A Low-overhead Rate Control Algorithm for Maximizing Aggregate Receiver Utility for Multirate Multicast Sessions

Koushik Kar Saswati Sarkar Leandros Tassiulas
Department of Electrical & Computer Engg.
University of Maryland
College Park, MD 20742, USA
{koushik,swati,leandros}@isr.umd.edu

Abstract

In multirate multicasting, different users (receivers) within the same multicast group could receive service at different rates, depending on user requirements and network congestion level. Compared to unirate multicasting, this provides more flexibility to the user, and allows more efficient usage of network resources. In this paper, we address the rate control problem for multirate multicast sessions, with the objective of maximizing the total receiver utility. This aggregate utility maximization problem not only takes into account the heterogeneity in user requirements, but also provides a unified framework for diverse fairness objectives. We propose an algorithm for this problem and show, through analysis and simulation, that it converges to the optimal rates. In spite of the non-separability of the problem, the solution that we develop is completely decentralized, scalable and does not require the network to know the receiver utilities. Moreover, the algorithm requires very simple computations both for the user and the network, and also has very low overhead of network congestion feedback.

I. Introduction

In conventional or unirate multicasting, all receivers of the same multicast group receive service at the same rate. However, in general, different receivers belonging to the same multicast group can have widely different characteristics. Thus a single rate of transmission per multicast group is likely to overwhelm the slow receivers and starve the fast ones. Multirate transmission, where the receivers of the same multicast group could receive data at different rates, can be used to accommodate these diverse requirements. Naturally, multirate multicasting is the preferred mode of data delivery for many real-time applications, including teleconferencing and audio/video broadcasting. Multirate transmission allows a receiver to receive data at a rate that is commensurate with its requirements and capabilities, and also with the capacity of the path leading to it from the source. One way of achieving multirate transmission is through hierarchical encoding of real time signals. In this approach, a signal is encoded into a number of layers that can be incrementally combined to provide progressive refinement. This layered transmission scheme can be used for both audio and video transmissions over the internet [6] [26], and has potentials for use in ATM networks as well [12]. In the case of the internet, each layer can be transmitted as a separate multicast group and receivers can adapt to congestion by joining and leaving these groups (see [15] and [17] for internet protocols for adding and dropping layers). Note that in multirate multicasting, there is no unique multicast session rate, and one needs to consider receiver rates separately. Also note that in this case, the transmission rate of a multicast session (multicast group) on a link needs to be equal to the maximum of the rates of all receivers downstream of that link (since it has to match the fastest of the downstream receivers).

Compared to unirate multicasting, multirate multicasting also allows more efficient use of the network resources. For efficient use of the network, an effective rate control strategy is necessary. The rate control algorithm should ensure that the traffic offered to a network by different traffic sources remain within the limits that the network can carry. Moreover, it should also ensure that the network resources are shared by the competing flows in some fair manner. It may therefore be desirable that the rate control algorithm would steer the network towards a point where some measure of global fairness is maximized.

There can be many acceptable definitions of fairness, some well-known ones being max-min fairness [4], proportional fairness [10]. Fairness definitions can be generalized in a nice way by using utilities. Utility of an user is a function connecting the bandwidth given to the user with the "value" associated with the bandwidth (note that throughout the paper, the terms "user" and "receiver" are used synonymously). The utility could be some measure of say, the perceived quality of audio/video, the user satisfaction, or even the amount paid by the user for the bandwidth allotted to it. In this paper, we try to design the rate control algorithms such that they maximize the sum of the utilities over all receivers, subject to the link capacity constraints. This objective that was proposed recently by Kelly [10]. It is easy to see that various fairness objectives can be realized within this utility maximization framework for different choices of the utility functions (see [16]). Note that in our problem, the utility functions can be different for different users (receivers). Thus this framework allows us to differentiate among receivers on the basis of their requirements and/or revenues. This is important, since receivers could have heterogeneous requirements, and the same amount of bandwidth could be valued differently by different receivers.

Very recently, there has been a considerable interest in the problem of fair allocation of resources for multirate multicast sessions. However, most of the work in this area is concerned only with the notion of max-min fairness (see [20], [22], [23], [24]). Although there has been a lot of research on the utility maximization problem for unicast case ([11],[14],[25],[13],[7]), the multirate multicast case has not received significant attention. It is worth noting here that there are a number of factors which make the multirate multicast problem significantly different and considerably more complex than its unicast version.[1] For instance, the problem in the multirate multicast case is non-separable and non-differentiable, unlike the unicast case (we discuss more on this in the subsequent sections). To the best of our knowledge, [8] is the only work that addresses the multirate multicast utility maximization problem. Here, the authors propose a distributed and scalable algorithm for the problem; the solution is based on dual methods. In this paper, we take a different approach, and derive a primal algorithm based on non-differentiable optimization methods. The algorithm that we propose is scalable, distributed, and does not require the network to know the receiver utilities. Moreover, both the user and network (link) sub-algorithms are extremely simple, and the overhead of the communication between the network and the user is also very low. These features makes the algorithm very attractive in terms of practical deployment. On the other hand, the algorithm in [8] suffers from several practical shortcomings (a detailed comparison of the algorithm proposed in this paper and that in [8] is presented in Section VIII of this paper).

The paper is structured as follows. In Section II, the rate control problem is presented formally as an optimization problem. In Section III, we state the algorithm requirements, and outline our basic solution approach. In Section IV,

[1]However, note that the problem for the unirate multicast case is much simpler, and in general, the solutions for the unicast case can be directly extended to that case.

we present an iterative algorithm for the rate control optimization problem. Section V presents the convergence analysis for this iterative optimization algorithm. In Section VI, we describe how this algorithm can be implemented in a real network. In Section VII, we demonstrate the convergence of our algorithm in an asynchronous network environment through simulations. We compare our approach with the existing approaches in Section VIII, and conclude in Section IX.

II. Problem Statement

We will first describe the network model, and formulate the rate control problem as a convex optimization problem. In the subsequent sections, we will show how we can achieve the optimal rates for this problem.

Consider a network consisting of a set L of unidirectional links, where a link $l \in L$ has capacity c_l. The network is shared by a set of M multicast groups (sessions). Each multicast group is associated with a unique source, a set of receivers, and a set of links that the multicast group uses (the set of links form a tree[2]). Thus any multicast group $m \in M$ is specified by $\{s_m, R_m, L_m\}$ where s_m is the source, L_m is the set of links in the multicast tree, and R_m is the set of receivers in group m. As already mentioned, the total rate of traffic of a multicast group over any link on the tree must be equal to the maximum of the traffic rates of all downstream receivers of the group.

Let R be the set of all receivers over all multicast groups. Also let $S_l \subseteq R$ denote the set of receivers using link $l \in L$. Each session has a minimum required transmission rate $b_r \geq 0$, and a maximum required transmission rate $B_r < \infty$. Moreover, each session r is associated with a utility function $U_r : \Re_+ \rightarrow \Re$, which is assumed to be concave, continuous, bounded and increasing in the interval $X_r = [b_r, B_r]$.[3] Thus receiver r has a utility $U_r(x_r)$ when it is transmitting at a rate x_r, where $x_r \in X_r$.

We are interested in maximizing the "social welfare", i.e., sum of the utilities over all receivers, subject to the link constraints, as well as the maximum/minimum rate constraints. The problem can be posed as:

$$\mathbf{P}: \quad \text{maximize} \quad \sum_{r \in R} U_r(x_r)$$

$$\text{subject to} \quad \sum_{m \in M} \max_{r \in S_l \cap R_m} x_r \leq c_l \qquad \forall\, l \in L \tag{1}$$

$$x_r \in X_r \qquad \forall r \in R \tag{2}$$

Note that $S_l \cap R_m$ is the set of receivers of group m that use link l. Thus the term $\max_{r \in S_l \cap R_m} x_r$ denotes the rate of traffic of multicast group m on link l. Also note that when $S_l \cap R_m = \phi$, the term $\max_{r \in S_l \cap R_m} x_r$ in (1) should be interpreted as zero.

III. Preliminaries

In this section, we introduce some new terminology, which will help us in describing the algorithms presented in the subsequent sections of this paper. We then discuss the features that are necessary in any multirate multicast rate control algorithm for the algorithm to be practically viable. In this section, we also outline the basic solution approach that is used in deriving the optimization algorithm presented in the next section.

[2]We assume fixed path routing. So the tree associated with each multicast group is fixed.

[3]We assume that, in general, the function U_r is known only to the receiver r.

A. Terminology

Consider Figure 1, which shows an example of a multicast tree where s is the source node and $\{r_1, r_2, r_3, r_4\}$ is the set of receiver nodes. The rest of the nodes in the multicast tree can be classified into *junction nodes* and *non-junction nodes*, as shown in the figure. Junction nodes are the forking nodes, i.e., nodes where the multicast tree "branches off". Thus in Figure 1, $\{r_5, r_6, r_7\}$ are junction nodes. Receiver/junction nodes of different multicast groups are considered to be logically different, even if they are physically located at the same node. In the rest of the paper, we assume that the receivers are only at the leaf nodes of the multicast tree. There is no loss of generality in assuming this, since a receiver at a non-leaf node can be replaced by creating a new leaf node and placing the receiver in it, and connecting the new leaf node to the non-leaf node (where the receiver is actually located) by a link with infinite capacity. Moreover, note that any leaf node must be a receiver node. The *parent* of a receiver/junction node r refers to the closest junction/source node in the upstream path from r towards the source. Also, by *child* of junction/source node r, we would refer to any receiver/junction node whose parent is the node r. Thus in Figure 1, r_5 is the parent of r_1, r_7 is the parent of r_5, s is the parent of r_7. Similarly, r_7 is a child of s, while r_5, r_6 are children of r_7, and so on.

