambda_9, \lambda_{10}, \lambda_{11})$, given in Table 1.

	Link Prices
Case 1	(1 0.1 0.3 0.7 0.2 1 2 1 0.6 1 0.3)
Case 2	(1 1 1 1 1 1 1 1 1 1 1)

Table 1 Data table for the inner loop.

In Figures 4 and 5 we present the numerical results for the above two cases, by plotting the service prices for each user at each iteration of the algorithm. We note that in both cases the price splitting algorithm converges quickly (in number of iterations) to the optimal service prices corresponding to the (fixed) link prices.

The numerical results for the inner loop algorithm have been conducted on a Pentium III machine. For most of the examples considered for the tree in Figure 3, the optimal service prices were determined in less than one second. Using the same tree as in Figure 3 for examples where the prices on most of the links were shared among downstream users, the algorithm took up to four seconds to generate service prices equal (up to 5 significant digits) to the optimal service prices.

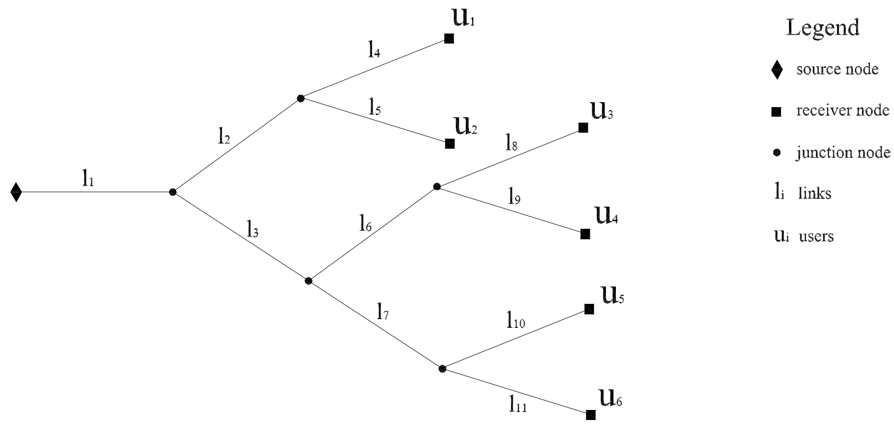

Fig. 3 A multicast tree.

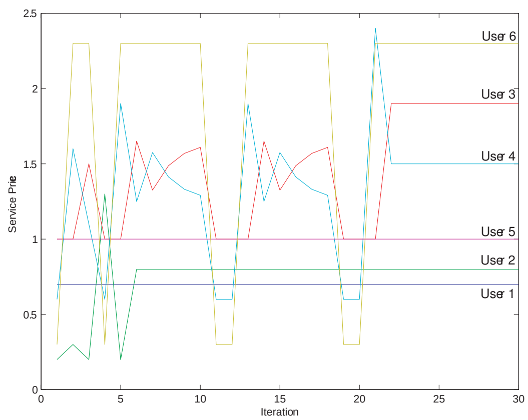

Fig. 4 Price Splitting Algorithm for Case 1.

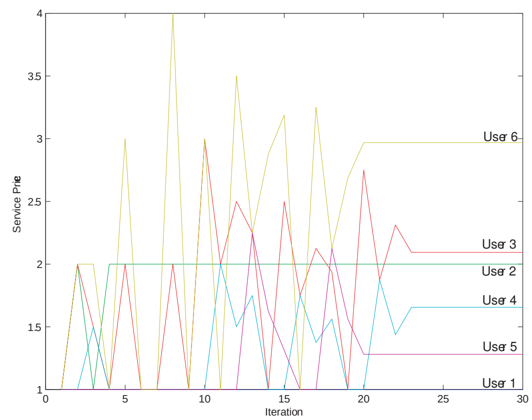

Fig. 5 Price Splitting Algorithm for Case 2.

4.2 Outer loop algorithms

In this subsection we show numerical results resulting from the combination of the inner and outer loops. As in Section 4.1, the algorithm described in Part I of this two-part paper is used for the inner loop, while for the outer loop we use both Scarf's algorithm and Eaves' (K1) algorithm.

Figures 6 and 7 show the results from Scarf's algorithm with $D = 50$ and $D = 200$, respectively. In both figures we present the prices on the links at each iteration of the algorithm. In particular, only the prices on links $1, 5$ and 6 have been displayed since the other prices remained zero. As expected, for $D = 200$ Scarf's algorithm takes more iterations to converge than for $D = 50$, but gives a better resolution.

Table 2 compares the optimal service prices to those resulting from the market mechanism used in this paper (Figures 6 and 7). We note that as the value of D increases the link prices determined by Scarf's algorithm generate service prices which approach the optimum. We know from the theoretical analysis that the optimal service prices resulting from the market mechanism are arbitrarily close to the optimal service prices as $D \to \infty$.

From the computational point of view, Scarf's algorithm has two major characteristics: 1) It requires that the algorithm be initiated at a vertex of the unit simplex; and 2) If an answer is obtained with a fixed grid whose accuracy is inadequate for the problem at hand, the algorithm must be restarted with a finer grid, and the results of the previous calculations are discarded completely. The algorithms introduced by Merrill [15], van der Laan and Talman [33, 34], and Eaves [4] permit the computation to be initiated at an arbitrary point in the simplex and allow a continual refinement of the grid. They yield a vast improvement in computational speed over the earlier algorithms of Scarf that require a fixed simplex decomposition, and are used in virtually all practical applications of fixed point methods.

