Optimization Based Rate Control for Multirate Multicast Sessions

Koushik Kar† Saswati Sarkar* Leandros Tassiulas†

† ECE Department
University of Maryland
College Park, MD 20742, USA
{koushik,leandros}@eng.umd.edu

* EE Department
University of Pennsylvania
Philadelphia, PA 19104, USA
swati@ee.upenn.edu

Abstract— **Multirate multicasting, where the receivers of a multicast group can receive service at different rates, is an efficient mode of data delivery for many real-time applications. In this paper, we address the problem of achieving rates that maximize the total receiver utility for multirate multicast sessions. This problem not only takes into account the heterogeneity in user requirements, but also provides a unified framework for diverse fairness objectives. We propose two algorithms and prove that they converge to the optimal rates for this problem. The algorithms are distributed and scalable, and do not require the network to know the receiver utilities. We discuss how these algorithms can be implemented in a real network, and also demonstrate their convergence through simulation experiments.**

I. Introduction

Many present day real-time applications, like teleconferencing and audio/video broadcasting, require communication within a group, and hence multicast is the inherent mode of delivery for these applications. In conventional 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. It is therefore desirable to use multirate transmission, where the receivers of the same multicast group can receive service at different rates. 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. Multirate transmission can be attained by hierarchically encoding real time signals. In this approach, a signal is encoded into a number of layers that can be incrementally combined to provide progressive refinement. Every layer is transmitted as a separate multicast group and receivers adapt to congestion by joining and leaving these groups. Refer to [11] and [12] for internet protocols for adding and dropping layers. This layered transmission scheme have been used for both audio[4] and video[19] transmissions over the internet and has potentials for use in ATM networks as well [8]. Note that in multirate multicast transmission, there is no concept of a unique multicast session rate, and one needs to consider receiver rates separately. Also note that in this case, the rate 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).

An effective rate control strategy is required to ensure that traffic offered to a network by different traffic sources remain within the limits that the network can carry. Besides ensuring stability, the rate control strategy should ensure efficient use of the network, and also that the network resources are allocated to 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 could be many acceptable definitions of fairness. However, since receivers could have heterogeneous requirements, the same amount of bandwidth could be valued differently by different receivers (note that throughout the paper, we use the terms "receiver" and "user" synonymously). Therefore it is important to generalize the notions of fairness so that one can differentiate among receivers within the framework of fairness . This can be done by associating an utility function (assumed to be concave) with each receiver, which could be a measure of say, the perceived quality of audio/video, the user satisfaction, or even the amount paid by the receiver. In this paper, we try to design the rate control algorithms such that they maximize the sum of the utilities over all receivers, an objective that was proposed in [6]. Even if all the utility functions are the same, it can be shown that various fairness objectives can be realized within this framework for different choices of the utility functions.

Very recently, the problem of fair allocation of resources in multirate multicast networks has received considerable attention. However, most of the current research in this context is concerned only with the notion of max-min fairness (see [13], [15], [16], [17]). Amongst these, [15] considers the problem of utility allocation. Whereas we are concerned about maximizing the aggregate utility, the objective in [15] is to allocate the utilities fairly.

The problem of maximizing the aggregate utilities has not been explored in the multirate multicast context. However, several rate control algorithms that attain this objective for unicast sessions have been proposed in recent literature. In [7], both primal and dual algorithms that solve an approximate version of the actual problem, are presented. A dual algorithm that converges to the optimal solution is proposed in [10]. In these algorithms, an unicast source updates its rate based on the congestion information ("congestion price") communicated to it by the network. The network updates the congestion information based on the source rates, either communicated by the source or measured. For some other approaches in the unicast case, see [9], [18].

As already mentioned, in this paper we consider the problem of maximizing the aggregate receiver utility for the case of multirate multicast sessions. The solutions that we propose are distributed and scalable, and are practical for implementation in a large network. Like the approach used in [10] for the unicast case, our approach also uses dual methods. In fact, in the special case where all the sessions are unicast, the two algorithms that we propose reduce to the algorithm proposed in [10]. However, as we will see in the following sections of this paper, there are several factors that make the problem in our case much more complex than its unicast equivalent.[1]

The paper is organized as follows. The problem is stated formally in Section II, while the the basic solution approach is presented in Section III. We present two different algorithms in Sections IV and V, and describe their practical implementation in Section VI. We present some experimental results in Section VII and conclude in Section VIII.

II. Problem Statement

First we present a mathematical formulation of the optimization problem. We then provide an alternative formulation of the problem, which will form the basis of the solutions we propose.

A. The Optimization 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. 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 denote the set of receivers using link $l \in L$. Each receiver $r \in R$ is associated with an utility function $U_r(x_r)$, where x_r is the rate at which r receives data.[3] Let $b_r \geq 0$ and $B_r < \infty$ be the minimum and maximum rates, respectively, required by receiver r. Let $X_r = [b_r, B_r]$ denote the interval in which the receiver rate x_r must lie, and let $X = \{(x_1, ..., x_{|R|}) : x_r \in X_r \;\; \forall r \in R\}$. Let $x = (x_r, r \in R)$ be the vector of all the receiver rates.

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:

$$\mathbf{P}: \qquad \max \sum_{r \in R} U_r(x_r)$$

subject to

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

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

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.

Throughout the rest of the paper, we will make two assumptions on the primal problem $\mathbf{P}$:

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

Assumption 2: (Strict Concavity) The utility functions U_r are increasing, twice continuously differentiable and strictly concave in the interval X_r. Thus $-U_r''(x_r) \geq \gamma_r > 0$ for all $x_r \in X_r$, for all $r \in R$.

Note that the interior point assumption also implies that the problem $\mathbf{P}$ is feasible, i.e., it has a solution. The strict concavity assumption implies that the solution is unique.

Next we will present an equivalent formulation of this utility maximization problem. This equivalent formulation would be the key in developing the algorithms that we propose subsequently. Before we proceed with this alternative formulation, we introduce some new terminology that will help us in formulating the problem and describing our algorithms.

B. 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 nodes where the multicast tree "branches off". Thus in Figure 1, $\{\hat{r}_5, \hat{r}_6, \hat{r}_7\}$ 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. The junction nodes, apart from the source and the receiver nodes, play an important role in the optimization process, as we describe later. 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 set of links between a source/junction node and its immediate downstream junction/receiver node will be called a *branch*. For the multicast tree in Figure 1, there are 7 branches, as shown by $j_1, j_2, ..., j_7$. Note that any multicast tree can be broken up into a number of branches. Also note that branches belonging to the same multicast tree are disjoint, i.e., they have no link in common. Moreover, branches belonging to different multicast groups (multicast trees) are considered to be logically different (even if they consist of the same set of links). 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, $\hat{r}_5$ is the parent of r_1, $\hat{r}_7$ is the parent of $\hat{r}_5$, s is the parent of $\hat{r}_7$. Similarly, $\hat{r}_7$ is a child of s, while $\hat{r}_5, \hat{r}_6$ are children of $\hat{r}_7$, and so on. Parent and child branches of a branch j are also defined similarly.

