Optimization-Based Congestion Control for Multicast Communications

CMPSCI TR 00-33

Jonathan K. Shapiro Don Towsley
Jim Kurose
Department of Computer Science
University of Massachusetts at Amherst
{jshapiro, towsley, kurose}@cs.umass.edu

September 25, 2000

Abstract

Widespread deployment of multicast depends critically on the existance of congestion control protocols that are provably fair to unicast traffic. In this work, we present an optimization-based congestion control mechanism for one-to-many communication with provable fairness properties. The optimization-based approach attempts to find an allocation of rates that maximizes the aggregate utility of the network. We show that the utility of multicast sessions must be defined in a particular way if a widely accepted property of aggregate utility is to hold. Our definition of session utility amounts to maximizing a weighted sum of simple utility functions, with weights determined by the number of receivers. The fairness properties of the optimal rate allocation depend both on the weights and form of utility function used. We show that although it is not strictly fair to unicast, the unfairness of our mechanism is bounded and can be controlled.

1 Introduction

Widespread deployment of multicast communication in the Internet depends critically on the existence of practical congestion control mechanisms that allow multicast and unicast traffic to share network resources fairly. Most service providers recognize multicast as an essential service to support a range of emerging network applications including audio and

video broadcasting, bulk data delivery, and teleconferencing. Nevertheless, these network operators have been reluctant to enable multicast delivery in their networks, often citing concerns about the congestion such traffic may introduce. There is a clear need for multicast congestion control algorithms that are provably fair to unicast traffic if these concerns are to be addressed. In this paper, we present a congestion control mechanism for single-rate multicast traffic based on an economic pricing model and show that although it is not strictly fair to unicast traffic, its unfairness is bounded and can be controlled.

We first formulate the multicast congestion control problem as a utility maximization problem, extending existing work for unicast. A naive generalization of the existing formulation would treat single-rate multicast sessions no differently from unicast sessions, modeling each by an unweighted utility function and maximizing the sum of session utilities. One problem with the naive approach is that it penalizes individual multicast sessions for using more network resources than unicast sessions without rewarding them for the bandwidth saved on links shared by multiple receivers. More serious than its unfairness to multicast sessions, an approach that maximizes the unweighted sum of utilities turns out to violate a generally accepted property of aggregate utility, namely, that the preference of the aggregate does not change if we simply measure utility on a different scale. This common-sense notion is why, for example, we reject as nonsense the statement that, as a group, residents of New York prefer a temperature of 70 degrees to 60 degrees Fahrenheit, but prefer a temperature of 15.5 to 21 degrees Celsius. If this invariance property is violated in the congestion control problem, the network operating point deemed most desirable would depend on an arbitrary choice of utility scale. We define an approach that uses session weights based on the number of receivers that satisfies this property. Moreover, we show that this approach is necessary to preserve invariance under a change in utility scale.

A consequence of adding session weights based on the number of receivers is that the the resulting rate allocations tend to favor sessions with more receivers over those with fewer. Since the weighted sum does not remove the original penalty against sessions that use more resources, it is not immediately clear whether multicast sessions fare better or worse than unicast under our modified formulation. We show that while our formulation favors multicast sessions, the resulting unfairness can be controlled and remains bounded in the

network topologies we have considered.

There has been much recent interest in understanding the behavior of networks in the presence of diverse congestion control mechanisms. Because network users are widely distributed, they only become aware of each other through the congestion signals they receive as they make use of network resources. Several conditions impact the degree of effectiveness of closed-loop congestion control in this environment. First, congestion signals must accurately inform the users about the effect of their own traffic on the state of the network. Second, users at the edges of the network must be able to adapt their rates in response to these signals in ways that are both fair to other users and appropriate for the application at hand. Third, users must have an incentive to engage in congestion control. The issue of incentives is perhaps controversial since it is well known that successful protocols such as TCP provide no such incentive and yet appear to operate in a fair and stable manner. However, as Floyd and Fall have observed [1], the continued stability of the network is by no means assured as a diversity of applications with different types of control mechanisms account for a growing portion of network traffic. In particular, TCP itself may be at risk of being starved by more aggressive control mechanisms.