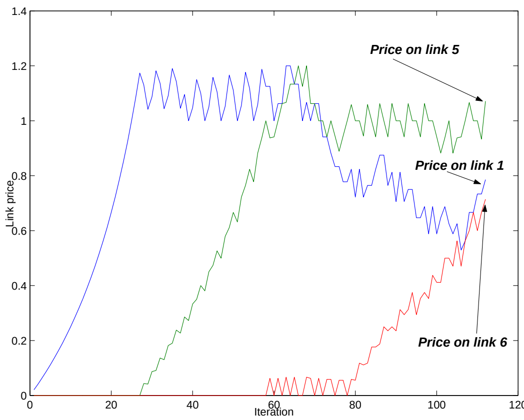

Fig. 6 Scarf's algorithm for D = 50.

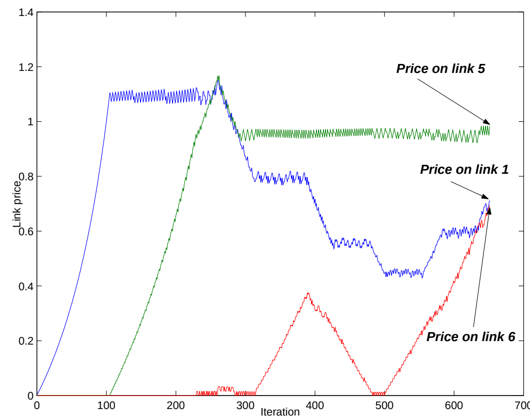

Fig. 7 Scarf's algorithm for D = 200.

		User 1	User 2	User 3	User 4	User 5	User 6
Scarf's	Price	0.1122	1.0714	0.3954	0.3189	0.1684	0.5051
D = 50	Demand	88.093	17.667	77.398	77.398	88.093	88.093
Scarf's	Price	0.0172	0.9831	0.3847	0.3102	0.1526	0.4577
D = 200	Demand	97.3283	19.3438	79.5871	79.5871	97.3283	97.3283
Optimal	Price	0.0990	0.9523	0.3827	0.3086	0.1485	0.4455
	Demand	100	20	80	80	100	100

Table 2 The service prices and user demands generated from the results of Figures 6 and 7.

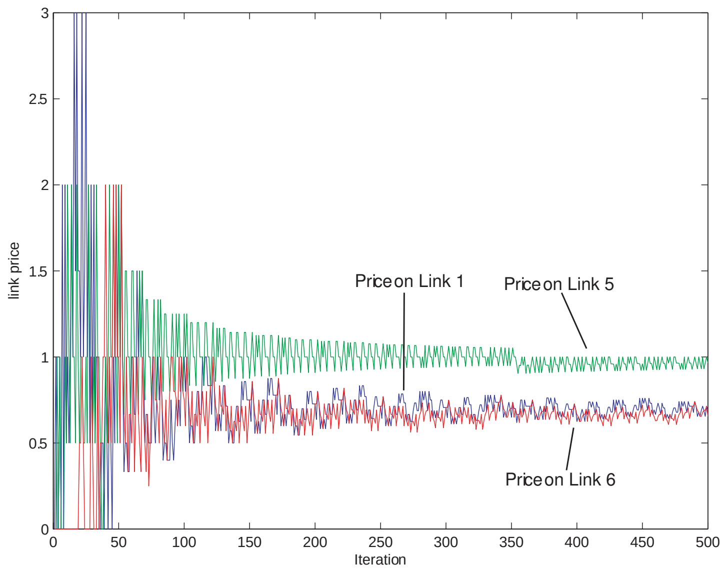

Fig. 8 The simulation result using Eaves (K1) algorithm.

For comparison purposes, in Figure 8 we present the simulation result for the same problem as above with the outer loop being implemented by Eaves' (K1) algorithm [4] rather than Scarf's algorithm. We note that after 80 iterations the algorithm achieves link prices which are close to those achieved by Scarf's algorithm.

The numerical results of the outer loop have been conducted on a Pentium III machine. For the examples presented, Scarf's algorithm (D=50), Scarf's Algorithm (D=200), and Eaves K1 algorithm (first 82 iterations) took approximately one minute, three minutes and 1.5 minutes respectively.

5 Critique, Discussion and Reflection

We have presented an approach for optimal admission and resource allocation control in multi-rate multicast. This approach has the following features.

(1) It provides a framework for solving decentralized constrained resource allocation problems that are more general than the problem invested in this paper.
(2) Resource allocations are based on the solution of a constraint static optimization problem, namely **Max 1**. The solution of **Max 1** is realized by a market mechanism that is hierarchical and satisfies the informational constraints of the network resource allocation problem.
(3) There is no cost associated with the supply of network resources.