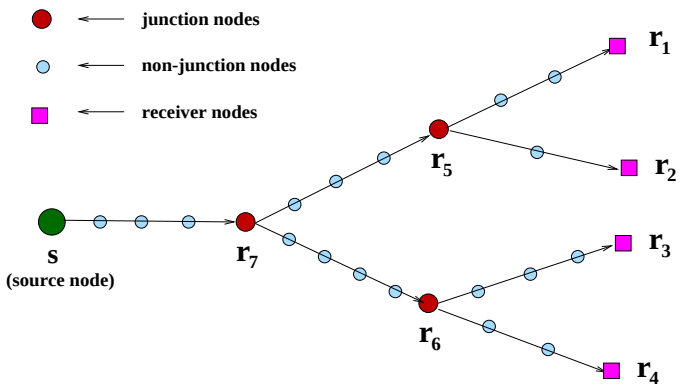

Fig. 1. An example of a multirate multicast tree

In general, we assume that the receiver decides its rate based on its utility function and the network congestion feedback. It then sends its request to its parent node. A junction node gathers all such requests (from its children nodes), takes the maximum of all the rates requested, and requests that rate from its parent node. Requests go up the tree through the junction nodes in this fashion until it reaches the source node. The source sends traffic to its children nodes at their requested rates; these nodes then send traffic to their children nodes, and so on, and the traffic finally reaches the receivers at their requested rates.

B. Algorithm Requirements

In order to be practically viable, a rate control algorithm must be decentralized. Thus we would like to have a solution where the nodes in the network act like processors in a distributed computation system (where the coordinating information is exchanged in terms of congestion and rate feedbacks) and reach the system optimum without any centralized coordinator.

Closely tied to decentralization is the issue of scalability. A solution would not scale if, for example, the source or a junction node in the multicast tree has to maintain some state information for all downstream receivers of the tree. Since

the number of receivers in the group can be very large, this might lead to tremendous processing/storage pressure on such a node, particularly if the node is the source, or a junction node close to the source. Therefore we would like to have a solution where processing/storage overhead at a node in a multicast tree does not depend significantly on the size of the tree.

Conformity with existing standards in another important criterion. The rate control algorithm should be such that it can be implemented without a major modification to the existing standards. In the current networking standards like IP multicast, a junction node may not know the identity of all the downstream receivers, but will only know the downstream nodes it must forward a packet to. Therefore we require a rate control algorithm which does not require a junction node to communicate with nodes other than its immediate neighbors.

Moreover, we would also like to have a solution where the overhead of state information maintained (stored/updated) at the routers is as small as possible. For routers that are junction nodes of one or more multicast sessions, some overhead is unavoidable. This is because a junction node needs to store at least the rate information about all of its children. Thus a router has to maintain per-session information for all multirate multicast sessions for which it is a junction node. However, we would like to have a solution where the routers would not need to maintain any state information for a session for which it is a non-junction node. Thus in such a solution, no per-flow state would be required at the network nodes if all the sessions are unicast (since there are no junction nodes in the unicast case).

We would also prefer to have a solution where most of the computations (required for the optimization process) are limited to the end-host only. For practical viability, the computations that the core routers are required to perform must be kept very simple.

It is also desirable that the overhead of information exchange (required in the optimization process) between the network and end-hosts would be as low as possible, such that it can be contained within a few bytes in the packet header.

The rate control algorithm that we propose in this paper satisfies all of the above criteria. It is distributed, and the user and network algorithms are appealingly simple. The algorithm also has a very low network feedback overhead. In the solution we propose, the amount of extra state information that a junction node in the multicast tree has to maintain, depends only on the number of links (of that particular multicast tree) originating from that node. Moreover, a source/junction/receiver node only needs to communicate with its parent or children nodes, and does not need to know about the nodes further downstream/upstream. Thus the solution is scalable, and conforms well with the existing standards. In our algorithm, all the network needs to know from the receivers are the receiver rates. This, however, can also estimated by measuring the rates at the network nodes. In our algorithm, with measurement-based estimation of receiver rates, a router does not need to maintain per-session information for sessions for which it is a non-junction node; the per-session information required for sessions for which it is a junction node is also small, as already mentioned.

C. Solution Approach

Note that in the unicast version of the problem, the link constraints are linear and the problem $\mathbf{P}$ is separable. Separable problems are amenable to distributed solutions (see [3]). In our case, however, the problem $\mathbf{P}$ contains some $\max$ functions. The $\max$ functions, besides being nonlinear, couple several variables together, making the problem non-separable. Moreover, note that the $\max$ functions are non-differentiable. All these factors make the problem significantly

different than its unicast version. Obtaining a solution that is satisfies all the requirements described in the last subsection is an interesting and challenging problem.

Note that $\max$ functions are piecewise linear, and hence the constraint set in (1) can be replaced by a set of linear inequalities. Linearizing the constraint set makes the problem separable, and thus helps in obtaining a distributed solution. This linearization can be done in several different ways, and if not done appropriately, it can result in a very large number of linear constraints, posing a difficulty in obtaining a scalable solution. The linearization approach was taken in [8], which shows how the linearization can be done so as to obtain a scalable solution. Distributed algorithms are then obtained by applying dual methods to this linearized problem.

In this paper, we take a different approach. The algorithm that we propose is obtained through non-differentiable optimization methods, particularly those based on *subgradients*. A subgradient, defined in the context of convex/concave functions, can be viewed as a generalized gradient, and may exist even if the gradient does not (as is the case for non-differentiable functions). See Appendix I for the formal definition of subgradients and some of their important properties. The motivation, derivation and analysis of our algorithm are heavily based on results in subgradient optimization theory, mainly those by N.Z. Shor and B.T. Poljak [21] [18]. It is worth noting here that although the link constraints in problem $\mathbf{P}$ are non-separable, their subgradients are separable. This is one of the key observations that enables us to obtain a distributed solution to the non-separable problem $\mathbf{P}$ using a subgradient approach. Our algorithm is developed in such a way that the scalability and other requirements stated above are also appropriately addressed.

It is also worth noting here that unlike the algorithm in [8] (which is dual-based), our algorithm is a primal algorithm. The algorithm presented here has significant advantages over the one presented in [8], in terms of the objectives outlined in the previous subsection (see Section VIII).

IV. An Optimization Algorithm

In this section, we present an iterative optimization algorithm for the problem $\mathbf{P}$. The convergence properties of the algorithm is investigated in the next section. In Section VI we show how this algorithm can be implemented in a real network in a distributed and scalable way.

A. Notation

Before we present the algorithm, we introduce some notation that we will use. For quick reference, the notation introduced here is also presented in Appendix III. Let$\hat{R}$ be the set of all junction nodes (over all multicast groups). Let $\tilde{R} = R \cup \hat{R}$ be the set of all receiver and junction nodes (over all multicast groups). For any $r \in\tilde{R}$, let π_r denote the parent node of r. Thus in Figure 1, $\pi_{r_1} = r_5$, $\pi_{r_7} = s$, etc. For any $r \in \hat{R}$, let $C_r = \{r^{'} : \pi_{r^{'}} = r\}$ denote the set of all children nodes of r. Thus in Figure 1, $C_{r_7} = \{r_5, r_6\}, C_{r_5} = \{r_1, r_2\}$, etc. For any $r \in \tilde{R}$, let $\tilde{L}_r$ denote the set of all links whose immediate downstream junction/receiver node is r. In other words,$\tilde{L}_r$ is the set of all links between the nodes π_r and r in the particular multicast tree to which π_r and r belongs. Thus in Figure 1, $\tilde{L}_{r_7}$ consists of all links between s and r_7, $\tilde{L}_{r_5}$ consists of all links between r_7 and r_5, $\tilde{L}_{r_1}$ consists of all links between r_5 and r_1, and so on. Now define the set $\tilde{S}_l \subseteq \tilde{R}$ as $\tilde{S}_l = \{r : l \in \tilde{L}_r\}$. Thus $\tilde{S}_l$ consists of all junction and receiver nodes that are the immediate downstream nodes of sessions that go through link l.

For any $r \in \hat{R}$, let T_r denote the set of receiver nodes that are included in the tree rooted at r. Thus in Figure 1, $T_{r_5} = \{r_1, r_2\}$, $T_{r_7} = \{r_1, r_2, r_3, r_4\}$ etc. Now for each $r \in \hat{R}$, define a variable x_r such that it denotes the rate of traffic that the junction node r receives from its parent node (we will call these "junction rates" in analogy with "receiver rates"). Note that the junction rates are simply a function of the receiver rates. Thus for any $r \in \hat{R}$, x_r is defined as $x_r = \max_{r' \in T_r} x_{r'}$. Moreover, with this notation, for any $r \in \hat{R}$, $x_r = \max_{r' \in C_r} x_{r'}$. Also note that the capacity constraint for link l (cf., (1)) can now be simply written as $\sum_{r \in \tilde{S}_l} x_r \leq c_l$.

For any $r \in \tilde{R}$, let Q_r denote the set of all junction and receiver nodes from the source node to r, including r but excluding the source node. Thus in Figure 1, $Q_{r_5} = \{r_7, r_5\}$, $Q_{r_1} = \{r_7, r_5, r_1\}$, and so on.

B. An Iterative Algorithm

For any $r \in R$, let $x_r^{(n)}$ denote the rate of the receiver node r at the nth iterative step. Then for any $r \in \hat{R}$, $x_r^{(n)} = \max_{r' \in T_r} x_{r'}^{(n)}$ denotes the rate of the junction node r at the nth iterative step.