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

[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.

Thus j_5 is the parent branch of j_1, j_7 is the parent branch of j_5, while j_5, j_6 are children branches of j_7, and j_1, j_2 are children branches of j_5, and so on. Note that j_7 does not have a parent branch, while j_1, j_2, j_3 and j_4 do not have any child branch. The (q, x, y, z) variables stored at the various junction/receiver nodes (as shown in Figure 1) will be explained later, when we refer to this figure again in Section VI.

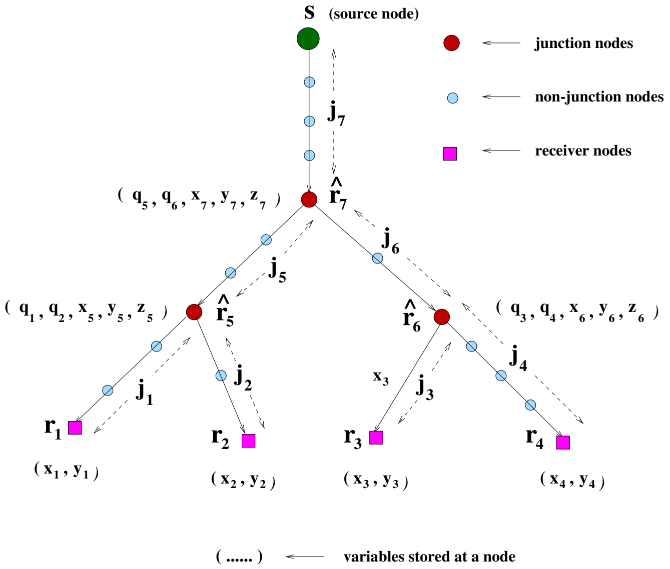

Fig. 1. An example of a multirate multicast tree

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 till 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.

C. An Alternative Formulation

There can be a large number of ways of solving the optimization problem $\mathbf{P}$. However, the challenge is to obtain a solution that is *distributed* and *scalable*. Decentralization and scalability are necessary conditions for the solution to be of any practical importance. A solution would not scale if, for example, the source or a junction node in the multicast tree has to maintain data for, send data to, or receive data from all downstream receivers of the tree. Since the number of receivers in the group can be very large, this might lead to tremendous processing and communication pressure on such a node, particularly if the node is the source, or a junction node close to the source. Also, 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 such a solution is clearly not implementable without a major modification to the existing standards. In the solution we propose, the amount of extra data that a 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. Thus the solution is scalable, and conforms with the existing standards.

Note that unlike the unicast case (where the constraints are linear), the link constraints in $\mathbf{P}$ contain some $\max$ functions. The $\max$ functions, besides being nonlinear, couple several variables together, and make the problem significantly more difficult. However, 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 could result in a very large number of linear constraints, which makes it difficult to develop a scalable solution. For example, each constraint in (1) can be replaced by a set of linear constraints, in each of which a $\max$ term is replaced by a variable included in it. However, it is easy to see that this could result in an exponential number of linear constraints.

Next we present a reformulation of the problem $\mathbf{P}$, where the constraint set is linearized such that it enables us to develop a scalable solution. First, we present some notation that we will use. Let $\hat{R}$ be the set of all junction nodes (over all multicast groups). Let $\tilde{R} = R \cup \hat{R}$. Note that each branch can be associated with a junction node or a receiver node (the junction/receiver node where the branch terminates). Thus in Figure 1, associate j_7 with $\hat{r}_7$, j_5 with $\hat{r}_5$, j_1 with r_1, j_2 with r_2 and so on. Therefore, $\tilde{J}$, the set of all branches (over all multicast trees) can be written as, $\tilde{J} = J \cup \hat{J}$, where J is the set of branches associated with receiver nodes ("receiver branches") and $\hat{J}$ is the set of branches associated with junction nodes ("junction branches"). Let $K_l \subseteq \tilde{J}$ be the set of branches that share link $l \in L$. Now associate a rate variable y_j with each branch $j \in \tilde{J}$ (therefore y_j is also the rate variable for the receiver/junction node associated with branch j). Let $y = (y_j, j \in \tilde{J})$ be the vector of these rate variables. Let $r(j) \in \tilde{R}$ denote the receiver/junction node associated with branch $j \in \tilde{J}$. For each receiver branch j, let U_j denote the utility function associated with the corresponding receiver, $r(j)$. Now consider the following problem

$$\mathbf{P}' : \qquad \max \sum_{j \in J} U_j(y_j)$$

subject to :

$$\sum_{j \in K_l} y_j \leq c_l \qquad \forall\, l \in L \tag{3}$$

$$y_j \leq y_{\pi(j)} \qquad \forall\, j \in \tilde{J} \text{ s.t. } \pi(j) \neq \phi \tag{4}$$

$$y_j \in Y_j \qquad \forall\, j \in \tilde{J} \tag{5}$$

where $\pi(j)$ is the parent branch of branch j and the intervals Y_j are defined as

$$Y_j = \begin{cases} X_{r(j)} & \text{if} \quad j \in J \\ [0, B] & \text{if} \quad j \in \hat{J} \end{cases} \tag{6}$$

where B is any number satisfying

$$B > \max_{r \in R} B_r \tag{7}$$

It is easy to see that the interior point assumption holds for $\mathbf{P}^{'}$ if it holds for $\mathbf{P}$. Comparing $\mathbf{P}^{'}$ with $\mathbf{P}$, note that we have replaced each $\max$ term in (1) with a separate variable (the rate variable for that branch) and added the constraints (4) in order to enforce the property that the rate on a branch cannot be less than that on its children branches. Note that since the objective function of $\mathbf{P}^{'}$ is strictly concave with respect to the variables $(y_j, j \in J)$, these variables are unique in any optimal solution of $\mathbf{P}^{'}$. The variables $(y_j, j \in \hat{J})$ in an optimal solution may not be unique, though.

Theorem 1: Let $y^* = (y_j^*, j \in \tilde{J})$ be any optimal solution of $\mathbf{P}^{'}$. Then $y_J^* = (y_j^*, j \in J)$ is the unique optimal solution of $\mathbf{P}$.