We now discuss and critique each one of the above features:
(1) We believe that this work provides a framework for developing market methods for solving decentralized constrained resource allocation problems that are more general than the problem investigated here. As an example consider the following situation. Services are again provided along fixed multicast trees, but there are additional Quality of Service (QoS) requirements expressed by the end-to-end delay, and end-to-end percentage of packet loss etc. The objective function and the informational constraints remain the same as in Problem **Max 1** and Section 3.1. The Market mechanism proposed in this paper is ideally suited to handle end-to-end QoS requirements. This was already demonstrated in the unicast problem [30] as well as in the unicast routing problem [29]. The service provider that determines the optimal price sharing along each link of a multicast tree (for fixed link prices) can also ensure that the services provided satisfy their QoS requirements.
(2) Problem **Max 1** is a static constrained optimization problem. Its solution can be interpreted as the set of "equilibrium allocations". Thus, even though the market mechanism described in this paper is an iterative process, its outcome is a static equilibrium solution and it can not handle dynamic user arrivals and departures. The iterative nature of the market mechanism is necessitated by the fact that the overall network system (network management and users) is an informationally decentralized system. To achieve an optimal solution of the corresponding static centralized problem the network management and the users must exchange information/messages with one another. Such a message exchange process must possess the "privacy preserving" property; that is, the network manager's and each user's messages at each stage of the iterative process must be based only on their private information and the information they have received from previous communications ([6, 7, 16]). Furthermore, the message generating functions must satisfy certain "regularity conditions" ([6, 7, 16]) that guarantee that the outcome of the resource allocation process is "robust" with respect to errors (due for example to approximations) that may occur in the message exchange process. This information exchange allows the network management to "learn" about the users' preferences (utilities) and eventually maximize the network's utility to its users. "Learning" requires an iterative process of information exchange and such an iterative process is described by our market mechanism. In the case where the users' utility functions are general concave functions the network manage does not "learn" perfectly the users' utility functions, yet the market mechanism converges to an optimal solution of the centralized problem **Max 1**. When the users' utility functions came from a class which is parameterized by a finite number of parameters the network manager "learns" perfectly (in the first iteration of the outer loop) the users' utility functions.
(3) We have assumed that there is no cost in supplying network resources (such as bandwidth and buffers) to the market. We can incorporate the cost of supplying network resources into our model by subtracting if from the objective function of the optimization problem **Max 1**. We believe that the problem arising in this situation will have the same qualitative properties as **Max 1**, and the resource allocation methodology proposed in this paper can be used for its solution.

Next we comment on issues associated with our approach to the solution of the multi rate multicast rate allocation problem formulated in this paper. Specifically, we address the following:

(a) The relation between a solution of **Max 1** and the choice of a particular D, defined in Section 3.4.
(b) Improvement in performance of the computation of the link price share.
(c) The assumption that the user utility functions are differentiable.
(d) The uniqueness of the optimal link price shares for a fixed set of link prices.
(e) The fact that the determination of service prices along each multicast tree is independent of the demand on other multicast trees (see Part I, Section 3, Property 3.6).

We discuss each issue separately.

(a) We proved the existence of a solution to the optimization problem **Max 1** and presented in Section 3.2.1 an algorithm (Algorithm ⊛) that converges to such a solution.

Algorithm ⊛ works by taking a sequence of sets, denoted by $\{P^D\}_{D\in\mathbb{N}}$, of evenly distributed points in the simplex S. For each one of these sets the algorithm generates a primitive set W^D, satisfying the properties of Lemma 1. The theory tells us that as D goes to infinity, a subsequence W^D of primitive sets will contain a solution of **Max 1**. Since the size (diameter) of W^D goes to 0 as D goes to infinity, we are able to approximate a solution of **Max 1** by Algorithm ⊛.

The question that remains is: "For a given D, do we know if a solution of **Max 1** is contained in W^D, and if not, how far away is W^D from a solution of **Max 1**?" We can not answer this question in general. The answer depends on the behavior of the excess demand function $z_l(\lambda)$, for all $l \in L$. Hence, without any further assumptions on the behavior of $z_l(\lambda)$, $l \in L$, all we can guarantee is that in the limit, as $D \to \infty$, Algorithm ⊛ will lead to a solution of **Max 1**. That is, as $D \to \infty$ the size of W^D decreases, and under certain regularity conditions (that relate the local maxima and local minima of $z_l(\lambda)$, $l \in L$) W^D will contain a solution of **Max 1**. On the other hand, based on the observations made while conducting the simulation of the algorithms we have concluded that in the case in which the utility functions are "smooth" (e.g. $\log(x)$, $\sqrt{x}$) Scarf's algorithm "closely" approximates an optimal set of link prices for relatively small values of D.

(b) We assumed that the computation performed by the price sharing algorithm in Part I of this two-part paper at each iteration of the auctioneer price adjustment scheme is done independently of prior iterations of the price sharing algorithm. It may be possible for the service provider to use the data from previous iterations of the price sharing algorithm to expedite the computational process. For that matter, learning theory may be useful to the service provider in determining the users' utility functions.

(c) We assumed that the user utility functions are differentiable. This assumption is essential in the proof of convergence of the price splitting algorithm used in the inner loop (Part I Lemma B.3) and the proof of the continuity of the excess demand function with respect to λ (Lemma 6). Both of these results are crucial in proving that the market based mechanism converges to an optimal solution of Problem **Max 1**. There is a possibility that using different proof technique methods the differentiability assumption may be removed.

We would also like to note that the differentiability assumption is not required in the development of the properties presented in Part I Section 3.

(d) We proved in Part I of this paper that for any link price λ we can determine an optimal service price $p(r, \lambda)$. In Appendix B we prove that for any $r \in R$ the allocation $x_r(p(r, \lambda))$ and the excess demand $z(\lambda)$ are continuous functions of λ. Due to the differentiability of the utility functions this implies that for any λ there is a unique $x_r(p_r(r, \lambda))$ and a unique $p_r(r, \lambda)$.

This is a surprising result to us since it shows that although the dimensionality of the shadow prices γ is much larger than the dimensionality of the link prices λ, for any set of link prices λ any set of optimal shadow prices γ generated from λ will generate the same service price $p(r, \lambda)$.

We also note that at the optimal solution of Problem **Max 1**, since the utility functions are strictly concave and the set of feasible solutions is convex, $x_r(p_r(r, \lambda))$ is unique. Since the utility functions are also assumed to be differentiable we also have that the set of service prices $p_r(r, \lambda)$ is unique. However, the set of λ generating the optimal set of service prices may not be unique.