In our algorithm, the rate update for receiver r at the nth iterative step can be summed up as follows: $x_r^{(n)}$ increases according to the "incremental utility" $U_r'(x^{(n)})$, while it decreases according to the "congestion penalty" $p_r^{(n)}$ ($p_r^{(n)}$ will be defined shortly). $p_r^{(n)}$ can thought of as a measure of the congestion caused by r at step n, and thus determines the rate at which r "backs off" on detecting congestion in its path. As we will see later (when we describe the practical implementation of the algorithm in Section VI), the congestion penalty is basically the congestion feedback provided by the network to the receiver (user). Before we describe the rate update procedure in detail, let us define the congestion penalty formally in terms of the receiver rates and network parameters.

First we will introduce a few variables that will be useful in defining the congestion penalty $p_r^{(n)}$. For each link $l \in L$, define $\varepsilon_l^{(n)}$ as

$$\varepsilon_l^{(n)} = \begin{cases} 0 & \text{if} \quad \sum_{m \in M} \sum_{r \in \tilde{S}_l} x_r^{(n)} \leq c_l \\ 1 & \text{if} \quad \sum_{m \in M} \sum_{r \in \tilde{S}_l} x_r^{(n)} > c_l \end{cases} \tag{3}$$

We will refer to the variable ε_l as the "link congestion indicator" for link l. Now, for each $r \in \tilde{R}$, define $e_r^{(n)}$ as

$$e_r^{(n)} = \sum_{l \in \tilde{L}_r} \varepsilon_l^{(n)} \tag{4}$$

Therefore, $e_r^{(n)}$ indicates how many of the links in $\tilde{L}_r$ are congested at step n.

Let Ω be the set of all source nodes (over all multicast groups). Let $\tilde{R}_\Omega \subseteq \tilde{R}$ be the set of all junction and receiver nodes whose parent node is a source node. Thus $\tilde{R}_\Omega = \{r : \pi_r \in \Omega\}$. Associate a variable α_r satisfying $0 \leq \alpha_r \leq 1$ with each $r \in \tilde{R} \setminus \tilde{R}_\Omega$. We will refer to α_r as the "penalty splitting factor" associated with junction/receiver node r, the reasons for which will be clarified shortly. Let $\alpha_r^{(n)}$ denote the penalty splitting factor for r at the nth iterative step. We will require these penalty splitting factors to satisfy certain conditions, as we will see later.

Now, for each $r \in R$, define $p_r^{(n)}$ as

$$p_r^{(n)} = \sum_{r' \in Q_r} \Big(\prod_{r'' \in Q_r \setminus Q_{r'}} \alpha_{r''}^{(n)} \Big) \; e_{r'}^{(n)} \tag{5}$$

Note that for $r' = r$, the term $\prod_{r'' \in Q_r \setminus Q_{r'}} \alpha_{r''}^{(n)}$ should be interpreted as 1. The above definition can be motivated as follows. Let us interpret $\varepsilon_l^{(n)}$ as the penalty to be paid for congesting link l (by each of the multicast sessions using link l) at step n. Now consider a junction node r' belonging to any multicast group m. Then $e_{r'}^{(n)}$ is the total penalty to be paid by m for congesting the links in $\tilde{L}_{r'}$. Let this penalty be charged to r' (recall that for links in $\tilde{L}_{r'}$, r' is the closest downstream node belonging to m's multicast tree). Now let r' split this penalty among its children nodes. Also for any $r'' \in C_{r'}$, let $\alpha_{r''}^{(n)}$ be the factor that determines what proportion of this penalty is charged to r'' (thus r'' is charged a penalty of $\alpha_{r''}^{(n)} e_{r'}^{(n)}$). Each child node then splits the penalty charged to it amongst its children nodes (again according to the splitting factors of the nodes that are charged), and this goes on until the penalties are transferred to the receivers. It is then easy to see that the penalty charged to receiver $r \in T_{r'}$ is equal to $(\prod_{r'' \in Q_r \setminus Q_{r'}} \alpha_{r''}^{(n)})\ e_{r'}^{(n)}$. Also note that for any receiver node r, the penalty for congesting the links in $\tilde{L}_r$ should be charged entirely to r, since it is the only receiver in its multicast group that uses those links. It is then easy to see that $p_r^{(n)}$, as defined in (5), is the sum of the penalties over all links that receiver r uses (the set of links from the source to the receiver). Note that $p_r^{(n)}$ is zero if none of the links that receiver r uses is congested.

Now we state the update procedure for the receiver rates. In the update procedure stated below, $[\ \cdot\ ]_{X_r}$ denotes a projection[4] on the set X_r. For each $r \in R$, x_r is updated as follows

$$x_r^{(n+1)} \quad = \quad [\ x_r^{(n)} + \lambda_n\ (\ U_r'(x_r^{(n)}) - K p_r^{(n)}\ )\ ]_{X_r} \tag{6}$$

where K (the "penalty scaling factor") is a positive constant, and λ_n is the step-size at the nth iterative step.[5]

C. Conditions on the splitting factors

For our algorithm to work correctly, at every step n, the splitting factors α_r must satisfy the following conditions

$$\alpha_r^{(n)} \quad \geq \quad 0 \qquad\qquad \forall\ r \in \tilde{R} \setminus \tilde{R}_\Omega \tag{7}$$

$$\sum_{r' \in C_r} \alpha_{r'}^{(n)} \quad = \quad 1 \qquad\qquad \forall\ r \in \hat{R} \tag{8}$$

$$\alpha_r^{(n)} \quad = \quad 0 \quad \text{if } x_r^{(n)} < x_{\pi_r}^{(n)} \qquad \forall\ r \in \tilde{R} \setminus \tilde{R}_\Omega \tag{9}$$

(7) states that splitting factors are non-negative. (8) states that the sum of the splitting factors of all of the children of a junction node must add up to 1. (9) states that the splitting factor of a node is zero if it is not receiving the same rate as its parent. Since the rate of the parent node is the maximum of the rates of its children, this implies that the splitting factor of a node is zero if its rate is not the maximum amongst the rates of all of its sibling nodes. In other words, the penalty at a node is split amongst only those children who are receiving the maximum rates.

Note that the above constraints allows us to have both fractional and integral (0 and 1) splitting factors. Choosing fractional splitting factors, however, has certain drawbacks in terms of practical implementation, as we will discuss in Section VI. Therefore, in the rest of the paper, we will only be concerned with integral splitting factors. In that case, (7)

[4]Since $X_r = [b_r, B_r]$, thus for any scalar y, $[y]_{X_r} = \min(B_r, \max(b_r, y))$.

[5]Note that the rate update procedure in (6) inherently assumes that the function U_r is differentiable in X_r. This, in general, is not necessary. If $U_r'(x_r)$ does not exist as some point $x_r \in X_r$, it can be replaced by a subgradient of U_r at x_r.

is replaced by the following constraint:

$$\alpha_r^{(n)} \quad \in \quad \{0,1\} \qquad \forall\, r \in \tilde{R} \setminus \tilde{R}_\Omega \tag{10}$$

V. Convergence Analysis

In this section, we investigate the convergence of the iterative algorithm outlined in the last section. The convergence analysis presented here assume that the splitting factors satisfy (8)-(10). The results hold even if the splitting factors satisfy the more general conditions (7)-(9). However, the analysis in that case is more complex and space-consuming, and can be found in [9].

In the following, let $x = (x_r, r \in R)$ denote the vector of the receiver rates. Let X_R denote the entire region in the $|R|$-dimensional space where x is constrained to lie due to (2), i.e., $X_R = \{(x_1, ..., x_{|R|}) : x_r \in X_r \ \ \forall r \in R\}$. Thus the set of constraints in (2) can be equivalently written as $x \in X_R$. Also, let X^* be the set of optimal solutions of $\mathbf{P}$ (note that the optimal solution can be non-unique).

A. Assumptions

In the convergence analysis, we will make the following assumptions on the problem $\mathbf{P}$.

Assumption 1: (Interior point) There exists a vector $\breve{x} \in X_R$ such that $\sum_{m \in M} \max_{r \in S_l \cap R_m} \breve{x}_r < c_l$ for all $l \in L$.

Note that the above assumption also implies that the problem $\mathbf{P}$ is feasible, i.e., it has a solution. Also note that in the special case when $b_r = 0\ \forall r \in R$, $c_l > 0\ \forall l \in L$, the interior point assumption is satisfied.

Assumption 2: (Bounded slope) There exists an $A < \infty$ such that $U_r'(x_r) \leq A \quad \forall x_r \in X_r$ for all $r \in R$. [6]

Define the overall user utility function $U : \Re_+^{|R|} \to \Re$ as $U(x) = \sum_{r \in R} U_r(x_r)$, and U^* be the corresponding optimal value. Thus $U^* = U(x^*)$ for any $x^* \in X^*$. Also let $\rho(x, Y) = \min_{y \in Y} ||x - y||$ denote the Euclidean distance of a point x from any compact set Y. Now we state some convergence results under various conditions of the step-sizes. In all of the convergence results stated below, we assume that the initial receiver rates are feasible.

B. Exact convergence with diminishing step-sizes

Assume that the sequence of step-sizes $\{\lambda_n\}$ in (6) satisfies the following criteria

$$\lim_{n \to \infty} \lambda_n = 0 \qquad \sum_{n=1}^{\infty} \lambda_n = \infty \tag{11}$$

As an example, $\lambda_n = (1/n)$ is a sequence that satisfies (11).