The proof of the above result is straightforward, and is left to the reader (it can also be found in [5]). The theorem shows that we can obtain the optimum solution of $\mathbf{P}$ by solving $\mathbf{P}^{'}$. Note that the variables $(y_j, j \in \hat{J})$ may not necessarily be equal to the corresponding $\max$ values (the actual rates on the corresponding branches) at optimality, but may be greater, as (4) indicates. We will therefore call the variable y_j the *pseudo-rate* on branch j. In the subsequent sections, we will show how we can solve $\mathbf{P}^{'}$ to obtain the optimal pseudo-rates, and use them to obtain the optimal (actual) rates.

III. Solution Approach

Since the problem $\mathbf{P}^{'}$ is separable, dual methods provide attractive approaches for obtaining distributed solutions (see Chp. 6 of [2]).

A. *The Dual Problem*

In the rest of the paper, we will often refer to the dual variables as "prices", in accordance with the terminology used in [6], [7], [10] etc. However, these prices may not necessarily have any correlation with the amount of revenue paid, if any, by the users for the network resources they use. These prices are to be interpreted as "congestion prices", and can be viewed as a measure of congestion caused by an user at a network resource. As we will see later, these prices represent the congestion information that the network elements need to share with each other and with the users, so that traffic rates on the different links/branches of the network can be adjusted appropriately.

Let p_l be the dual variable (price) associated with the link constraint (3) for link $l \in L$. Let $\tilde{p}_j$ be the sum of the prices of all the links in branch j, i.e., $\tilde{p}_j = \sum_{l \in \tilde{L}_j} p_l$, where $\tilde{L}_j$ is the set of links in branch j. We will call p_l the *link price* of link l and $\tilde{p}_j$ the *branch price* of branch j. Let $\tilde{J}_S$ be the set of branches that start from source nodes, i.e., $\tilde{J}_S = \{j : \pi(j) = \phi\}$. Then each constraint in (4) can be associated with a branch in $\tilde{J} \setminus \tilde{J}_S$. Let q_j be the dual variable (price) associated with the constraint (4) for branch $j \in \tilde{J} \setminus \tilde{J}_S$. Also let C_j be the set of children for any branch $j \in \tilde{J}$, i.e., $C_j = \{j^{'} : \pi(j^{'}) = j\}$. Note that $C_j = \phi$ for any $j \in J$ (receiver branch). Let $p = (p_l, l \in L)$ and $q = (q_j, j \in \tilde{J} \setminus \tilde{J}_S)$ be the price vectors.

The Lagrangian [2] for $\mathbf{P}^{'}$ is

$$\begin{aligned} & L(y,p,q) \\ &= \sum_{j \in J} U_j(y_j) - \sum_{l \in L} p_l \Big(\sum_{j \in K_l} y_j - c_l \Big) \\ &\quad - \sum_{j \in \tilde{J} \setminus \tilde{J}_S} q_j (y_j - y_{\pi(j)}) \\ &= \sum_{j \in J \cap \tilde{J}_S} \{ U_j(y_j) - y_j \tilde{p}_j \} \\ &\quad + \sum_{j \in J \setminus \tilde{J}_S} \{ U_j(y_j) - y_j(\tilde{p}_j + q_j) \} \\ &\quad + \sum_{j \in \hat{J} \cap \tilde{J}_S} \{ - y_j \, (\tilde{p}_j - \sum_{j^{'} \in C_j} q_{j^{'}}) \} \\ &\quad + \sum_{j \in \hat{J} \setminus \tilde{J}_S} \{ - y_j \, (\tilde{p}_j + q_j - \sum_{j^{'} \in C_j} q_{j^{'}}) \} \\ &\quad + \sum_{l \in L} p_l c_l \end{aligned} \tag{8}$$

We can simplify the expression in (8) by introducing additional variables $(q_j, j \in \tilde{J}_S)$, assumed to be identically zero. Then the Lagrangian in (8) can be written as

$$\begin{aligned} & L(y,p,q) \\ &= \sum_{j \in J} \{ U_j(y_j) - y_j(\tilde{p}_j + q_j) \} \\ &\quad + \sum_{j \in \hat{J}} \{ - y_j \, (\tilde{p}_j + q_j - \sum_{j^{'} \in C_j} q_{j^{'}}) \} \\ &\quad + \sum_{l \in L} p_l c_l \end{aligned} \tag{9}$$

Let $Y = \{(y_1, ..., y_{|\tilde{J}|}) : y_j \in Y_j \;\; \forall j \in \tilde{J}\}$. The dual of the primal problem $\mathbf{P}^{'}$ is (see [2])

$$\mathbf{D}^{'} : \qquad \min_{p,q \geq 0} D(p,q) \tag{10}$$

where the dual objective function $D(p,q)$ is given as

$$\begin{aligned} D(p,q) &= \max_{y \in Y} L(y,p,q) && (11) \\ &= \sum_{j \in J} \tilde{D}_j(p,q) + \sum_{l \in L} p_l c_l && (12) \end{aligned}$$

where $\tilde{D}_j(p,q)$ is given as

$$\begin{aligned} & \tilde{D}_j(p,q) \\ &= \begin{cases} \max_{y_j \in Y_j} \{ U_j(y_j) - y_j(\tilde{p}_j + q_j) \} & \text{if } j \in J \\ \max_{y_j \in Y_j} \{ - y_j \, (\tilde{p}_j + q_j - \sum_{j^{'} \in C_j} q_{j^{'}}) \} & \\ & \text{if } j \in \hat{J} \end{cases} \end{aligned} \tag{13}$$

As (11)-(13) show, the problem of evaluating the dual objective function for a given (p,q) (or in other words, finding $y \in Y$ such that it maximizes the Lagrangian $L(y,p,q)$) can be decomposed into separate branch optimization problems, one for each of its branches. These decompositions are possible due to the separable nature of the problem $\mathbf{P}^{'}$. Moreover, note from (13) that in order to evaluate $\tilde{D}_j(p,q)$ for any branch j, the only prices that are required are the p and q prices associated with that branch, and the q prices of its children branches.

B. Interpretation of the Prices

In this section, let us interpret the pseudo-rate on a branch as the actual rate on that branch. Then the interpretation of the link prices p is straightforward. As noted in [10] for the unicast case of this problem, p_l can be interpreted as the "congestion price" of link l. Note that at optimality, from Kuhn-Tucker conditions, $p_l > 0$ if and only if $\sum_{j\in K_l} y_j = c_l$. Therefore, at optimality, price of an uncongested link is zero.