(e) In Part I (Section 3 Property 3.6) we showed that the determination of service prices along a given multicast tree is independent of the demand on other multicast trees. This result has an interesting implication. Note that unicast is a special case of multicast with only one user connected to the tree. Therefore the same solution approach introduced in this paper may be applied equally to all flows (multicast and unicast) simultaneously on the same network by treating unicast flows as multicast. Under this utility maximizing formulation, the capacity constraints will be the total capacity on each link, therefore we do not need to dedicate link capacity solely for multicast services.

6 Conclusion

In Part II of this two-part paper we presented a market based mechanism and an adjustment process that satisfy the informational constraints imposed by the nature of multirate multicast. When they are combined with the results of Part I they result in an optimal solution of the corresponding centralized multirate multicast problem.

The two parts combined present a systematic approach to the optimal rate allocation in multirate multicast networks, that is described by a two-level convergent iterative procedure that leads to an optimal solution of a general decentralized rate allocation problem.

The main contributions of this two-part paper include (1) the development of properties of the optimal price per unit of service given an optimal price per unit of rate on each link (presented in Part I); (2) the construction of a distributed algorithm that determines the optimal service prices given a fixed set of prices per unit of rate on each link (presented in Part I); and (3) the development of a market-based mechanism which achieves an aggregate utility maximizing (i.e., welfare maximizing) solution for the informationally decentralized network problem (presented in Part II). The notion of "splitting tree" introduced in this paper was key to our overall development. In addition we presented numerical results on the convergence of both the price splitting algorithm (the inner loop) and the market-based iterative procedure (the outer loop).

As we have pointed out, the multicast rate allocation formulation adopted in this two-part paper is a reasonable one when the users are assumed to have no knowledge about the type of services they are receiving and how resources are shared under these services. If the users do possess such information, then a different formulation of the problem (e.g., regarding the links shared by multiple users as public goods) may be more appropriate. This remains an open problem, and is part of our future work.

Appendices

A Parameterized Utility Functions

In this section we present two classes of parameterized utility functions which appear in economic literature [13]. We assume that the exchange of information between the network and users is through prices, i.e. network advertises a price per unit of service to each user, and each user, based on its price, has a demand which maximizes its utility.

Bernoulli Utility Functions

Let $U(x)$ be of the form:

$$U(x) = \beta x^{\varrho} + \gamma \qquad \beta > 0,\ \varrho \in [0, 1),\ \gamma \in \mathbb{R}. \tag{A.1}$$

Note that since the user is utility maximizing, for a price p advertised by the network, the user will have a demand $x(p)$ for which:

$$U'(x(p)) \leq p \tag{A.2}$$

with equality being satisfied when $x(p)$ is an interior point of the set of the possible demands.

For this example equation (A.2) takes the following form:

$$p \geq \beta \varrho x(p)^{\varrho - 1} \tag{A.3}$$

Taking $p_1 \neq p_2$, and p_1, p_2 positive, we obtain

$$\frac{p_1}{p_2} = \left(\frac{x(p_1)}{x(p_2)} \right)^{\varrho - 1} \tag{A.4}$$

from which we can determine ϱ. Substituting ϱ back into equation (A.3) we can find the value of β. Note that the value of γ is irrelevant for the purpose of maximizing users' total utility, and it can be taken to be any arbitrary constant. So if the utility functions are of the form (A.1) the network can determine them in two iterations.

Exponential Utility Functions

Let $U(x)$ be of the form:

$$U(x) = 1 - e^{-ax} \qquad a > 0. \tag{A.5}$$

Substituting (A.5) in $U'(x(p)) \leq p$ we obtain:

$$p \geq ae^{-ax(p)} \tag{A.6}$$

Note that for values of p greater than a the demand $x(p)$ is zero. Also note that for any non-zero p which is smaller then a the demand will be strictly positive. Given two non-zero pairs of $(p, x(p))$ satisfying equation (A.6) one can determine the utility function $U(x)$.

In this case the number of iterations needed by the network in order to determine the users' utility functions is not fixed. This is due to the fact that the network has to first guess a small enough value of p which is less than a. After such a value of p is determined, only one more iteration is needed in order to determine parameter a. Note that if we know that the value of a is bounded below by some constant c, then by choosing p to equal $\frac{c}{2}$ and $\frac{c}{4}$ for example, we can determine the utility function.

In this appendix we presented methods for determining utility functions that belong to families parameterized by two parameters. A similar approach can be applied to utility functions that belong to families that are parameterized by a different number of parameters.

B Proof of Proposition 1

The proof of this proposition uses much of the terminology, notation, and many results developed in Part I of this two-part paper. We refer the reader to Part I of the paper for any notation or results not defined/stated in this section.

We will first show that $x_r(p(r, \lambda))$ is continuous function of λ. Using Property 3.6 from Part I, it is enough to only look at one fixed multicast tree, say $m \in M$. To prove continuity of the user demand $x_r(p(r, \lambda))$ with respect to link prices λ we have to show that given a fixed set of link prices $\{\lambda_l : l \in L_m\}$, and any link $l \in L_m$, a continuous change in the price λ_l will result in a continuous change in the demand of the users of multicast tree m. After establishing the continuity of $x_r(p(r, \lambda))$ with respect to λ, the continuity of excess demand $z(\lambda)$ with respect to λ follows immediately.