The following theorem shows that our algorithm converges to the optimum if the step-sizes satisfy the above condition.

Theorem 1: Consider the iterative procedure stated in (3)-(6), with the splitting factors satisfying (8)-(10), and the step-sizes satisfying (11). Then there exists a $\tilde{K}$ satisfying $0 \leq \tilde{K} < \infty$, such that for all $K > \tilde{K}$,

$$\lim_{n \to \infty} \rho(x^{(n)}, X^*) = 0$$

The above theorem is proved in Appendix II. Note that from the continuity of U it follows that $\lim_{n \to \infty} U(x^{(n)}) = U^*$.

[6] If U_r is non-differentiable in X_r, we will assume that Assumption 2 holds for all subgradients of U_r in X_r.

C. Convergence and parameter choices

Theorem 1 states that there is a minimum value of the penalty scaling factor K beyond which our algorithm converges to the optimum set of solutions. Various upper-bounds of this minimum value can be calculated from the parameters of the problem **P** (see Appendix II). For example if we define $B = \max_{r \in R}(B_r - b_r)$ and $c = \min_{l \in L}(c_l - \sum_{m \in M} \max_{r \in S_l \cap R_m} b_r)$ (note that this is positive by the interior point assumption), then $(|R|AB)/c$ is an upper-bound on $\tilde{K}$ (although it may not be tight).

As one would intuitively expect, is also possible to prove the convergence of our algorithm with the fixed parameter K being replaced by a sequence K_n such that $\lim_{n\to\infty} K_n = \infty$. If we ensure that $\lambda_n K_n \to 0$, then the unicast case of this version of our algorithm is similar in certain aspects with the unicast algorithm presented in [7]. There are certain differences, however, and these allow us to derive convergence results under weaker assumptions for our algorithm, as compared to the unicast algorithm in [7].

Our algorithm can be also be shown to converge if the step-sizes λ_n satisfy $\lambda_n = \Lambda\lambda^n$ where $0 < \lambda < 1$ and Λ is a "sufficiently large" constant. Note that all these step-sizes satisfy $\lim_{n\to\infty} \lambda_n = 0$. This condition is required due to the non-differentiability of the problem. In practice, it may not be possible (or efficient) to decrease the step-size beyond a certain value. In the next subsection, therefore, we investigate the convergence of our algorithm with constant step-sizes.

D. Approximate convergence with constant step-sizes

If the step-sizes do not tend to zero, we can not guarantee convergence in the sense stated in Theorem 1. However, when the step-sizes are constant, we can prove a slightly weaker convergence result, as we state below. A similar result holds even in the case where the step-sizes are not constant but converge to some positive value. Let $\Phi_\delta(X^*)$ be the set of all points at a distance of δ or less from X^*, i.e., $\Phi_\delta(X^*) = \{x : \rho(x, X^*) \leq \delta\}$.

Theorem 2: Consider the iterative procedure stated in (3)-(6), with the splitting factors satisfying (8)-(10), and the step-sizes satisfying $\lambda_n = \lambda$ for all n. Then given any $\delta > 0$, there exists $\tilde{\lambda}_\delta > 0$ and $\tilde{K}_\delta < \infty$, such that for any λ, K satisfying $0 < \lambda < \tilde{\lambda}_\delta$ and $K > \tilde{K}_\delta$,

$$\lim_{n\to\infty} \rho(x^{(n)}, \Phi_\delta(X^*)) = 0$$

The above theorem can be proved along the same lines as Theorem 1, and omitted in this paper due to space constraints. The proof can be found in [9]. The theorem implies that given any neighborhood around the optimum, we can choose the step-size λ sufficiently small and the penalty scaling factor K sufficiently large so that our algorithm (with constant step-sizes) converges to that neighborhood. For a given δ, the values of $\tilde{\lambda}_\delta$, $\tilde{K}_\delta$ can be calculated in terms of the parameters of the problem **P** (see [9]).

VI. Distributed Implementation

Now we describe how the algorithm described in Section IV can be implemented in an asynchronous network environment in a distributed and scalable way .

A. Protocol Description

First we will describe how the protocol works. As mentioned before, in our algorithms, a source/junction/receiver node needs to communicate only with its parent and children nodes. Assume that the each source/junction node sends congestion packets (CP) (containing the congestion penalty information) to its children nodes. Also assume that each receiver/junction node sends rate packets (RP) (containing the rate information) to its parent node. Thus the CPs move in the downstream direction of the tree, while the RPs move in the upstream direction (see Figure 2). The CPs that a junction node sends to its children are sent out when the junction node receives a CP from its parent. Moreover, the RP that a junction node sends to its parent is formed by merging the RPs that it receives from all of its children. As in the figure, each CP contains a congestion penalty field $\overline{p}$, while each RP contains a rate field $\overline{x}$.

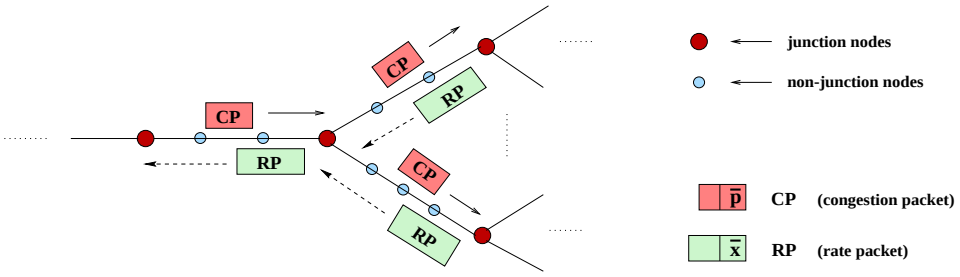

Fig. 2. Message exchanges

A junction/receiver node communicates its rate request to its parent node through the $\overline{x}$ field of the RP. This is to let the parent node know at what rate it needs to send traffic to the corresponding child. The parent node also uses these communicated rates to determine which of the children are requesting the maximum rates, and penalize only those children. For this purpose, each junction node r maintains $C_r^{\max}(\subseteq C_r)$, the set of the children requesting the maximum rates.

A source/junction node conveys the appropriate congestion penalty to its children nodes through the $\overline{p}$ field of the CP. Note that choosing fractional splitting factors makes the penalty term fractional, and this makes it difficult to convey it to the receiver using a few bytes, without sacrificing precision. For good precision, we require that the $\overline{p}$ field be fairly large, and this results in a high protocol overhead. To avoid this problem, we can just assign integral splitting factors, i.e., 0 and 1. In this case, conditions (8)-(9) require that a splitting factor of 1 be assigned to *any* one of the children that is requesting the maximum traffic rate, while a splitting factor of 0 be assigned to all other children (whether they are requesting the maximum traffic rate or not). Note that it does not matter which one of the children (amongst those that request the maximum rate) is chosen to pay the penalty, and the child that is penalized could be different at different times (iterations). The algorithms described below assume this kind of penalty splitting. This ensures that the number of bytes that need to be allocated to the $\overline{p}$ field is small (we discuss more on this later).

Also assume that link l (i.e., the node associated with link l, which is usually the node where the link originates) is responsible for keeping track of the link congestion indicator variable $\hat{q}_l$. Moreover, for any receiver/junction node r, the node itself is responsible for keeping track of the receiver/junction rate x_r.

B. Link and Node Algorithms

On receiving RPs from all of its children nodes, a junction node computes the maximum of the rates requested, and sends an RP to its parent, setting the $\overline{x}$ field to this maximum requested rate. When an RP is going through link l, the node reads the field $\overline{x}$ and uses it to update the congestion indicator $\tilde{q}$ (see the link algorithm below).

When sending a CP to a child, a source node stamps 0 in the $\overline{p}$ field of the CP. Each link on the path to the child adds the link congestion indicator (0 or 1) in the $\overline{p}$ field of the CP. A junction node transfers the $\overline{p}$ field of the CP that it receives from its parent node to the CP of one of the children that has requested the maximum rate; the $\overline{p}$ fields of the CPs for the rest of the children are stamped as 0. Thus when a receiver node receives a CP, the $\overline{p}$ field contains the appropriate congestion penalty for that receiver, which it uses for updating its rate according to (6).

In the algorithms stated below, the updates of the rates, congestion indicators etc., are triggered by arrival of a CP or an RP. In practice, however, these updates can also be done after some fixed time-intervals.

In the algorithms described below, the step-size for rate updates is kept constant at λ.

Link l's algorithm:

On receiving an RP:

1. Read the $\overline{x}$ field to know the updated rate for the session, and update the link congestion indicator ε_l as

$$\varepsilon_l \leftarrow \begin{cases} 1 & \text{if} \quad \sum_{r \in \tilde{S}_l} x_r > c_l \\ 0 & \text{if} \quad \sum_{r \in \tilde{S}_l} x_r \leq c_l \end{cases}$$

and forward the RP on to the next link.

On receiving a CP:

1. Add ε_l to the $\overline{p}$ field of the CP and forward it on to the next link.

Source node s's algorithm:

On receiving an RP:

1. Read the $\overline{x}$ field to know the new rate requested by the child.

2. Send a CP to that child, setting the $\overline{p}$ field to 0.

Receiver node r's algorithm:

On receiving a CP:

1. Read the $\overline{p}$ field of the CP to know the updated congestion penalty p_r, and calculate the new receiver rate as:

$$x_r \leftarrow x_r + \lambda U_r'(x_r) - \lambda K p_r$$

If $x_r < b_r$, set $x_r \leftarrow b_r$, and if $x_r > B_r$, set $x_r \leftarrow B_r$.