Attempts to achieve effective congestion control have included improvements to the quality of congestion signals provided by the network [2, 3] and the development of guidelines for designers who wish to develop application-specific protocols that are both fair and effective [4, 5]. In multicast congestion control, responding to congestion is complicated by the source's need to handle both the feedback implosion and potential redundancy associated with receiving congestion signals from a large set of receivers. Many proposals address both problems by responding to congestion feedback from a small set of representatives. Since the criteria for selecting these representatives affects the fairness of the resulting protocol, a common heuristic is to respond to feedback from the most congested receiver, causing the multicast session to share bandwidth fairly on its most congested path [6]. Identifying the potentially changing set of representatives and suppressing feedback from other receivers in a practical protocol remains a difficult problem. The congestion control mechanism we present here does not require identifying distinguished receivers, allowing the problems of eliminating redundancy and feedback implosion to be treated separately from the problem

of ensuring fairness.

In the field of economics, the allocation of resources among autonomous, self-interested actors has long been a subject of study. The similarity of network congestion to other resource allocation problems commonly studied in economics has not escaped the attention of researchers [7, 8, 9, 10, 11], who have tried to adapt economic mechanisms such as auctions and markets to problems in communications networks. Pricing mechanisms in economics have the same features mentioned above that contribute to effective congestion control. Market prices convey an accurate summary of resource supply levels to consumers. Consumers, in turn, determine their own usage levels by weighing the cost of the resource against their subjective estimation of its worth. The monetary aspect of prices can provide an incentive to consume resources conservatively in times of scarcity.

Our work is based on a promising economics-inspired approach called *optimization-based congestion control* [12], which casts the congestion problem as one of utility maximization (alternately, cost minimization). This approach provides an elegant theoretical framework in which congestion signals are interpreted as prices, network users are modeled as utility maximizers, and the network sets prices in such a way to drive this set of self-interested users to a desirable operating point where their aggregate utility is maximized. This approach is appealing because it provides a sound formal foundation with which to develop congestion control mechanisms and understand their impact on the global behavior of the network. Specific link service disciplines and rate-control algorithms at end-hosts can be thought of as components of a distributed computation to solve the global optimization problem. Thus, improvements in congestion control can proceed in a principled fashion, driven by improvements in the underlying optimization algorithm. While the optimization-based approach has received much attention [13, 14, 15, 16, 17, 18, 19, 20, 21], it has not yet, to our knowledge, been applied to multicast congestion control. We will see in this paper, that applying this model to multicast offers a formal foundation for developing fair multicast congestion control algorithms.

The rest of this paper is structured as follows: In Section 2, we introduce the optimization-based approach and review related work. We extend the problem formulation to single-rate multicast in Section 3. In Section 4 we consider multicast session utility functions in

detail, presenting an axiomatic argument in favor of a particular definition. The fairness properties of our definition are analyzed in Sections 5-7, where we show that multicast sessions are favored and present evidence that such unfairness can be controlled. We conclude by briefly discussing the development of practical control mechanisms based on our results and highlighting future work.

2 Background and Related Work

Optimization-based congestion control casts the problem of bandwidth sharing as one of utility maximization. In a basic formulation, the network is modeled as a set of unidirectional links $\mathcal{L} = \{1, \ldots, L\}$. Associated with each link is a capacity, c_l for link l. Let $C = (c_l, l \in \mathcal{L})$. The workload for the network is generated by a set of sessions[1] S, which consume bandwidth. The set of links used by a particular session, s, is $L(s) \subseteq \mathcal{L}$. For a unicast session, the links of $L(s)$ are arranged end-to-end, forming an acyclic path between a source and a receiver. However, the topology of sessions is not explicit in the formulation. The set $L(s)$ can be any subset of links—for example, a tree in the case of multicast. The set of sessions using any particular link, l, is $S(l) \subseteq S$.

Each session is characterized by a utility function,

$$U_s : \mathbb{R}_+ \to \mathbb{R},$$

which is assumed to be an increasing and strictly concave function of session rate x_s. The network's objective is to optimize social welfare:

$$\max_{x_s \geq 0} \quad \sum_{s \in S} U_s(x_s) \tag{1}$$

$$\text{subject to} \sum_{s \in S(l)} x_s \leq c_l \quad l = 1, \ldots, L \tag{2}$$

The desirable network operating point is defined by the vector of session rates that solves this problem. The problem (1-2) can be solved using convex optimization techniques [22]. Under a standard economic interpretation, the Lagrange multipliers of such techniques are

[1]The terms 'session' and 'user' are synonymous in this paper.

referred to as *shadow prices* and can be shown to function as prices of network links [23]. The essential step in developing practical rate-control algorithms is to find a distributed algorithm for solving (1-2) in which each individual session need only compute a local optimization to set its own rates. There is a growing body of research devoted to finding such a distributed algorithm and using it as a basis for unicast rate-control in practical protocols [13, 14, 15, 16, 17, 18, 20, 19, 21].