We proceed as follows: In Lemma 3 we describe a property of the rate demanded by the users of a splitting tree. In Lemmas 4 and 5 we obtain relations between changes in link prices, service prices and splitting trees. The results of Lemmas 3-5 are used to obtain Lemma 6 which establishes that the user service prices are monotonic functions of λ. Finally Lemma 6 together with Property 3.2 from Part I are combined to complete the proof of the theorem.

Lemma 3 *For any $r \in R_m$ and any $\lambda \geq \bar{0}$, $x_r(p(r, \lambda)) \geq x_q(p(q, \lambda))$, where $q \in \mathfrak{R}_r(\gamma(\lambda))$.*

This Lemma states that the rate demanded by the users of the splitting tree of user r is less then or equal to the rate of r.

Proof This follows directly from the definition of a splitting tree (Definition 3.2 from Part I).

Lemma 4 *Given λ and λ', if there exists $r \in R_m$ such that $p(r, \lambda) > p(r, \lambda')$, then there exists $r' \in R_m$ such that $p(r', \lambda) > p(r', \lambda')$ and $\mathcal{T}_{r'}(\gamma(\lambda)) \subseteq \mathcal{T}_{r'}(\gamma(\lambda'))$.*

Lemma 4 states that if for two different sets of link prices the service price of one user decreases, then there exist a user in the same multicast tree (possibly the same user) for whom the service price decreases and its splitting tree increases.

Proof Let $r' = \text{argmax}_{q \in \mathfrak{R}_r(\gamma(\lambda))} x_q(p(q, \lambda'))$. By Lemma 3 we have that $\mathcal{T}_{r'}(\gamma(\lambda)) \subseteq \mathcal{T}_r(\gamma(\lambda))$. Since $x_{r'}(p(r', \lambda')) \geq x_q(p(q, \lambda'))$, for any $q \in \mathfrak{R}_r(\gamma(\lambda))$, by the definition (Definition 3.2 Part I) of a splitting tree $\mathcal{T}_r(\gamma(\lambda)) \subseteq \mathcal{T}_{r'}(\gamma(\lambda'))$. From Lemma 3, the fact that the utilities are strictly concave, and the fact that $r \in \mathfrak{R}_r(\gamma(\lambda))$ respectively, we get the following sequence of inequalities: $x_{r'}(p(r', \lambda)) \leq x_r(p(r, \lambda)) < x_r(p(r, \lambda')) \leq x_{r'}(p(r', \lambda'))$. But these inequalities imply that $p(r', \lambda) > p(r', \lambda')$ which concludes the proof.

Lemma 5 *Given λ and λ', if there exists $r \in R_m$ such that $p(r, \lambda) < p(r, \lambda')$, then there exists $r' \in R_m$ such that $p(r', \lambda) < p(r', \lambda')$ and $\mathcal{T}_{r'}(\gamma(\lambda')) \subseteq \mathcal{T}_{r'}(\gamma(\lambda))$.*

Lemma 5 proves the converse of Lemma 4. In particular it states that if for two different sets of link prices the service price of one user increases, then there exist a user in the same multicast tree (possibly the same user) for which the service price increases and its splitting tree decreases.

Proof Consider the set of $\mathfrak{S} \triangleq \{q : q \in R_m,\ p(q,\lambda) < p(q,\lambda')\}$. Note that $\mathfrak{S} \neq \emptyset$ since $r \in \mathfrak{S}$. Let $s \in \mathfrak{S}$ such that $\ell(\mathcal{T}_s(\gamma(\lambda))) \geq \ell(\mathcal{T}_q(\gamma(\lambda)))$ for any $q \in \mathfrak{S}$. Note that if s is such that $\mathcal{T}_s(\gamma(\lambda))$ is the whole multicast tree, then $\mathcal{T}_s(\gamma(\lambda')) \subseteq \mathcal{T}_s(\gamma(\lambda))$ and the lemma is true, otherwise we prove that $\mathcal{T}_s(\gamma(\lambda')) \subseteq \mathcal{T}_s(\gamma(\lambda))$ by contradiction. Assume by contradiction that $\mathcal{T}_s(\gamma(\lambda)) \subset \mathcal{T}_s(\gamma(\lambda'))$. Let t be the parent branch of the root of $\mathcal{T}_s(\gamma(\lambda))$ (t exists since we have assumed that $\mathcal{T}_s(\gamma(\lambda))$ is not the whole multicast tree). Pick q to be a receiver such that $t \in \mathcal{T}_q(\gamma(\lambda))$ (q exists since there is at least one user downstream link t with a rate demand equal to the rate on link t). Note that $t \in \mathcal{T}_s(\gamma(\lambda'))$ which implies that $q \in \mathcal{T}_s(\gamma(\lambda'))$ (as t is the parent branch of the root of $\mathcal{T}_s(\gamma(\lambda))$ and $t \in \mathcal{T}_q(\gamma(\lambda))$), so $q \in \mathfrak{R}_s(\gamma(\lambda'))$. Also note that $\ell(\mathcal{T}_q(\gamma(\lambda))) > \ell(\mathcal{T}_s(\gamma(\lambda)))$, which implies that $q \notin \mathfrak{S}$. We have the chain of inequalities: $x_q(p(q,\lambda')) \leq x_s(p(s,\lambda')) < x_s(p(s,\lambda)) < x_q(p(q,\lambda))$, where the first inequality follows from Lemma 3, the second from the strict concavity of the utility functions and the fact that $s \in \mathfrak{S}$, and the third from the fact that $t \in \mathcal{T}_q(\gamma(\lambda))$ but $t \notin \mathcal{T}_s(\gamma(\lambda))$. The inequalities give us that $p(q,\lambda) < p(q,\lambda')$, so $q \in \mathfrak{S}$, which is a contradiction. Consequently $\mathcal{T}_s(\gamma(\lambda')) \subseteq \mathcal{T}_s(\gamma(\lambda))$.