2. Send an RP to the parent node, setting the field $\overline{x}$ to x_r.

Junction node r's algorithm:

On receiving a CP:

1. Send one CP to each of the children nodes, setting the $\overline{p}$ field as follows:

(a) Pick any child node in $C_r^{\max}$, and set the $\overline{p}$ field of its CP to the $\overline{p}$ field of the CP received from the parent node.

(b) Set the $\overline{p}$ fields of the CPs of all other children nodes to 0.

On receiving an RP:

1. Read the $\overline{x}$ field to know the new rate requested by the child, and do the following:

(a) Update the junction rate x_r as: $x_r \leftarrow \max_{r' \in C_r} x_{r'}$.

(b) Update $C_r^{\max}$ as: $C_r^{\max} \leftarrow \{r' : x_{r'} = x_r\}$.

2. On receiving RPs from all of the children nodes, send an RP to the parent node, setting the $\overline{x}$ field to x_r.

C. Implementation Issues

Let us calculate the number of bits that must be allocated to the $\overline{p}$ field of the CP. Firstly note that the value of in the $\overline{p}$ field of a CP to a receiver can at most be equal to the number of links on the path from the source to the receiver. This is due to the fact that worst case, all the links from the source to a receiver can be congested, and the penalty splitting factors of all the junction/receiver nodes on that path could be one. Thus the value of in the $\overline{p}$ field of any CP can be at most $L^{\max}$, where $L^{\max}$ is the maximum number of links on the path from a source to the receiver. This implies that we need to allocate $\lfloor \log_2 L^{\max} \rfloor + 1$ to the $\overline{p}$ field. Therefore for most real networks, including the internet, allocating just one byte for the congestion penalty field should be sufficient (note that one byte would allow 255 links on a path from the source to the receiver). Thus the overhead of the network congestion feedback to the receivers is quite small.

The implementation of the link algorithm, as described in the previous subsection, has an important drawback. Note that in the implementation described, the link would have to keep track of the rates of all individual sessions that traverse that link. This implies that a router would have to maintain per-session state even for sessions for which it is a non-junction node. This is certainly undesirable, as we have argued in Section III-B. However, note that the session rates can also be estimated by traffic measurement at the links. Also note that in order to determine whether a link is congested or not, we only need to know the *total* rate of traffic at that link, and not the individual session rates (see the link algorithm described above). Thus we could determine the value of the link congestion indicator just by measuring the total arrival rate at the link. Thus with measurement-based rate estimation, maintaining of per-flow state at the links is not necessary. It is easy to see that the with this modification, the distributed implementation of our algorithm (as described in the previous subsection) satisfies all the desirable features listed in Section III-B.

VII. Simulation Results

In Section V, we have analytically proved the convergence of our algorithm in a synchronous environment. Simulations carried out on various network topologies confirm that our algorithm achieves the optimal rates even in an asynchronous slowly time-varying environment. In this section, we present a few representative examples to demonstrate this fact. In these experiments, the algorithms are implemented as described in the previous section, and the step-size of rate updates (λ) is kept fixed. However, updates of the link congestion indicators occur at regular intervals, and not on arrivals of RPs.

Figure 3 shows the example network that we consider, which consists of two multicast groups sharing a 10-link network.

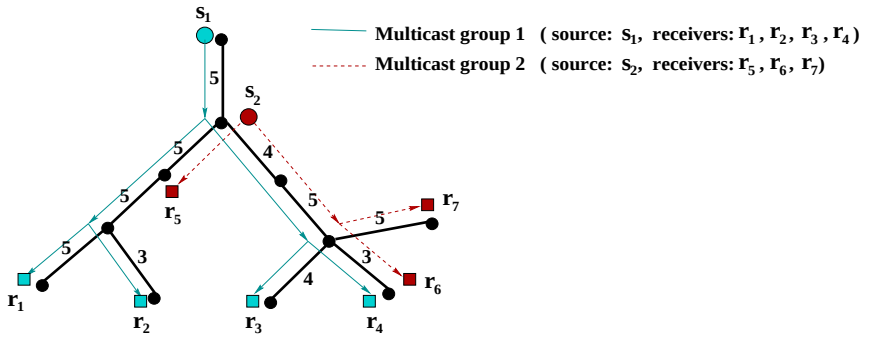

Fig. 3. An example network (The numbers associated with the links are the link capacities (in Mbps). The propagation delay for each link is 1 ms.)

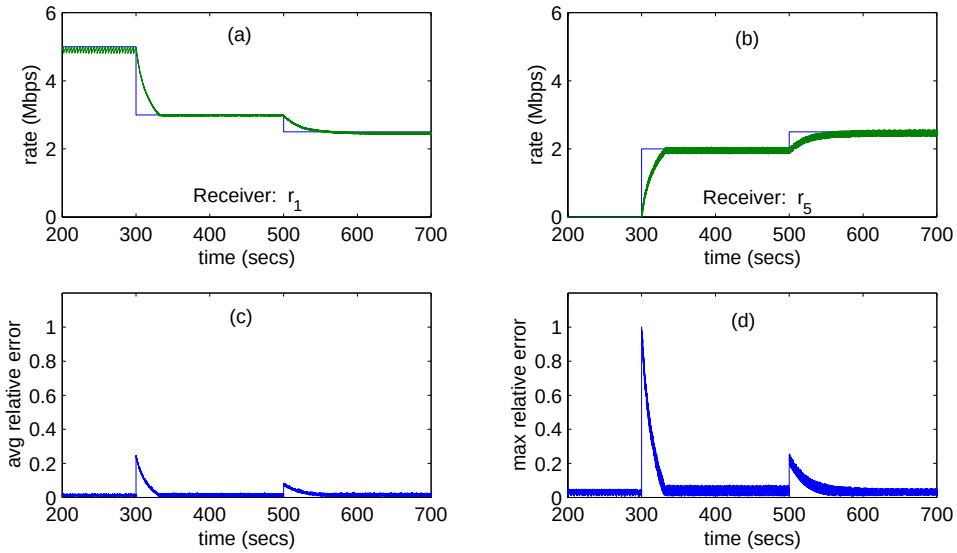

Fig. 4. Convergence of achieved rates. (The straight lines in (a) and (b) are the optimal (theoretical) rates.)

The utility functions of all receivers of group 1 and r_5 of group 2 are $\ln(1+x)$, while those of the rest are $2\ln(1+x)$. The minimum and maximum receiver rates are 0 and 5 Mbps respectively. Assume that receivers r_1, r_2, r_3, r_4, r_6 and r_7 arrive at time $t = 0$. Also, receiver r_5 joins at $t = 300$ secs, while receiver r_2 leaves at $t = 500$ secs. Figure 4, which shows some rate plots in the time window 200-700 secs, demonstrate the performance of our algorithm in this particular example. The step-size of rate updates, λ is 0.0025 and K is set to 1.6. The update intervals of the link congestion indicators is 0.1 sec. Note that the receivers that joined at time $t = 0$ have already reached their steady-state (optimal) rates at $t = 200$ secs. Figure 4(a) and (b) show the (achieved) receiver rates of r_1 and r_5 along with the optimal rates (the curves of the other receiver rates also exhibit a similar trend). Figure 4(a) shows that the observed rate of r_1 tracks the optimal rate closely even as the optimal rate changes (due to the arrival/departure of other receivers). Figure 4(b) also shows the same trend. Figure 4(c) and (d) show the average and maximum relative errors over all receivers. If $x_r^a(t)$ and $x_r^o(t)$ respectively denote the achieved and optimal rates of receiver r at time t, the relative error for receiver r at time t is defined as $|1 - \frac{x_r^a(t)}{x_r^o(t)}|$. The sharp peaks exhibited by the curves at $t = 300$ and $t = 500$ secs are due to the sudden change in the optimal rates due to the arrival/departure of receivers. The relative errors decrease with time and gradually approach zero, indicating the convergence of all receiver rates to the optimal values.

As a careful look at Figure 4(c) and (d) shows, the average and maximum relative errors do not exactly reach zero,

but fluctuate rapidly, remaining close to zero. The thickening of the receiver rate plots in Figure 4(a) and (b) are also due to these small but rapid fluctuations around the optimal values. Recall that in Section V, we argued that due to the non-differentiability of the problem we need step-sizes close to zero in order to guarantee exact convergence. When the step-size is constant, but small, as in the case of the plots in Figure 4, then we can only guarantee that the rates would converge to a neighborhood of the optimum (Theorem 2). When the total traffic is close to the link capacity, the link congestion indicator fluctuates rapidly between 0 and 1, as can be expected from intuition. Moreover, when multiple children request the maximum traffic rate from a junction node, the penalty splitting factors for those children could also fluctuate rapidly, as can be expected from the description of the junction node's algorithm presented in the last section. This could cause the receiver penalty p_r to fluctuate rapidly, causing rate fluctuations like those seen in Figure 4. Smaller step-sizes cause smaller fluctuations, but also results in lower convergence speeds. Thus the choice of the step-size is a tradeoff between the convergence speed and magnitude of fluctuations. In practice, a receiver could choose large step-sizes initially (to ensure fast convergence), and reduce the step-sizes once it detects that its rate is fluctuating around the same mean value (to reduce fluctuations when the rates are close to the optimal values). Recall that convergence is guaranteed only when the constant K is "sufficiently large" (Theorems1 and 2). However, setting K to a very large value could reduce the average throughput considerably, as we would intuitively expect. Therefore, the value of K should be chosen carefully to ensure good performance in practice.