Now let us try to interpret the q prices. Let $\tilde{J}_m$ denote the set of all branches that belong to the multicast tree of group m. Then $\tilde{J}_m = J_m \cup \hat{J}_m$, where J_m and $\hat{J}_m$ are respectively the set of the receiver and junction branches in group m's multicast tree. Consider any branch j belonging to group m's multicast tree, i.e., $j \in \tilde{J}_m$. Let T_j be the set of all branches downstream of (and including) branch j in group m's multicast tree. For example, in Figure 1, $T_{j_5} = \{j_1, j_2, j_5\}$. Now for any branch $j^{'} \in \tilde{J}_m$, consider the "branch profit" term $P_{j^{'}}$ defined as (see (13))

$$P_{j^{'}} = \begin{cases} U_{j^{'}}(y_{j^{'}}) - y_{j^{'}}(\tilde{p}_{j^{'}} + q_{j^{'}}) & \text{if } j^{'} \in J_m \\ -\, y_{j^{'}}\,(\tilde{p}_{j^{'}} + q_{j^{'}} - \sum_{j^{''}\in C_{j^{'}}} q_{j^{''}}) & \text{if } j^{'} \in \hat{J}_m \end{cases} \quad (14)$$

Summing up the branch profits $P_{j^{'}}$ for all $j^{'} \in T_j$, we obtain $P(T_j)$, the overall profit of the subtree T_j, as

$$\begin{aligned} P(T_j) \;\; = \;\; & \sum_{j^{'}\in J_m\cap T_j} U_{j^{'}}(y_{j^{'}}) - \sum_{j^{'}\in T_j} \tilde{p}_{j^{'}} y_{j^{'}} \\ & - q_j y_j - \sum_{j^{'}\in T_j - \{j\}} q_{j^{'}}(y_{j^{'}} - y_{\pi(j^{'})}) \quad (15) \end{aligned}$$

Note that at optimality, from Kuhn-Tucker conditions, $q_{j^{'}}(y_{j^{'}} - y_{\pi(j^{'})}) = 0$. Therefore, at optimality,

$$P(T_j) = \sum_{j^{'}\in J_m\cap T_j} U_{j^{'}}(y_{j^{'}}) - \sum_{j^{'}\in T_j} \tilde{p}_{j^{'}} y_{j^{'}} - q_j y_j \quad (16)$$

Note that $\sum_{j^{'}\in J_m\cap T_j} U_{j^{'}}(y_{j^{'}})$ is the total utility of all the receivers in T_j. Also, if we interpret p_l as the price per unit bandwidth on link l (and hence $\tilde{p}_j$ as the price per unit bandwidth on branch j), then $\sum_{j^{'}\in T_j} \tilde{p}_{j^{'}} y_{j^{'}}$ is the price paid for the usage of all the links in the subtree T_j. However, note that the receivers in T_j also use some bandwidth in the links of the branches from the source down to (and excluding) branch j, which are not included in T_j. Therefore, from (16), we see that $q_j y_j$ can be interpreted as the price paid by the receivers in T_j for the usage of links that are outside T_j. Note that at optimality, $q_j = 0$ if $y_j < y_{\pi(j)}$. In other words, the receivers of T_j are not charged for the usage of links in the branches from the source to branch $\pi(j)$ if they use less rates than the rate at branch $\pi(j)$. Thus a receiver does not get charged for a link/branch if it is not using the maximum amount of bandwidth amongst all receivers of its group that use that link/branch.

Now let us calculate the total profit of the multicast group m, i.e., the sum total of the profits of all of its branches. At optimality, we obtain,

$$\sum_{j\in\tilde{J}} P_j = \sum_{j\in J_m} U_j(y_j) - \sum_{j\in\tilde{J}_m} \tilde{p}_j y_j \quad (17)$$

which can be interpreted as the profit of the entire group, i.e., sum of the receiver utilities − price paid for link (branch) usages. Since the branch rates y_j are chosen to maximize the branch profits (see (13)), it follows that at optimality, the rates $y \in Y$ are such that these "group profits" are maximized for all groups. This can be seen as a generalization of the fact that in the unicast case, it is the "session profit" that is maximized (see [10]).

For the unicast case described in [10], each user r chooses its rate such that its "session profit" $\{U_r(y_r) - p^r y_r\}$ is maximized (where p^r is the sum of the prices of the links on its path). Considering the branch maximization problem of (13) and comparing it with the unicast case, we see that the session optimization problem of the unicast case has been replaced by the branch optimization problem in the multirate multicast case. In the latter case, the different branches act like somewhat independent unicast sessions, with the q variables ensuring that the rate relationships between parent/child branches (as in (4)) hold at optimality.

C. Dual Minimization

Since the objective function of the primal problem $\mathbf{P}^{'}$ is concave and the constraints linear, there is no duality gap (Proposition 5.2.1 of [2]). Moreover, the interior point assumption (Assumption 1) implies that the set of all dual solutions (which is the same as the set of Lagrange multipliers) is bounded [2](pp. 450). Let (p^*, q^*) be any dual optimal solution, and let $y(p^*, q^*) = (y_j(p^*, q^*), j \in \tilde{J})$ attain the maxima in (13). Then it is easy to show that $y_J(p^*, q^*) = (y_j(p^*, q^*), j \in J)$ is the unique optimal solution of $\mathbf{P}$.

Now let us see how we can minimize the dual as in (10). Gradient-based methods are, in general, attractive approaches to carry out minimizations of this type. Unfortunately, in our case, the dual objective function is *nondifferentiable*, and therefore its gradient may not always exist (unlike the unicast case, where the dual is differentiable). This is because in general, differentiability of the dual requires a unique primal optimizer (see [2] (Chp. 6)), whereas in our case, the optimal values of the variables $(y_j, j \in \hat{J})$ can be non-unique. Therefore the well-known gradient-based algorithms do not apply in this case.

One way to handle the nondifferentiability problem is to use *subgradient*[4] methods, which do not require differentiability of the objective function. Another alternative is to make an approximation to the original problem so that the dual becomes differentiable, and then apply gradient methods. In the following sections, we propose algorithms based on both these approaches.