Kelly [13] decomposes the basic optimization problem (1-2) into session and link subproblems, enabling the distributed computation of optimal rates. In addition to demonstrating the possibility of such a decomposition, Kelly also shows that the fairness properties satisfied by the resulting rate allocation are dependent on the functional form of session utility. In particular, *proportional fairness* is shown to result from the use of a logarithmic utility function, $U_s(x) = \log(x)$.

A vector of session rates x^* is said to be proportionally fair if it is feasible and for any other feasible vector x

$$\sum_{s \in S} \frac{x_s - x_s^*}{x_s^*} \leq 0$$

Proportional fairness favors smaller flows less strictly than the more commonly used fairness criterion of max-min fairness [24]. A vector of rates x^* is *max-min fair* if it is feasible and no individual's rate x_s can be feasibly increased without decreasing the rate of a less well-off individual $x_{s'} \leq x_s$. Kelly formally identifies proportional fairness and max-min fairness as two points on a continuum. An interesting finding in our work, presented in section 6.1, is the location of tcp-fairness within this continuum.

Low and Lapsley [17] have proposed a distributed algorithm based on a solution to the dual of problem (1-2); this work provides the basis for our work presented here. The basic idea of their algorithm is that each link in the network continuously adjusts its price and each session computes its optimal rate as a simple function of the total price of the links it uses. In order to offset a bias that favors short unicast paths over long ones, Low and Laplsey consider a modified optimization objective function that maximizes a weighted sum of session utilities. We too will use session weights to deal with a similar source of unfairness and show how these weights can be meaningfully assigned to multicast session utility functions.

Kunniyur and Srikant [16] use a penalty function formulation of problem (1-2) to separate the problem into a set of session optimization problems that can be solved by individual sender-receiver pairs using only end-to-end packet loss information, or with ECN-style marks on packets if available. In addition to the logarithmic utility function, these authors investigate the minimum potential delay (MPD) utility function [25], $U(x) = -1/x$.[2] In the penalty function formulation, the optimality condition for each session's local optimization problem is given by

$$x_s\, u'(x_s) - \beta\,(x_s - y_s) = 0, \tag{3}$$

where β is a constant and $(x_s - y_s)$ is the difference between sent and received rates for the session. For $u(x) = -1/x$, this condition is rewritten

$$1 - \beta\, x_s\,(x_s - y_s) = 0 \tag{4}$$

A natural rate-control algorithm simply attempts maintain the optimality condition.

$$\dot{x_s} = \gamma(1 - \beta\, x_s\,(x_s - y_s)), \tag{5}$$

where γ is a step size parameter. We see that in the penalty function formulation, an additive-increase/multiplicative-decrease control algorithm (the class of algorithm to which TCP congestion control belongs) emerges as a natural way for individual sessions to maintain a local optimality condition under the MPD utility function.

More recently, Bansal and Balakrishnan [26] have discovered a family of tcp-friendly control algorithms of the form,

$$\dot{x_s} = \alpha/x_s^k - \beta\, x_s^l\,(x_s - y_s), \tag{6}$$

where $k + l = 1$ and $l \leq 1$. It is straightforward to show that this entire family of algorithms is consistent with the optimality condition (4). In light of these results, we will represent the utility of unicast TCP sessions using the minimum potential delay utility function in this paper.

[2]The minimum potential delay utility function gets its name from the fact that the reciprocal rate $1/x$ can be thought of as a delay. Minimizing this delay is equivalent to maximizing $-1/x$.

3 Problem Formulation

3.1 Solving the Dual

Following the approach of Low and Lapsley [17], we find the solution to optimization problem (1) by solving its dual. The Lagrangian of (1) is defined as

$$\begin{aligned} \mathscr{L}(x;\lambda) &= \sum_s U_s(x_s) - \sum_l \lambda_l \Big(\sum_{s\in S(l)} x_s - c_l\Big) \\ &= \underbrace{\sum_s \Big(U_s(x_s) - x_s \sum_{l\in L(s)} \lambda_l\Big)}_{\text{separable in } s} + \sum_l \lambda_l c_l, \end{aligned} \tag{7}$$

where the multipliers λ_l are interpreted as link prices. The objective function $D(p)$ of the dual problem is the maximized Lagrangian.

$$D(p) = \max_x \mathscr{L}(x;\lambda) = \sum_s B_s(\lambda^s) + \sum_l \lambda_l c_l$$

where

$$\begin{aligned} B_s(\lambda^s) &= max_{x_s}(U_s(x_s) - x_s\lambda^s) \\ \lambda^s &= \sum_{l\in L(s)} \lambda_l \end{aligned} \tag{8}$$