The following lemma shows that if any of the link prices is increased (respectively decreased) then the optimal service price for any receiver can not decrease (respectively can not increase); that is for each receiver, the optimal service price is a monotonically increasing function of λ.

Lemma 6 *Let $\delta > 0$ and $\lambda'_l = \lambda_l + \delta$, and $\lambda''_l = \max(\lambda_l - \delta, 0)$, for some $l \in L_m$ and $\lambda'_e = \lambda''_e = \lambda_e$ for all $e \in L_m$, $e \neq l$. Then for all $r \in R_m$, $p(r,\lambda') \geq p(r,\lambda) \geq p(r,\lambda'')$.*

Proof In order to prove that $p(r,\lambda') \geq p(r,\lambda)$ it is enough to prove that $x_r(p(r,\lambda')) \leq x_r(p(r,\lambda))$. Assume by contradiction that there exists $r' \in R_m$ such that $x_{r'}(p(r',\lambda')) > x_{r'}(p(r',\lambda))$. Using Lemma 4 we can find an r (possibly same as r') such that $x_r(p(r,\lambda)) > x_r(p(r,\lambda))$ and $\mathcal{T}_r(\gamma(\lambda)) \subseteq \mathcal{T}_r(\gamma(\lambda'))$.

The sum of the service prices of the receivers of the splitting tree $\mathcal{T}_r(\gamma(\lambda))$ at link price λ' is:

$$\tau + \sum_{e \in L_m \cap \mathcal{T}_r(\gamma(\lambda))} \lambda_e + \varphi = \sum_{q \in \mathfrak{R}_r(\gamma(\lambda))} p(q,\lambda') \tag{B.1}$$

where τ is the price incurred by the receivers $\mathfrak{R}_r(\gamma(\lambda))$ from the links preceding the root of $\mathcal{T}_r(\gamma(\lambda))$, and φ is δ if $l \in L_m \cap \mathcal{T}_r(\gamma(\lambda))$ or 0 otherwise.

This sum can also be rewritten as:

$$\sum_{q \in \mathfrak{R}_r(\gamma(\lambda))} p(q,\lambda') = \sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \cap \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') + \sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \setminus \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') \tag{B.2}$$

where the first summation of the right hand side is the sum of the service prices of receivers of the splitting tree $\mathcal{T}_r(\gamma(\lambda))$, which at λ' have the same demand as r, and the second sum corresponds to the sum of the optimal service prices of the users which are not splitting with r.

For any $q \in \mathfrak{R}_r(\gamma(\lambda)) \cap \overline{\mathfrak{R}}_r(\gamma(\lambda'))$, $x_q(p(q,\lambda')) = x_r(p(r,\lambda')) > x_r(p(r,\lambda)) \geq x_q(p(r,\lambda))$, implies $p(q,\lambda') < p(q,\lambda)$. This gives us that:

$$\sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \cap \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') < \sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \cap \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) \tag{B.3}$$

Also note that

$$\sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \setminus \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') = \sum_{e \in \mathfrak{L}_r(\gamma(\lambda)) \setminus \overline{\mathfrak{L}}_r(\gamma(\lambda'))} \lambda_e + \varphi \tag{B.4}$$

$$\leq \sum_{q \in \mathfrak{R}_r(\gamma(\lambda)) \setminus \overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) + \varphi. \tag{B.5}$$

We derive (B.4) and (B.5) as follows: From Part I, Property 3.2, we have that the sum of link prices in a splitting tree is equal to the sum of service prices over all the users of that particular splitting tree. From Part I, Property 3.3, we have that on a fixed splitting tree, the sum of the link prices over the links with maximal rate is equal to the sum of the service prices over all the users demanding the maximal rate on that particular splitting tree. Combining these two results we have that for a fixed splitting tree, the sum of the service prices over the users demanding strictly less then the maximal rate on that splitting tree is equal to the sum of the link prices over the links on which the rate demanded is not maximal, which implies the equality in (B.4). Property 3.4 from Part I states that for any given subtree, the sum of service prices of the users on that subtree is equal to the sum of link prices of that subtree plus the price on the links incurred upstream of the subtree. This result implies the inequality in (B.5).

Combining equations (B.3)-(B.5) we get:

$$\sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\cap\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') + \sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda') \\ < \sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\cap\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) + \sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) + \varphi \quad \text{(B.6)}$$

while,

$$\sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\cap\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) + \sum_{q\in\mathfrak{R}_r(\gamma(\lambda))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda'))} p(q,\lambda) + \varphi = \sum_{q\in\mathfrak{R}_r(\gamma(\lambda))} p(q,\lambda) + \varphi \\ = \sum_{e\in L\cap\mathcal{T}_r(\gamma(\lambda))} \lambda_e + \varphi. \quad \text{(B.7)}$$

Combining equations (B.1),(B.2),(B.6), and (B.7) we get $\tau+\sum_{e\in L_m\cap\mathcal{T}_r(\gamma(\lambda))}\lambda_e+\varphi < \sum_{e\in L_m\cap\mathcal{T}_r(\gamma(\lambda))}\lambda_e+\varphi$. This contradiction is due to the fact that we have assumed that $x_r(p(r,\lambda')) > x_r(p(r,\lambda))$. Consequently, $p(r,\lambda') \geq p(r,\lambda)$.