IV. Subgradient Algorithm

The subgradient algorithm that we propose next is based on the subgradient method developed by N. Z. Shor, among others (see [14] (Chp. 2) for a detailed discussion on this method). In our problem, although the dual gradient does not exist, subgradients do. Moreover, from Proposition 6.1.1 of [2], it follows

[4]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. At points where the function is differentiable, the subgradient is unique, and is the same as the gradient.

that we can obtain a subgradient ∂D at (p, q) as

$$\partial D|_{p_l} = c_l - \sum_{j \in K_l} y_j(p, q) \tag{18}$$

$$\partial D|_{q_j} = y_{\pi(j)}(p, q) - y_j(p, q) \tag{19}$$

where $\partial D|_{p_l}$ and $\partial D|_{q_j}$ are the components of ∂D for p_l and q_j, respectively, and $y_j(p, q)$ are such that they maximize the Lagrangian $L(y, p, q)$ at p, q (i.e., attains the maximum in (13)). Now consider a sequence α_n satisfying

$$\lim_{n \to \infty} \alpha_n = 0 \qquad \sum_{n=1}^{\infty} \alpha_n = \infty \tag{20}$$

As an example, $\alpha_n = (1/n)$ is a sequence that satisfies (20). The iterative step of the subgradient algorithm is similar to the gradient projection algorithm, with the subgradient replacing the gradient and the step-size satisfying (20). Thus if $p_l^{(n)}, q_j^{(n)}$ are the prices at the nth step, then the update procedure of the prices are

$$p_l^{(n+1)} = [\, p_l^{(n)} + \alpha_n \,(\sum_{j \in K_l} y_j^{(n)} - c_l)\,]_+ \tag{21}$$

$$q_j^{(n+1)} = [\, q_j^{(n)} + \alpha_n \,(y_j^{(n)} - y_{\pi(j)}^{(n)})\,]_+ \tag{22}$$

where $[\,\cdot\,]_+ = \max(0, \cdot)$ and $y_j^{(n)}$ is such that it attains the maximum in (13) with prices $p_l^{(n)}, q_j^{(n)}$. The maximization in (13) is easy to carry out; for $j \in J$, $y_j^{(n)}$ can be expressed as

$$y_j^{(n)} = \begin{cases} b_{r(j)} & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} \geq U_j'(b_{r(j)}) \\ B_{r(j)} & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} \leq U_j'(B_{r(j)}) \\ U_j'^{-1}(\tilde{p}_j^{(n)} + q_j^{(n)}) & \text{o.w.} \end{cases} \tag{23}$$

while for $j \in \hat{J}$, $y_j^{(n)}$ can be expressed as

$$y_j^{(n)} = \begin{cases} 0 & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} - \sum_{j' \in C_j} q_{j'}^{(n)} > 0 \\ B & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} - \sum_{j' \in C_j} q_{j'}^{(n)} < 0 \\ b \quad \text{for any } b \in [0, B] & \text{o.w.} \end{cases} \tag{24}$$

Note that existence of the inverse function $U_j'^{-1}$ in (23) follows from the strict concavity assumption on the function U_j. Now assuming that the initial prices are feasible, i.e., $p^{(0)}, q^{(0)} \geq 0$, we can obtain the following convergence result.

Theorem 2: Consider dual subgradient algorithm as described in (21)-(24) with the step-sizes satisfying (20). Then the sequence of vectors $y_J^{(n)} = (y_j^{(n)}, j \in J)$ converges to the unique optimal solution of $\mathbf{P}$.

The proof of the above theorem is stated in [5]. Note that Theorem 2 does not state anything about the convergence of $y_{\hat{J}}^{(n)} = (y_j^{(n)}, j \in \hat{J})$. In general $y_{\hat{J}}^{(n)}$ may not converge, as can be expected from (24) (this does not matter since we infer the actual rates only from $y_J^{(n)}$). Note that since $\alpha_n \to 0$, the prices p, q (updated according to (21)-(22)) converge, even though the pseudo-rates y may not.

In practice, it may be difficult to implement the step-size constraint (20); a constant step-size may be more practical. For a constant step-size, however, the subgradient method may not converge to an optimal solution. However some weaker results can be derived in that case. More specifically, it can be shown that the constant step-size subgradient method can be made to converge in any given positive neighbourhood around the optimum, by choosing the step-size to be "sufficiently small". In practice too, a constant step-size works well, provided the step-size is small (see Section VII).

In section VI, we describe a distributed implementation of this algorithm with constant step-sizes.

V. Proximal Approximation Algorithm

The reason behind the nondifferentiability of the dual objective function is the lack of strict concavity of the primal objective function in $\mathbf{P}'$ (note that the primal objective function is strictly concave with respect to the variables $(y_j, j \in J)$, but not so with respect to the variables $(y_j, j \in \hat{J})$). Note, we can make the primal objective function strictly concave by adding some strictly concave terms, for each of the variables $(y_j, j \in \hat{J})$, and thereby make the dual differentiable. The algorithm that we describe next uses this idea, and is based on the *proximal approximation method* proposed in [3](Section 3.4.3).

For each $j \in \hat{J}$, define $U_j(y_j) = -\frac{1}{2\kappa}(y_j - z_j)^2$, where $\kappa > 0$ is any constant, and $(z_j, j \in \hat{J})$ are additional variables. Now consider the approximate primal problem $\mathbf{P}''$ which has the same set of constraints as $\mathbf{P}'$ ((3)-(5)), but with the objective function $\sum_{j \in \tilde{J}} U_j(y_j) = \sum_{j \in J} U_j(y_j) - \frac{1}{2\kappa} \sum_{j \in \hat{J}} (y_j - z_j)^2$. The proximal approximation algorithm is an iterative procedure, as stated below. In the procedure, assume $y^{(0)}$ is any feasible point, and $z^{(0)} = y_{\hat{J}}^{(0)}$.

<u>Proximal Approximation Algorithm</u>

For $n = 1, 2, ...$, do:

<u>*Step 1:*</u> Solve $\mathbf{P}''$, with $z_j = z_j^{(n)}$ for all $j \in \hat{J}$, to obtain the new optimal solution $y^{(n+1)}$.

<u>*Step 2:*</u> Set $z_j^{(n+1)} = y_j^{(n+1)}$ for all $j \in \hat{J}$

Then, in this case too, we can show a convergence result similar to Theorem 2.

Theorem 3: Under the proximal approximation algorithm, the sequence of vectors $y_J^{(n)} = (y_j^{(n)}, j \in J)$ converges to the unique optimal solution of $\mathbf{P}$.

For the proof of the above theorem, see [5].

We will solve Step 1 of the algorithm described above using the dual approach. Following the analysis of Section III, it is easy to see that the dual function $D(p, q)$ in this case is given as

$$D(p, q) = \sum_{j \in \tilde{J}} \tilde{D}_j(p, q) + \sum_{l \in L} p_l c_l \tag{25}$$

where $\tilde{D}_j(p, q)$ is given as (compare with (13))

$$\tilde{D}_j(p, q) = \begin{cases} \max_{y_j \in Y_j} \{U_j(y_j) - y_j(\tilde{p}_j + q_j)\} & \text{if } j \in J \\ \max_{y_j \in Y_j} \{U_j(y_j) - y_j\,(\tilde{p}_j + q_j - \sum_{j' \in C_j} q_{j'})\} & \\ & \text{if } j \in \hat{J} \end{cases} \tag{26}$$