Exploiting the separability of the first term in the Lagrangian, we can allow each user to perform an independent maximization of its own benefit $B_s(\lambda^s)$, where λ^s is the sum of all the link prices seen by user s [3]. The dual problem is a minimization of the dual objective function with respect to the link prices.

$$\min_{\lambda\geq 0} D(\lambda) \tag{9}$$

The network computes the link prices λ^* that solve (9). By duality theory, the rate vector that results from each user's maximization, $(x_s(\lambda^*), s \in S)$, contains the optimal rates in the original problem (1). The link prices summarize all of the congestion information relevant to each user, allowing them to perform a local maximization without coordinating with other

[3] In the unicast case, λ^s would be the price of the path between source and receiver.

users. Solving the original optimization problem in this way requires that users have some way to learn about the prices of the network resources they use.

The authors proceed to develop a distributed algorithm to solve the dual problem by gradient projection. Each link individually adjusts its price in the opposite direction of the gradient of $D(p)$. From the first derivative condition of (8), the rate that maximizes the user's benefit for a fixed λ^s is given by

$$x_s(\lambda^s) = U_s'^{-1}(\lambda^s)$$

In the synchronous rate control algorithm, links and users adjust their prices and rates according to the following update rules:

$$\lambda_l(t+1) = [\lambda_l(t) + \gamma(x^l(t) - c_l)]^+ \quad (10)$$

$$x_s(t+1) = [U_s'^{-1}(\lambda^s)]_{m_s}^{M_s} \quad (11)$$

where γ is a step-size parameter, $[x]_{m_s}^{M_s} = \max(m_s, \min(M_s, x))$, and m_s and M_s are the minimum and maximum rates required by user s.

In an asynchronous version of this algorithm, communication between users and links is not coordinated and some of the adjustments may be performed on the basis of outdated prices or rates. The authors are able to establish convergence for both synchronous and asynchronous versions provided the step-size γ is sufficiently small.[4]

As one would expect from Kelly's work [13], the fairness criteria satisfied by the resulting allocation depends on the utility functions of the users. Proportional fairness results if all of the users' utility functions are logarithmic, $U_s(x_s) = \log x_s, \forall s \in S$. More generally, if all users have the same well-behaved utility function and minimum and maximum rates m_s and M_s the following properties hold [17]:

- If two users s_1 and s_2 share the same path, their equilibrium rates $x_{s_1}^*$ and $x_{s_2}^*$ will be the same.

- If the path of s_1 is a subset of the path of s_2, then $x_{s_1}^* \geq x_{s_2}^*$ because $s1$ sees a total path price that is less than or equal to that of $s2$.

[4]Choosing the best value for γ remains an open problem.

- If the equilibrium price seen by s_1 is less than that seen by s_2, then $x^*_{s_1} \geq x^*_{s_2}$. This property is a generalization of the previous one since it holds even for sessions whose paths do not intersect.

These properties would seem to imply that the mechanism discriminates against long path lengths[5]. However, by an appropriate choice of utility function and weighting parameters it is always possible to obtain a proportionally fair or max-min fair equilibrium rate vector.

3.2 Generalization to Multicast

The problem formulation (1) and (9) is equally applicable to unicast and multicast. Recall that in the multicast case, the set of session links are those in that session's multicast tree. In a generalization of problem (1-2), we allow each session to use a different rate on each link. The rate of session s on link l is denoted x_{sl} and $\mathbf{x}_s = (x_{sl},\, l \in L(s))$ is a vector of rates for the session. The session utility function is now a function of this vector. In single-rate multicast, the rate x_{sl} is the same on all links, but the generalized formulation allows us to define a variety of session utility functions. The network optimization problem is

$$\max_{x=(\mathbf{x}_s, s\in S)} \sum_s U_s(\mathbf{x}_s) \tag{12}$$

subject to

$$x_{sl} = x_s \quad \forall\, l \in L(s) \tag{13}$$

$$\textstyle\sum_{s\in S(l)} x_s \leq c_l \quad \forall\, l \in L \tag{14}$$

The Lagrangian of the single-rate optimization problem (12) is identical to (7) and admits the same separation into per-session optimization problems as the unicast formulation. [6] Solving the single rate problem (12) is a straightforward extension of the solution presented in Subsection 3.1. As before, the objective function of the dual problem is the maximized

[5] or large multicast trees

[6] The multi-rate optimization problem is more difficult than the single-rate problem because its Lagrangian does not contain an expression that is separable by session.