For the second inequality the problem is similar. Assume that there exists $r \in R$ such that $x_r(p(r,\lambda)) > x_r(p(r,\lambda''))$. Pick the r such that Lemma 5 is satisfied. Then:

$$\sum_{e\in L_m\cap\mathcal{T}_r(\gamma(\lambda''))} \lambda_e - \varphi = \sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))} p(q,\lambda'') \quad \text{(B.8)}$$

$$= \sum_{q\in\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda'') + \sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda'') \quad \text{(B.9)}$$

$$> \sum_{q\in\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda) + \sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda) - \varphi \quad \text{(B.10)}$$

$$= \sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))} p(q,\lambda) - \varphi \quad \text{(B.11)}$$

$$= \sum_{e\in L_m\cap\mathcal{T}_r(\gamma(\lambda''))} \lambda_e + \tau - \varphi \quad \text{(B.12)}$$

Equations (B.8), (B.9), and (B.11) we get by rearranging the terms. Equation (B.10) is true by the same argument as in the proof of the first part, since $\sum_{q\in\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda'') > \sum_{q\in\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda)$, and $\sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda'') > \sum_{q\in\mathfrak{R}_r(\gamma(\lambda''))\setminus\overline{\mathfrak{R}}_r(\gamma(\lambda''))} p(q,\lambda) - \varphi$. Finally equation (B.12) follows by Property 3.4 from Part I, where τ is the price incurred on the links upstream of $\mathcal{T}_r(\gamma(\lambda''))$.

The above chain of equations gives us the contradiction, which is due to the assumption that $x_r(p(r,\lambda'')) < x_r(p(r,\lambda))$. Consequently, $p(r,\lambda) \geq p(r,\lambda'')$.

Based on Lemma 6 and Property 3.2 from Part I we complete the proof of Lemma 1. Let $\varepsilon > 0$ and fix $l \in L_m$ and $\lambda \geq 0$. Using Assumption 1, for each receiver r we can choose a $\delta_r > 0$ such that for any $y \in \big(p(r,\lambda)-\delta_r, p(r,\lambda)+\delta_r\big)$, $\|x_r(y) - x_r(p(r,\lambda))\| < \varepsilon$. Take $\delta = \min_{r\in R_m} \delta_r$.

We prove now that the function $x_r(p(r,\lambda))$ is continuous from the right. Let $\lambda'_l \in \big(\lambda_l, \lambda_l+\delta\big)$. Using Property 3.2 from Part I we have:

$$\sum_{r\in R_m} p(r,\lambda) = \sum_{e\in\mathfrak{L}_r(\gamma(\lambda))} \lambda_e = \sum_{e\in\mathfrak{L}_r(\gamma(\lambda'))} \lambda_e + (\lambda_l - \lambda'_l) = \sum_{r\in R_m} p(r,\lambda') + (\lambda_l - \lambda'_l). \quad \text{(B.13)}$$

From Lemma 6 we have that

$$p(r,\lambda) \le p(r,\lambda') \quad \forall r \in R_m. \tag{B.14}$$

Combining equations (B.13) and (B.14) we get that for all $r \in R_m$ we have $|p(r,\lambda) - p(r,\lambda')| < \delta$, which implies that $\|x_r(p(r,\lambda')) - x_r(p(r,\lambda))\| < \varepsilon$. This concludes the proof of the right continuity of $x_r(p(r,\lambda))$ in λ.

For the proof of left continuity we proceed in a similar fashion. Let $\lambda_l'' \in \big(\lambda_l - \delta, \lambda_l\big)$. Using Property 3.2 from Part I we have:

$$\sum_{r\in R_m} p(r,\lambda) = \sum_{e\in\mathfrak{L}_r(\gamma(\lambda))} \lambda_e = \sum_{e\in\mathfrak{L}_r(\gamma(\lambda''))} \lambda_e + (\lambda_l - \lambda_l'') = \sum_{r\in R_m} p(r,\lambda'') + (\lambda_l - \lambda_l''). \tag{B.15}$$

From Lemma 6 we have that

$$p(r,\lambda) \ge p(r,\lambda'') \quad \forall r \in R_m. \tag{B.16}$$

Combining equations (B.15) and (B.16) we get that for all $r \in R_m$ we have $|p(r,\lambda) - p(r,\lambda'')| < \delta$, which implies that $\|x_r(p(r,\lambda'')) - x_r(p(r,\lambda))\| < \varepsilon$. This concludes the proof of the left continuity of $x_r(p(r,\lambda))$ in λ. Since $x(p(r,\lambda))$ is left and right continuous in λ it is a continuous function in λ.

Since max and summation are continuous operators, the continuity of $x_r(p(r,\lambda))$ implies that $z(\lambda)$ will be a continuous function of λ. □

C Proof of Proposition 2

Let $q \in \mathfrak{C}$. Then $q_0 > 0$ because $q \in \mathfrak{C}_l$. Furthermore, since $q \in \mathfrak{C}_0$ and $q_0 > 0$, we have that:

$$\sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda(q))) \le c_l, \quad \text{for all} \quad l \in L. \tag{C.1}$$

which makes $x(\lambda(q)) \triangleq \{x_r(p(r,\lambda(q)))\}_{r\in R}$ a feasible solution to problem **Max 1**. We now define the Lagrangian function:

$$\Lambda(x,\lambda) \triangleq \sum_{r\in R} U_r(x_r) - \sum_{l\in L} \lambda_l \Big(\sum_{m\in M} \max_{r\in R_{l,m}} x_r - c_l \Big) \tag{C.2}$$