Now since the primal objective function is strictly concave, the dual is differentiable, and the components of the gradient ∇D at (p, q) are obtained as

$$\nabla D|_{p_l} = c_l - \sum_{j \in K_l} y_j(p, q) \tag{27}$$

$$\nabla D|_{q_j} = y_{\pi(j)}(p, q) - y_j(p, q) \tag{28}$$

where $\nabla D|_{p_l}$ and $\nabla D|_{q_j}$ are the components of ∇D for p_l and q_j, respectively, and $y_j(p, q)$ are such that they attain the maximum in (26). Now applying the gradient projection method with a constant step-size α, the price update procedures at the nth step become

$$p_l^{(n+1)} = [\, p_l^{(n)} + \alpha\, (\sum_{j \in K_l} y_j^{(n)} - c_l)\,]_+ \tag{29}$$

$$q_j^{(n+1)} = [\, q_j^{(n)} + \alpha\, (y_j^{(n)} - y_{\pi(j)}^{(n)})\,]_+ \tag{30}$$

where, as before, $y_j^{(n)}$ is such that it attains the maximum in (26) with prices $p_l^{(n)}, q_j^{(n)}$. The maximization in (26) is easy to carry out; for $j \in J$, $y_j^{(n)}$ can be expressed as (compare with (23))

$$y_j^{(n)} = \begin{cases} b_{r(j)} & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} \geq U_j^{'}(b_{r(j)}) \\ B_{r(j)} & \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} \leq U_j^{'}(B_{r(j)}) \\ U_j^{'\,-1}(\tilde{p}_j^{(n)} + q_j^{(n)}) & \text{o.w.} \end{cases} \tag{31}$$

while for $j \in \hat{J}$, $y_j^{(n)}$ can be expressed as (compare with (24))

$$y_j^{(n)} = \begin{cases} 0 \;\; \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} - \sum_{j' \in C_j} q_{j'}^{(n)} \geq U_j^{'}(0) \\ B \;\; \text{if } \tilde{p}_j^{(n)} + q_j^{(n)} - \sum_{j' \in C_j} q_{j'}^{(n)} \leq U_j^{'}(B) \\ U_j^{'\,-1}(\tilde{p}_j^{(n)} + q_j^{(n)} - \sum_{j' \in C_j} q_{j'}^{(n)}) \quad \text{o.w.} \end{cases} \tag{32}$$

Now assuming that the initial prices are feasible, i.e., $p^{(0)}, q^{(0)} \geq 0$, we can obtain the following convergence result. Let $\bar{K} = \max_{l \in L} |K_l|$ be the maximum number of branches on any link, and $\bar{L} = \max_{j \in \tilde{J}} |\tilde{L}_j|$ be the maximum number of links on any branch. Let $\bar{\gamma} = \min(\frac{1}{\kappa}, \gamma)$, where $\gamma = \min_{r \in R} \gamma_r$, and γ_r is defined as in Assumption 2.

Theorem 4: Consider the dual gradient projection algorithm as described in (29)-(32) with the step-size α satisfying $0 < \alpha < \frac{2\bar{\gamma}}{\bar{K}(\bar{L}+1)}$. Then the sequence of vectors $y^{(n)} = (y_j^{(n)}, j \in \tilde{J})$ converges to the unique optimal solution of $\mathbf{P}^{''}$.

The above result can be proved in the same way as Theorem 1 of [10]. An alternative proof is stated in [5] (which however proves the result for a smaller upper-bound on the step-size).

Note that the proximal approximation algorithm is a two-level optimization procedure, as evident from its description. Thus Step 2 of the algorithm is executed only when procedure used to solve Step 1 converges. Since this convergence can be asymptotic, in a real network, it may be difficult to decide when Step 2 should be executed. However, note that if κ is large, the objective function of $\mathbf{P}^{''}$ is close to the actual objective function. Therefore, we would expect that Step 1 would converge to a value close to the optimum, even when executed once. In general, a larger κ requires fewer iterations (of Steps 1 & 2) for convergence of the proximal approximation method (see Section 3.4.3 of [3]). The disadvantage of using a large κ is that the convergence is guaranteed only if the step-size is sufficiently small (Theorem 4), and a small step-size may lead to slow convergence. A practical implementation of the algorithm is described in the next section.

VI. DISTRIBUTED IMPLEMENTATION

Now we describe how the subgradient algorithm (SGA) and the proximal approximation algorithm (PAA) can be implemented in a real network in a distributed and scalable way .

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 forward control packets (FCP) to its children nodes. Also assume that each receiver/junction node sends backward control packets (BCP) to its parent node (see Figure 2). Note that the BCP that a junction node sends to its parent is formed by merging the BCPs that it receives from all of its children. As in the figure, each FCP contains a price field $\overline{P}$, while each BCP contains a field $\overline{X}$ (which contains the actual rate) and a field $\overline{Y}$ (which contains the pseudo-rate). Note that in real implementation, these control packets need not be communicated as separate packets; the FCPs can be piggybacked on the data packets, while the BCPs can be piggybacked on the acknowledgement (ACK) packets.

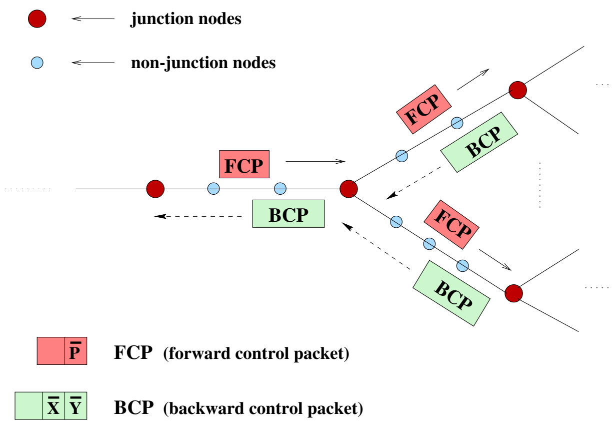

Fig. 2. Message exchanges

For any branch $j \in \tilde{J}$, the actual rate variable x_j and the pseudo-rate variable y_j (and also the variable z_j, in the case of PAA) are stored in $r(j)$, the junction/receiver node associated with branch j. A junction node $r(j)$ also stores the variables (prices) $q_{j'}$, for all $j^{'} \in C_j$ (see Figure 1). The variable (price) p_l for any link $l \in L$ is stored in the node where the link originates.

When sending an FCP to a child, a source/junction node stamps the corresponding q price in the $\overline{P}$ field of the packet (the source always stamps a price of 0). Each subsequent link (i.e., the node which stores the link price) on the branch leading to the child adds the link price in the $\overline{P}$ field. After the child (which is a junction or receiver node) receives the FCP, it uses the price in the $\overline{P}$ field to compute the new pseudo-rate (see the algorithms for the receiver and junction nodes, stated below).