VIII. Related Work

In this section, we mention some of the recent work on the unicast version of the problem that we have addressed in this paper. We also compare, in detail, the algorithm presented in this paper with an alternative algorithm for the same problem (the multirate multicast case), presented in [8].

An aggregate utility maximization approach to flow control was suggested recently by Kelly [10]. This problem, for the case of all unicast sessions, has received considerable attention in recent literature. Several flow control algorithms, both rate-based and window-based, have been proposed (see [14], [11], [13], [25], [7]). These algorithms were derived using different optimization approaches, and we will not discuss them here. Amongst these, the unicast algorithm presented in [7] is also based on subgradient optimization methods. However, the algorithm in this paper is derived using a different approach than that in [7], and this allows us to handle the multirate multicast problem effectively. For the special case of all unicast sessions, the algorithm presented in this paper reduces to a form that has certain similarities with the algorithm in [7] (particularly the fact that in both cases, the congestion feedback from the network to the user is the number of congested links on user's path). However, compared to the the algorithm in [7], the all unicast version of our algorithm guarantees convergence under weaker assumptions on the receiver utility functions and the penalty scaling factor.

For the case of multirate multicast sessions, the optimization based rate control problem has not been adequately addressed. As we have already argued in earlier sections, the non-differentiability of the problem and the multicast-specific requirements make this problem much more complex than its unicast equivalent. In [8], the authors address the multirate multicast utility maximization problem and propose dual-based algorithm for it. The algorithm is distributed, and does not require the network to know the receiver utilities. The processing, storage and communication overheads at a junction node is proportional to its number of children. In spite of these attractive features, the algorithm in [8] suffers

from certain drawbacks which limits its practical viability.

The most important limitation of the algorithm in [8] is that it requires per-session information to be maintained even at the non-junction nodes. In a network, we would expect that most of the sessions would be unicast. Even for multicast sessions, we can expect that most of the nodes in the multicast tree would be non-junction nodes. Therefore maintaining these per-flow states could be a huge overhead for the routers. Also note that in the algorithm in [8], the network determines its congestion level based on certain "pseudo-rates" which could be different from the actual rates. The pseudo-rates can not be inferred from the actual traffic rates. Therefore, these pseudo-rates need to be stored at junction nodes, and also to be communicated between a parent and children nodes, thus increasing the storage and communication overheads significantly. As we have argued before, in our algorithm, no per-session information needs to be maintained at the non-junction nodes. Moreover, we do not have any extra overhead of storing and communicating pseudo-rates.

In the algorithm proposed in [8] the congestion information ("congestion prices") that the network needs to communicate to the users are real numbers that could vary over a wide range. This poses a difficulty in communicating the price to the end-host using a small number of bits. While one can use some probabilistic marking policies (following the approach in [1]) to convey the congestion information is a single bit, it is not clear if such policies can provide theoretical convergence guarantees (note that even if the algorithm converges in that case, the convergence would be in some probabilistic sense). On the other hand, our algorithm has guaranteed deterministic convergence, and would require, in practice, no more than one byte in the packet header for conveying the congestion information.

In terms of computational overhead too, our algorithm is better than the one proposed in [8]. Firstly, the junction node algorithm is considerably simpler in our case as compared to the latter. Moreover, in certain cases, the receiver algorithm too could be much simpler in our case as compared to that in [8] (note that the algorithm in [8] requires the receiver to compute a maximizer, whereas in our case, the receiver only needs to compute a derivative).

It is also worth noting here that the algorithm presented here guarantees convergence for a wider class of utility functions as compared to the algorithm in [8]. Our algorithm guarantees convergence for linear utility functions (note that unlike the approach in [8], which requires strict concavity of utility functions, we require only concavity), and also a wide range of non-differentiable utility functions, which is outside the framework of the algorithm in [8].

IX. Concluding Remarks and Future Work

Note that in the problem discussed in this paper, we have assumed that bandwidth allocations can be continuous. In reality, however, the bandwidth allocations could be limited to some discrete levels only (for example, in layered video, there will be a few distinct bandwidth levels, one corresponding to each layer). This makes the problem much harder to solve (it becomes an integer programming problem). However, in that case, we can use the approach presented in this paper to obtain an approximate solution. We can "convexify" the problem first, and then apply our algorithms to the convexified problem (it is like solving the linear programming relaxation of the integer programming problem). If the set of discrete bandwidth levels are dense, then the rates obtained by rounding down the rates to which our algorithms converge would give rates that are both feasible and close to optimality.

There are several interesting and challenging questions related to this work that need to be investigated further. Note that in this paper we have only proved the convergence of our algorithm under the synchronous update assumption.

An interesting theoretical question is that of investigating the convergence of the algorithm in the case of asynchronous updates. Also note that the algorithm for which the convergence was proved assumed that the feedback that the receiver and the network provides to each other are exact, i.e., devoid of any errors. In practice, however, there will be errors due to measurement, rounding etc. Some of these errors could be modeled as noise. The performance of our algorithm under noise is another topic that requires further investigation. As already mentioned, in order to avoid maintaining per-flow state at the routers, the routers need to estimate the total rate by measurement. Convergence of our algorithm in such a case (i.e., where rates are measured and not communicated) needs to be investigated further. In particular, determining the class of link scheduling policies for which convergence of our algorithm (with estimated rates) can be guaranteed is a very interesting and important question, both from theoretical and practical perspectives.

REFERENCES

[1] S. Athuraliya, S. Low, D. Lapsley,"Random Early Marking", Submitted for publication, www.ee.mu.oz.au/staff/slow/research/

[2] D. P. Bertsekas, "Necessary and Sufficient Conditions for a Penalty Method to be Exact", *Math. Programming 9*, pp. 87-99, 1975.

[3] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1995.

[4] D. P. Bertsekas, R. Gallagher, *Data Networks*, Prentice Hall, Second Edition, 1992.

[5] D. P. Bertsekas, J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, 1989.

[6] T. Bially, B. Gold, and S. Seneff, "A technique for Adaptive Voice Flow Control in Integrated Packet Networks", *IEEE Transactions on Communications*, vol. 28, no. 3, March 1980, pp. 325-333.

[7] Omitted for anonymity.

[8] Omitted for anonymity.

[9] Technical Report. Omitted for anonymity.

[10] F. P. Kelly,"Charging and Rate Control for Elastic Traffic", *European Transactions on Telecommunications*, vol. 8, no. 1, 1997, pp. 33-37.

[11] F. Kelly, A. Maulloo, D. Tan,"Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability ", *Journal of Operations Research Society*, vol. 49, no. 3, 1998, pp. 237-252.

[12] 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*, vol. 7, no. 5, June 1989.

[13] R. La, V. Anantharam,"Charge-Sensitive TCP and Rate Control in the Internet", *Proceedings of Infocom 2000*, March 2000.

[14] S. Low, D. E. Lapsley, "Optimization Flow Control, I: Basic Algorithm and Convergence", *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, December 1999.

[15] X. Li, S. Paul and M. Ammar, "Layered Video Multicast with Retransmission (LVMR): Evaluation of Hierarchical rate control", *Proceedings of IEEE Infocom' 98*, March 1998.

[16] L. Massoulie and J. Roberts, "Bandwidth Sharing: Objectives and Algorithms", *Proceedings of IEEE Infocom 1999*, New York, USA, March 2000.

[17] S. McCanne, V. Jacobson and M. Vetterli, "Receiver-Driven Layered Multicast", *Proceedings of ACM Sigcomm '96*, Stanford, CA, September 1996.

[18] B. T. Poljak, "A General Method of Solving Extremum Problems", *Soviet Math Doklady*, vol. 8, no. 3, 1967, pp. 593-597.

[19] R. T. Rockafellar, *Convex Analysis*, Princeton Univ. Press, 1970.

[20] D. Rubenstein, J. Kurose and D. Towsley. "The Impact of Multicast Layering on Network Fairness", *Proceedings of ACM Sigcomm '99, Cambridge, MA, September, 1999*.

[21] N. Z. Shor, *Minimization Methods for Non-differentiable Functions*, Springer-Verlag, 1985.

[22] S. Sarkar and L. Tassiulas, "Fair Allocation of Utilities in Multirate Multicast Networks", *Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing*, 1999.

[23] S. Sarkar and L. Tassiulas, "Distributed Algorithms for Computation of Fair Rates in Multirate Multicast Trees", *Proceedings of IEEE Infocom 2000*, Tel Aviv, Israel, March 2000.

[24] S. Sarkar and L. Tassiulas, "Fair Allocation of Discrete Bandwidth Layers in Multicast Networks", *Proceedings of IEEE Infocom 2000*, Tel Aviv, Israel, March 2000.

[25] S. Kunniyur, R. Srikant,"End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks", *Proceedings of Infocom 2000*, March 2000.

[26] T. Turletti and J. C. Bolot, "Issues with Multicast Video Distribution in Heterogeneous Packet Networks", *Packet Video Workshop*, '94.

APPENDIX I: SUBGRADIENTS AND THEIR PROPERTIES

Definition 1: [21] *(Subgradient and Subdifferential*) Consider a convex and continuous function f defined on a convex set $F \subseteq \Re^k$. Then a vector $w_0 \in \Re^k$ is called a *subgradient* of f at a point $x_0 \in F$ if it satisfies

$$f(x) - f(x_0) \quad \geq \quad (w_0, x - x_0) \qquad \forall x \in F$$

The *subdifferential* of f at $x_0 \in F$, denoted by $\partial f(x_0)$, is the set of all subgradients of f at x_0, i.e.,

$$\partial f(x_0) \quad = \quad \{w_0 \in \Re^k : f(x) - f(x_0) \ \geq \ (w_0, x - x_0) \ \ \forall x \in F\}$$

In general, subgradient at a point may be non-unique. However, if $\nabla f(x_0)$ exists, then $\partial f(x_0) = \{\nabla f(x_0)\}$.