Let x be any other feasible solution to **Max 1**. Then from Part I, Section 3, we have that:

$$\Lambda\big(\, x(\lambda(q)), \lambda(q)\,\big) \ \ge \Lambda(x,\lambda(q)) \ , \tag{C.3}$$

From (C.3) we have,

$$\sum_{r\in R} U_r(x_r(p(r,\lambda(q)))) - \sum_{l\in L} \lambda_l(q) \Big(\sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda(q))) - c_l \Big) \\ \ge \sum_{r\in R} U_r(x_r) - \sum_{l\in L} \lambda_l(q) \Big(\sum_{m\in M} \max_{r\in R_{l,m}} x_r - c_l \Big) \tag{C.4}$$

Notice that $q \in \mathfrak{C}_l$ implies that $q_l = 0$ (i.e. $\lambda_l(q) = 0$), or that,

$$\sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda(q))) \ge y_l(\lambda(q)) = c_l. \tag{C.5}$$

If $q_l > 0$, then from (C.1) and (C.5) it follows that

$$\sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda(q))) = c_l. \tag{C.6}$$

Therefore,

$$\sum_{l\in L} \lambda_l(q) \Big(\sum_{m\in M} \max_{r\in R_{l,m}} x_r(p(r,\lambda(q))) - c_l \Big) = 0. \tag{C.7}$$

Since x is a feasible solution to **Max 1** we have that,

$$\sum_{m \in M} \max_{r \in R_{l,m}} x_r \leq c_l \ , \quad \text{for all} \quad l \in L \ . \tag{C.8}$$

Multiplying (C.8) by $\lambda_l(q) \geq 0$ and summing over all $l \in L$, gives,

$$\sum_{l \in L} \lambda_l(q) \bigg(\sum_{m \in M} \max_{r \in R_{l,m}} x_r - c_l \bigg) \leq 0 \ . \tag{C.9}$$

Inequalities (C.4), (C.7) and (C.9), give,

$$\sum_{r \in R} U_r\big(x_r(\lambda(q))\big) \geq \sum_{r \in R} U_r(x_r) \ , \tag{C.10}$$

which shows that $x(\lambda(q))$ solves **Max 1**. This concludes the proof of Proposition 2. □

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

References

1. M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming Theory and Algorithms.* John Wiley, New York, 1993.
2. 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.
3. 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.
4. B. Eaves. Homotopies for computation of fixed points. *Mathematical Programming*, 3(1):1–27, 1972.
5. 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.
6. L. Hurwicz. The design of mechanisms for resource allocation. *American Economic Review*, 63(2):1–30, 1973.
7. L. Hurwicz. *On incformational decentralization and efficiency in resource allocation mechanisms*, volume 25, pages 238–250. 1986.
8. K. Kar, S. Sarkar, and L. Tassiulas. Optimization based rate control for multirate multicast sessions. In *Proceedings of INFOCOM*, Alaska, 2001.
9. 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.
10. 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.
11. 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.
12. O. Mangasarian. *Nonlinear Programming.* SIAM, New York, 1994.
13. A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory.* Oxford University Press, New York, 1995.
14. S. McCanne, V. Jacobson, and M. Vetterli. Receiver driven layered multicast. In *Proceedings of ACM SIGCOMM*, Stanford, CA, 1996.
15. O. Merrill. *Applications and extensions of an Algorithm that Computes Fixed Points of CertainUpper Semi-Continuous Point to Set Mappings.* PhD thesis, University of Michigan, 1972.
16. K. Mount and S. Reiter. The informational size of message spaces. *J. Econ. Theory*, 8:161–192, 1971.
17. Y. Nesterov and A. Nemirovsky. *Interior Point Polynomial Algorithms in Convex Programming.* SIAM, 1994.
18. J. Renengar. *A Mathematical View of Interior-Point Methods in Convex Optimization.* MPS-SIAM Series on Optimization, Cornell University, 2001.
19. D. Rubenstein, J. Kurose, and D. Towsley. The impact of multicast layering on network fairness. In *Proceedings of ACM SIGCOMM*, Cambridge, MA, 1999.
20. S. Sarkar and L. Tassiulas. Fair allocation of resources in multirate multicast trees. In *Proceedings of Globecom*, 1999.

21. 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.
22. S. Sarkar and L. Tassiulas. Fair allocation of disrete bandwidth layers in multicast networks. In *Proceedings of INFOCOM*, Tel Aviv, Israel, 2000.
23. S. Sarkar and L. Tassiulas. Layered bandwidth allocation for multicasting of hierarchically encoded sources. 2000.
24. S. Sarkar and L. Tassiulas. Back pressure based multicast scheduling for fair bandwidth allocation. In *Proceedings of INFOCOM*, Alaska, 2001.
25. 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.
26. H. Scarf. *The Computation of Economic Equilibria*. Yale University Press, New Haven and London, 1973.
27. J. K. Shapiro, D. Towsley, and J. Kurose. Optimization-based congestion control for multicast communications. In *Proceedings of IEEE INFOCOM*, Tel Aviv, Israel, 2000.
28. C.P. Simon and L. Blume. *Mathematics for Economists*. W. W. Norton, New York, 1994.
29. 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.
30. 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.
31. T. Turletti and J.C. Bolot. Issues with multicast video distribution in heterogeneous packet networks. Packet Video Workshop, 1994.
32. 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.
33. G. van der Laan and A. Talman. A restart alrorithm for computing fixed points without an extra dimension. *Mathematical Programming*, 17:74–84, 1979.
34. G. van der Laan and A. Talman. Interpretation of the variable dimension fixed point algorithm with an artificial level. *Mathematics of Operations Research*, 8:86–99, 1983.