A junction/receiver node communicates the x, y variables stored at it to its parent node through the $\overline{X}$, $\overline{Y}$ fields of the BCPs. The y variable is used to update the corresponding q variable stored at the parent node. The x variable lets the parent node know at what rate it needs to send traffic to the corresponding child. The x variable at a receiver node is always set to the corresponding y variable. The x variable at a junction node is set to the maximum of the x variables (communicated through the BCPs) of its children nodes. As already mentioned, it is the x variable that determines the traffic rate on the corresponding branch (i.e., the rate at which the associated junction/receiver node receives traffic). Therefore, for the branches associated with junction nodes, at any step of the optimization process, the actual rate of traffic can be less than, equal to, or even greater than the corresponding pseudo-rate. Also note that when a BCP is going through the node associated with link l, the node reads the field $\overline{Y}$ and uses it to update the price p_l stored at the node.

In the algorithms stated below, the pseudo-rate computation occurs at a node on receiving an FCP, while updates of the variables p, q, x occur on receiving a corresponding BCP (at the receiver nodes, however, x is updated on the arrival of an FCP). In practice, these updates can also be done after some fixed time-intervals.

In the following, $\overline{P}_j$ denotes the $\overline{P}$ field of the FCPs on branch j, while $\overline{X}_j, \overline{Y}_j$ denote the $\overline{X}$, $\overline{Y}$ fields of the BCPs on branch j (note, these BCPs packets travel on the backward direction on branch j). Also, in the algorithms described below, the step-size for price updates is kept constant at α. Thus the algorithms for the source/receiver nodes for SGA and PAA become the same. However, the algorithm for the junction nodes for the two cases have a few differences. Below, we describe the junction node algorithms for SGA and PAA under the same heading, pointing out the specific differences between the two, wherever they exist. Also, for the case of PAA, we assume that the updating of the z variables (Step 2 of PAA) occur at regular intervals. Since we would like these updates to occur at a much larger timescale than the rest of the optimization procedure, these intervals will typically be much larger than the intervals at which the rate/price updates occur.

Link l's algorithm:

On receiving a BCP:

Read the $\overline{Y}$ field to know the updated pseudo-rates, and update the link's price p_l as

$$p_l \leftarrow [\, p_l + \alpha \,(\sum_{j \in K_l} \overline{Y}_j - c_l) \,]_+$$

and forward the BCP on to the next link.

On receiving an FCP:

Add the price p_l to the $\overline{P}$ field of the FCP and forward it on to the next link.

Source s's algorithm:

On receiving a BCP:

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

2. Send an FCP to that child, setting the $\overline{P}$ field to 0.

Receiver $r(j)$'s algorithm:

On receiving an FCP:

1. Read the $\overline{P}$ field to know the appropriate updated price, and calculate the new rates as:

$$x_j = y_j \leftarrow \arg\max_{y \in Y_j} \{\, U_j(y) - \overline{P}_j y \,\}$$

2. Send a BCP to the parent node, setting $\overline{X}_j \leftarrow x_j$ and $\overline{Y}_j \leftarrow y_j$

Junction node $r(j)$'s algorithm:

On receiving an FCP:

1. Read the $\overline{P}$ field to know the appropriate updated price, and calculate the new pseudo-rate as:

For SGA:

$$y_j \leftarrow \arg\max_{y \in Y_j} \{\, -(\overline{P}_j - \sum_{j' \in C_j} q_{j'})y \,\}$$

For PAA:

$$y_j \leftarrow \arg\max_{y \in Y_j} \{\, -\frac{1}{2\kappa}(y - z_j)^2 - (\overline{P}_j - \sum_{j' \in C_j} q_{j'})y \,\}$$

2. Send an FCP to each child node $r(j')$ ($j' \in C_j$), setting $\overline{P}_{j'} \leftarrow q_{j'}$.

On receiving a BCP:

1. Let $r(j')$ ($j' \in C_j$) be the child node that sent the BCP.
Read the $\overline{X}$ and $\overline{Y}$ fields to know the updated rates, and
(a) Update the price $q_{j'}$ as

$$q_{j'} \leftarrow [\, q_{j'} + \alpha\,(\overline{Y}_{j'} - y_j)\,]_+$$

(b) Update the actual rate as

$$x_j \leftarrow \max_{j' \in C_j} \overline{X}_{j'}$$

2. On receiving the BCP from all children nodes, send a BCP to the parent node, setting $\overline{X}_j \leftarrow x_j$ and $\overline{Y}_j \leftarrow y_j$.

At regular intervals: /* Only for PAA */

Update z_j as

$$z_j \leftarrow y_j$$

VII. Experimental Evaluation

In earlier sections, we have proved the convergence of our algorithms with the assumption of synchronous updates. However, simulations that we have carried out on various network topologies confirm that the algorithms achieve the optimal rates even in an asynchronous slowly time-varying environment. Next we present a few representative examples to demonstrate this fact. In these experiments, the algorithms are implemented as described in the previous section, with the difference that the prices are updated only at regular intervals.

Consider Figure 3, which shows two multicast groups sharing a 10-link network. 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

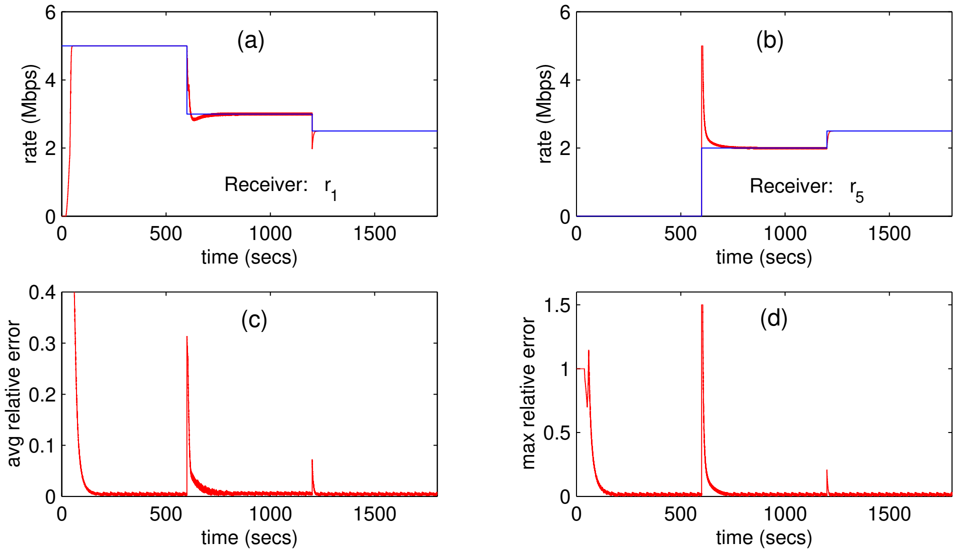

Fig. 4. Convergence of achieved rates for SGA. (The straight lines in (a) and (b) are the optimal (theoretical) rates.)