Next we state two properties of subgradients (see Theorems 1.12 & 1.13 of [21]), which will be useful in our analysis.

Lemma 1: Let I be a finite index set. Let $f_i, i \in I$, be convex, continuous functions defined on a convex set F. Let $x_0 \in F$, and $w_{i0} \in \partial f_i(x_0)$, $i \in I$.

(a) Let $f(x) = \sum_{i \in I} a_i f_i(x)$, where $a_i \geq 0$, $i \in I$. Then $\sum_{i \in I} a_i w_{i0} \in \partial f(x_0)$.

(b) Let $f(x) = \max_{i \in I} f_i(x)$. Define $\tilde{I}(x) = \{i \in I : f_i(x) = f(x)\}$. Then $w_{i0} \in \partial f(x_0)$, for all $i \in \tilde{I}(x_0)$.

APPENDIX II: PROOF OF THEOREM 1

We will first state a few lemmas that would be used in the proof of Theorem 1. For each $l \in L$, define $g_l : \Re_+^{|R|} \rightarrow \Re$ as $g_l(x) = \max_{r \in S_l \cap R_m} x_r - c_l \ = \ \sum_{r \in \tilde{S}_l} x_r - c_l$. Thus the capacity constraint for link l can be simply written as $g_l(x) \leq 0$. In the following, let $\{\mu_l^*, l \in L\}$ be the set of Lagrange multipliers for problem $\mathbf{P}$. Let $\mu_{\max}^* = \max_{l \in L} \mu_l^*$. Then it is easy to prove the following result (see [3] (pp. 450)):

Lemma 2: Consider a vector $\check{x} \in X_R$ such that $g_l(\check{x}) < 0$ for all $l \in L$. Then

$$\mu_{\max}^* \quad \leq \quad \frac{U^* - U(\check{x})}{\min_{l \in L}\{-g_l(\check{x})\}}$$

The above lemma shows that the lagrange multipliers are bounded. Note that the existence of a vector $\check{x}$ satisfying the required conditions is guaranteed by the Assumption 1.

Now consider the following problem

$$\tilde{\mathbf{P}}: \qquad \text{maximize} \quad \sum_{r \in R} U_r(x_r) - K \sum_{l \in L} \max\{0, g_l(x)\} \ , \qquad \text{subject to} \quad x_r \in X_r \quad \forall r \in R$$

where K is a non-negative constant.

Let $\tilde{X}^*$ denote the set of optimal solutions of $\tilde{\mathbf{P}}$. The following lemma follows as a special case of Proposition 1 of [2]:

Lemma 3: If $K > \tilde{K} = \mu_{\max}^*$, then $\tilde{X}^*$ coincides with X^*.

The above result is fairly intuitive. Comparing problems $\mathbf{P}$ and $\tilde{\mathbf{P}}$, we see that the link constraints in $\mathbf{P}$ have been transferred to the objective function in $\tilde{\mathbf{P}}$. The term $K\max\{0, g_l(x)\}$ can be interpreted as the penalty associated with the violation of the capacity constraint of link l. Thus the above lemma states that when the penalty associated with constraint violations is sufficiently large, the optimal solution set of the unconstrained problem $\tilde{\mathbf{P}}$ becomes the same as that of $\mathbf{P}$. Note that we can use Lemma 2 to compute upper bounds on $\tilde{K}$.

Now define a function $\tilde{U} : \Re_+^{|R|} \rightarrow \Re$ as $\tilde{U}(x) = \sum_{r\in R} U_r(x_r) - K\sum_{l\in L}\max\{0, g_l(x)\}$. Thus $\tilde{\mathbf{P}}$ is the problem of maximizing $\tilde{U}(x)$ subject to $x \in X_R$. Let $u_r^{(n)} = U_r'(x_r^{(n)})$, and $v_r^{(n)} = u_r^{(n)} - Kp_r^{(n)}$. Let $u^{(n)} = (u_r^{(n)}, r \in R)$ denote the vector of the utility derivatives, and $p^{(n)} = (p_r^{(n)}, r \in R)$ denote the vector of the congestion penalties. Let $v^{(n)} = (v_r^{(n)}, r \in R) = u^{(n)} - Kp^{(n)}$. Note that since $p^{(n)}$ depends on the penalty splitting factors $\alpha_r^{(n)}, r \in \tilde{R} \setminus \tilde{R}_\Omega$ (cf. (5)), so does $v^{(n)}$.

Lemma 4: If $\alpha_r^{(n)}, r \in \tilde{R} \setminus \tilde{R}_\Omega$ satisfy (8)-(10), then $v^{(n)} \in \partial\tilde{U}(x^{(n)})$.

Proof: (Outline) Define $P_l(x) = \max\{0, g_l(x)\}$, and $P(x) = \sum_{l\in L} P_l(x)$. Therefore, $\tilde{U}(x) = U(x) - KP(x)$.

We will first show that $p^{(n)} \in \partial P(x^{(n)})$. Define $g_{lm}(x) = \max_{S_l\cap R_m} x_r$. Then $g_l(x) = \sum_{m\in M} g_{lm}(x)$. Now for every $l \in L$, let $\beta_l^{(n)} = (\beta_{l,r}^{(n)}, r \in R)$ be a $|R|$-dimensional vector whose components are given as

$$\beta_{l,r}^{(n)} = \begin{cases} 0 & \text{if} \quad r \notin S_l \\ \prod_{r''\in Q_r\setminus Q_{r'}} \alpha_{r''}^{(n)} & \text{if} \quad r \in S_l \end{cases} \tag{12}$$

where r' is junction/receiver node immediately downstream of link l in the multicast tree to which r belongs. Note that since the splitting factors are either 0 or 1, the components $\beta_{l,r}^{(n)}$ and are also either 0 or 1. Then combining (5) and (4), it is easy to see that $p^{(n)}$ can be written as

$$p^{(n)} = \sum_{l\in L} \beta_l^{(n)}\varepsilon_l^{(n)} \tag{13}$$

Now for every $l \in L$ and every $m \in M$, let $\tilde{\beta}_{lm}^{(n)} = (\tilde{\beta}_{lm,r}^{(n)}, r \in R)$ be a $|R|$-dimensional vector whose components are given as

$$\tilde{\beta}_{lm,r}^{(n)} = \begin{cases} \beta_{l,r}^{(n)} & \text{if} \quad r \in R_m \\ 0 & \text{if} \quad r \notin R_m \end{cases} \tag{14}$$

Next we show that $\tilde{\beta}_{lm}^{(n)} \in \partial(\max_{r\in S_l\cap R_m} x_r^{(n)}) = \partial g_{lm}(x^{(n)})$. Consider any $l \in L$ and any $m \in M$. Now consider the two cases:

(i) $S_l \cap R_m = \phi$: In this case, it is easy to see that $\tilde{\beta}_{lm}^{(n)}$ is a zero vector. Thus $\tilde{\beta}_{lm}^{(n)} \in \partial(\max_{r\in S_l\cap R_m} x_r^{(n)})$, trivially.

(ii) $S_l \cap R_m \neq \phi$: From (8) and (10), it is easy to see that only one component of $\tilde{\beta}_{lm}^{(n)}$ is 1, and all the rest are 0. (Note that for $\tilde{\beta}_{lm,r}^{(n)}$ to be 1, the splitting factors of all the junction/receiver nodes in the downstream path from the link l to receiver r has to be 1. It is easy to see that this will happen for exactly one receiver of multicast group m). Let $r' \in R_m$ be such that $\tilde{\beta}_{lm,r'}^{(n)} = 1$. Then using (9), it is also easy to show that $x_{r'}^{(n)} = \max_{r\in S_l\cap R_m} x_r^{(n)}$. Then, from Lemma 1 (b), follows that $\tilde{\beta}_{lm}^{(n)} \in \partial(\max_{r\in S_l\cap R_m} x_r^{(n)})$.

From cases (i) and (ii), it follows that $\tilde{\beta}_{lm}^{(n)} \in \partial g_{lm}(x^{(n)})$ for all $l \in L$ and $m \in M$.

Note that $\beta_l^{(n)} = \sum_{m \in M} \tilde{\beta}_{lm}^{(n)}$. Hence, from Lemma 1 (a), it follows that $\beta_l^{(n)} \in \partial(\sum_{m \in M} g_{lm}(x^{(n)}) = \partial g_l(x^{(n)})$. Using this fact, and Lemma 1 (b), it is easy to show that $\beta_l^{(n)} \varepsilon_l^{(n)} \in \partial(\max\{0, g_l(x^{(n)})\}) = \partial P_l(x^{(n)})$. Then from (13), and using Lemma 1 (a), it follows that $p^{(n)} \in \partial(\sum_{l \in L} P_l(x^{(n)})) = \partial P(x^{(n)})$.

It is straightforward to show that $u^{(n)} \in \partial U(x^{(n)})$. Therefore, using Lemma 1 (a), $v^{(n)} = u^{(n)} - Kp^{(n)} \in \partial U(x^{(n)}) - KP(x^{(n)}) = \partial \tilde{U}(x^{(n)})$. □

Proof of Theorem 1: We will first show that $\lim_{n \to \infty} \rho(x^{(n)}, \tilde{X}^*) = 0$.