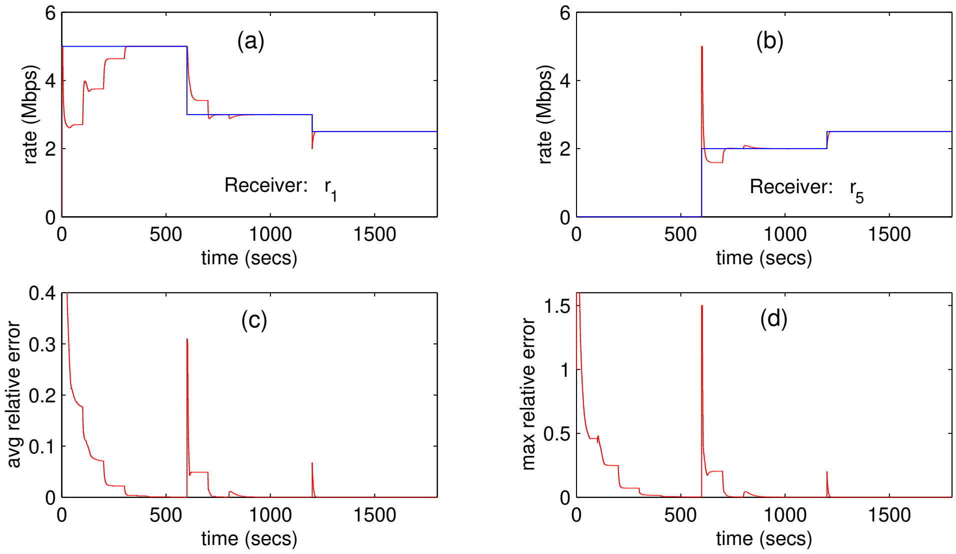

Fig. 5. Convergence of achieved rates for PAA. (The straight lines in (a) and (b) are the optimal (theoretical) rates.)

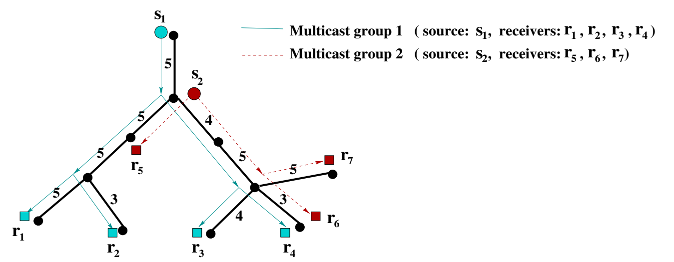

Fig. 3. An example network (The numbers associated with the links are the link capacities (in Mbps). The propagation delay for each link is 1 ms.)

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 $t = 600$ secs, while receiver r_2 leaves at $t = 1200$ secs. Figure 4, which shows some rate plots in the time window 0-1800 secs, demonstrate the performance of the subgradient algorithm (SGA) (with a constant step-size $\alpha = 0.001$ and a price update interval of 0.1 sec) in this scenario. Figure 4(a) and (b) show the computed (requested) 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 computed 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. Note that there is a sharp peak in Figure 4(b) at $t = 600$ secs (when r_5 arrives). This is because on arrival, r_5 receives a very small congestion price from the network and thus requests the maximum rate of traffic. However, as the figure shows, this rate decreases gradually and converges to the optimal rate. Figure 4(c) and (d) show the average and maximum relative errors over all receivers. If $x_r^c(t)$ and $x_r^o(t)$ respectively denote the computed and optimal rates of receiver r at time t, the relative error for receiver r at time t is defined as $|1 - \frac{x_r^c(t)}{x_r^o(t)}|$. The sharp peaks exhibited by the curves 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.

A careful observation of Figure 4(c) and (d) shows that even after reaching the optimal values, the rates exhibit very small fluctuations around the optimum. In general, for constant step-sizes, the computed rates in SGA converge to a neighborhood of the optimum rates and oscillate in the neighborhood thereafter. Smaller step-sizes reduce oscillations, but at the same time slow down the convergence. The choice of the step-size is trade-off between the speed of convergence and magnitude of oscillations.

As we would intuitively expect, the performance of PAA with a large proximal constant κ is very similar to that of SGA (in that case, PAA converges close to the optimal solution in a single execution of Step 1). Next we investigate the convergence properties of PAA for a small κ, in the same example as described above. Figure 5 shows some rate plots for PAA, when $\kappa = 10$, and the update interval for the z variables is 100 secs. All other parameters are the same as those for SGA. Figure 5 demonstrates the convergence of PAA in an asynchronous environment. Note that in this case, PAA converges after multiple executions of Steps 1 and 2.

Comparing Figures 4 and 5, we see that in this particular example, SGA converges faster that PAA. In this example, a larger value of κ in PAA results in faster convegence, and the convergence speed of PAA matches with that of SGA when κ is sufficiently large. However, in general, a larger value of κ may not always lead to better performance, for reasons we have stated in Section V. Moreover, note that in PAA, the rates do not show any fluctuations after they have converged to the optimum, unlike those in SGA.

VIII. CONCLUDING REMARKS AND FUTURE WORK

Note that in the algorithms presented here, the congestion prices (dual variables) could vary over a wide range. This poses a problem in communicating the prices to the user using a small number of bits, without compromising accuracy. In [1], the authors present a randomized marking based implementation of the algorithm presented in [10], where a single congestion indication bit is marked probabilistically based on the link price. Our algorithms for the multicast case could also be implemented in a similar way.

There are several related issues that need to be investigated further. Note that all the convergence results presented in the paper are for synchronous updates. Although the algorithms converged to the optimal rates in all of our experiments carried out in asynchronous environments, derivation of a formal proof of convergence for that case is an interesting problem from the theoretical point of view. Also note that in our algorithms, the pseudo-rates, required for price updates, need to be explicitly conveyed from the junction/receiver nodes to the links. Whether the algorithms can be suitably modified, without disturbing their convergence properties, to avoid this overhead of explicit pseudo-rate communication, remains an interesting open question.

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, *Nonlinear Programming*, Athena Scientific, 1995.

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

[4] 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.

[5] K. Kar, S. Sarkar, L. Tassiulas, "Optimization Based Rate Control for Multirate Multicast Sessions", Technical Report TR 2000-21, Institute for Systems Research and University of Maryland, 2000.

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

[7] 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.

[8] 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.

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

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

[11] 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.

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

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

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

[15] 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*.

[16] 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.

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

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

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