Choose an arbitrary $\delta > 0$. Let $\delta' = (\delta/2)$. For any $\epsilon' > 0$, define $D_{\epsilon'}$ as $D_{\epsilon'} = \{x : x \in X_R, \tilde{U}(x) \geq \tilde{U}^* - \epsilon'\}$. It follows from Theorem 27.2 of [19] that there exists an $\epsilon = \epsilon(\delta') > 0$ such that

$$D_\epsilon \quad \subset \quad \{x : \rho(x, \tilde{X}^*) \leq \delta'\} \tag{15}$$

Consider an n for which $x^{(n)} \notin D_\epsilon$. Therefore, $\tilde{U}(x^{(n)}) < \tilde{U}^* - \epsilon$. Choose any $\tilde{x}^* \in \tilde{X}^*$. Since $v^{(n)} \in \partial \tilde{U}(x^{(n)})$ (from Lemma 4), and using the definition of a subgradient (Definition 1), we obtain

$$(v^{(n)}, x^{(n)} - \tilde{x}^*) \quad \leq \quad \tilde{U}(x^{(n)}) - \tilde{U}(\tilde{x}^*) \quad < \quad -\epsilon \tag{16}$$

Note that $||u^{(n)}||$ is upper-bounded (from Assumption 2), and so are K and $||p^{(n)}||$. Therefore, $||v^{(n)}|| = ||u^{(n)} - Kp^{(n)}||$ is also upper-bounded. Let $||v^{(n)}|| \leq \tilde{A}$ for all n. Also note that the rate update procedure for the receiver nodes, as stated in (6), can be compactly stated as: $x^{(n+1)} = [\, x^{(n)} + \lambda_n v^{(n)} \,]_{X_R}$. Using these facts, and (16), we obtain,

$$\begin{aligned}
||x^{(n+1)} - \tilde{x}^*||^2 &= ||[\, x^{(n)} + \lambda_n v^{(n)} \,]_{X_R} - \tilde{x}^*||^2 \\
&\leq ||x^{(n)} + \lambda_n v^{(n)} - \tilde{x}^*||^2 \qquad (17) \\
&= ||x^{(n)} - \tilde{x}^*||^2 + \lambda_n^2 ||v^{(n)}||^2 + 2\lambda_n (x^{(n)} - \tilde{x}^*, v^{(n)}) \\
&< ||x^{(n)} - \tilde{x}^*||^2 + \tilde{A}^2 \lambda_n^2 - 2\epsilon\lambda_n \qquad (18)
\end{aligned}$$

Note that (17) follows from the fact that $\tilde{x}^* \in X_R$ (use projection theorem).

Since $\lambda_n \to 0$, $\lambda_n \leq (\epsilon/\tilde{A}^2)$ when n is sufficiently large. For all such n, from (18), we get

$$||x^{(n+1)} - \tilde{x}^*||^2 \quad < \quad ||x^{(n)} - \tilde{x}^*||^2 - \epsilon\lambda_n \tag{19}$$

Now, for the sake of contradiction, let us assume that there exists a $N'_\epsilon < \infty$ such that $x^{(n)} \notin D_\epsilon$ for all $n \geq N'_\epsilon$. Therefore, there exists $N_\epsilon \geq N'_\epsilon$ be such that (19) holds for all $n \geq N_\epsilon$. Summing up the inequalities obtained from (19) for $n = N_\epsilon$ to $N_\epsilon + m$, we obtain

$$||x^{(N_\epsilon + m + 1)} - \tilde{x}^*||^2 \quad < \quad ||x^{(N_\epsilon)} - \tilde{x}^*||^2 - \epsilon \sum_{n=N_\epsilon}^{N_\epsilon + m} \lambda_n \tag{20}$$

which implies that $||x^{(N_\epsilon + m + 1)} - \tilde{x}^*|| \to -\infty$ as $m \to \infty$, since $\sum \lambda_n$ diverges. This is impossible, since $||x^{(N_\epsilon + m + 1)} - \tilde{x}^*|| \geq 0$. Hence our assumption was incorrect. Hence, there exists an infinite sequence $n_{1,\epsilon} < n_{2,\epsilon} < n_{3,\epsilon} < ...$ such that $x^{(n_{i,\epsilon})} \in D_\epsilon$ for all $i = 1, 2, 3, ...$ This implies that there exists an i_1 such that (19) holds for all $n \geq n_{i_1,\epsilon}$. Also, since $\lambda_n \to 0$, there exists and i_2 such that $\lambda_n \leq (\delta'/\tilde{A})$ for all $n \geq n_{i_2,\epsilon}$.

Let $i' = \max(i_1, i_2)$. We show that $\rho(x^{(n)}, X^*) \leq \delta$ for all $n \geq n_{i',\epsilon}$. Pick any $n \geq n_{i',\epsilon}$. There can be three cases:
Case 1 : $n = n_{j,\epsilon}$ for some $j \geq i'$: In this case, $x^{(n)} \in D_\epsilon$. from (15), it trivially follows that $\rho(x^{(n)}, \tilde{X}^*) \leq \delta' < \delta$.
Case 2 : $n = n_{j,\epsilon} + 1$ for some $j \geq i'$: In this case, $x^{(n)} = x^{(n_{j,\epsilon}+1)} = [\ x^{(n_{j,\epsilon})} + \lambda_{n_{j,\epsilon}} v^{(n_{j,\epsilon})}\ ]_{X_R}$. Thus

$$\begin{aligned} ||x^{(n)} - x^{(n_{j,\epsilon})}|| &= ||[\ x^{(n_{j,\epsilon})} + \lambda_{n_{j,\epsilon}} v^{(n_{j,\epsilon})}\ ]_{X_R} - x^{(n_{j,\epsilon})}|| \\ &\leq ||x^{(n_{j,\epsilon})} + \lambda_{n_{j,\epsilon}} v^{(n_{j,\epsilon})} - x^{(n_{j,\epsilon})}|| \\ &= \lambda_{n_{j,\epsilon}} ||v^{(n_{j,\epsilon})}|| \ \leq\ \tilde{A}\lambda_{n_{j,\epsilon}} \ \leq\ \delta' \end{aligned} \tag{21}$$

From (21) and the fact that $\rho(x^{(n_{j,\epsilon})}, X^*) \leq \delta'$ (Case 1), we get

$$\rho(x^{(n)}, X^*) \ \leq\ \rho(x^{(n_{j,\epsilon})}, X^*) + ||x^{(n)} - x^{(n_{j,\epsilon})}|| \ \leq\ \delta' + \delta' \ =\ 2\delta' \ =\ \delta \tag{22}$$

Case 3 : $n_{j,\epsilon}+1 < n < n_{j+1,\epsilon}$ for some $j \geq i'$: Note that $x^{(n')} \notin D_\epsilon$ for all n' satisfying $n_{j,\epsilon} < n' < n_{j+1,\epsilon}$. From (19), it follows that $||x^{(n'+1)} - \tilde{x}^*|| < ||x^{(n')} - \tilde{x}^*||$. Summing up these inequalities obtained for $n' = n_{j,\epsilon}+1$ to $n-1$, we obtain $||x^{(n)} - \tilde{x}^*|| < ||x^{(n_{j,\epsilon}+1)} - \tilde{x}^*||$. Since this inequality holds for all $\tilde{x}^* \in \tilde{X}^*$, hence $\rho(x^{(n)}, \tilde{X}^*) < \rho(x^{(n_{j,\epsilon}+1)}, \tilde{X}^*)$. Since $\rho(x^{(n_{j,\epsilon}+1)}, \tilde{X}^*) \leq \delta$ (Case 2), it follows that $\rho(x^{(n)}, \tilde{X}^*) \leq \delta$.

From cases 1, 2, 3,, if follows that $\rho(x^{(n)}, \tilde{X}^*) \leq \delta$ for all $n \geq n_{i',\epsilon}$. By virtue of the arbitrariness of δ, it follows that $\lim_{n\to\infty} \rho(x^{(n)}, \tilde{X}^*) = 0$. Now, from Lemma 3, it follows that if $K > \tilde{K}$, then $\lim_{n\to\infty} \rho(x^{(n)}, X^*) = 0$. □

APPENDIX III: TABLE OF NOTATION

M	set of all multicast groups
R_m	set of receiver nodes in group m
R	set of all receiver nodes (over all multicast groups) ($= \cup_{m \in M} R_m$)
$\hat{R}$	set of all junction nodes (over all multicast groups)
$\tilde{R}$	set of all receiver + junction nodes (over all multicast groups) ($= R \cup \hat{R}$)
L	set of all links
S_l	set of receiver nodes using link l
$\tilde{S}_l$	set of all receiver + junction nodes that are the immediate downstream nodes of sessions going through link l
$\tilde{L}_r$	set of all links whose immediate downstream junction/receiver node is r
π_r	parent node of junction/receiver node r
C_r	set of all children nodes of junction node r
$C_r^{\max}$	set of all children nodes of junction node r which request the maximum rates
T_r	set of receiver nodes that are included in the tree rooted at junction node r.
Q_r	set of all junction + receiver nodes from the source node to r, including r but excluding the source node.
$x_r, r \in R$	rate of receiver node r
$x_r, r \in \hat{R}$	rate of junction node r ($= \max_{r' \in T_r} x_{r'}$)
x	vector of all receiver rates ($= (x_r, r \in R)$)
X_r	set of feasible rates of receiver r, defined by its max/min rate constraints ($= [b_r, B_r]$)
X_R	set of feasible receiver rate vectors, defined by the constraints in (2) ($= \{(x_1, ..., x_{\|R\|}) : x_r \in X_r \ \ \forall r \in R\}$)
X^*	set of optimal solutions of $\mathbf{P}$