Lagrangian, which admits separation into a maximization subproblem for each session similar to (8).

$$\begin{aligned} B_s(\lambda^s) &= \max_{x_s} U_s(\mathbf{x}_s) - x_s\lambda^s \\ \lambda^s &= \sum_{l \in L(s)} \lambda_l \end{aligned} \tag{15}$$

In the multicast case, λ^s is interpreted as the sum of link prices for the entire tree. [7] The essential difference between the session maximization problems (15) and (8) is the definition of the session utility function. In (15), the session utility function maps a vector of rates to a scalar utility value. More than one mapping can be defined and we will consider several alternatives later in this paper. At present, we consider one possible mapping that yields a basic control algorithm similar to Low and Lapsley's.

Consider the following session utility function, for which the session utility is defined as the sum of receivers' utilities in the multicast tree:

$$U_s(\mathbf{x}_s) = \sum_{l \in L(s)} U_{sl}(x_s)$$

$$U_{sl}(x_s) = \begin{cases} u(x) & \text{if } l \in R(s) \\ 0 & otherwise \end{cases}$$

where $R(s)$ is the set of links in s that terminate at receivers and $u : \mathbb{R}_+ \to \mathbb{R}$ is a utility function from a scalar to a scalar. We assume that all receivers have identical utility functions, $U_{sl}(x_s) = u(x_s)$, $\forall\, s \in S$. If $R_s = |R(s)|$, the first derivative condition for (15) is

$$\begin{aligned} \sum_{l \in R(s)} u'(x_s) &= \lambda^s \\ R_s u'(x_s) &= \lambda^s \\ u'(x_s) &= \frac{\lambda^s}{R_s} \\ x_s(\lambda^s) &= U_s'^{-1}\left(\frac{\lambda^s}{R_s}\right) \end{aligned} \tag{16}$$

[7] This difference complicates the development of a protocol based on the optimization formulation, since each individual receiver sees only the prices summed along its path from the source. Furthermore, one must take care not to count the price of a shared link more than once, therby overestimating the session price.

Ignoring, for the moment, the problems of determining λ^s in a multicast tree, we rewrite Low and Lapsley's synchronous update rules:

$$\lambda_l(t+1) = [\lambda_l(t) + \gamma(x^l(t) - c_l)]^+ \quad (17)$$

$$x_s(t+1) = [U_s'^{-1}(\lambda^s/R_s)]_{m_s}^{M_s} \quad (18)$$

The only difference between these update rules and Low and Lapsley's basic algorithm (10)-(11) is that the multicast session price is divided by the number of receivers. The effect of this price reduction is that the session can send at a higher rate than the basic algorithm (10)-(11) would allow. Since the price reduction depends on the size of the session, so does the increase in session rate. It is not clear whether an update rule like (18) leads to a rate allocation that one would consider fair, since it appears that larger sessions get faster rates. Since the update rules (17)-(18) were derived using a particular definition of session utility function, perhaps an alternate definition would lead to a more fair allocation. In the following sections we analyze the affect of this definition on the fairness properties of the resulting congestion control mechansim. It will turn out that a class of session utility functions that includes (17)-(18), while not absolutely fair to unicast sessions, does not starve them in the presence of larger sessions. Moreover, we will see that utility functions in this class makes sense in a way that other functions do not.

4 Multicast Utility Functions

In Section 3, we generalized the unicast optimization problem formulation to accommodate single rate multicast sessions. We were able to re-derive Low and Lapsley's update rules with only minor modifications. However, there is a subtle problem with this model that makes it difficult to apply to single-rate multicast. The problem concerns the definition of utility for an individual multicast session. A single scalar utility value is used to characterize the benefit of a higher rate to the session. For a unicast session, it makes little difference whether we consider this benefit to belong to the sender or receiver. For the purpose of unicast congestion control, we can treat the sender's and receiver's objectives as being one and the same. A multicast session, in contrast, has multiple receivers whose individual

objectives are generally not the same. A receiver connected by a high-bandwidth path from the source might benefit from a high session rate that would result in loss to other receivers with more constrained bandwidth.

One approach towards defining multicast session utility ignores this heterogeneity among receivers and defines its session utility function only with respect to the sender.[8] An alternative approach would be to define session utility as a function of the utilities of the receivers in the session. We informally refer to these two approaches as *sender-oriented* and *receiver-oriented*, respectively. It is not immediately clear which approach is most appropriate for multicast congestion control. Later in this section we will formalize these definitions and argue in favor of a receiver-oriented approach. Before doing so, however, we will digress briefly to provide some background about the use of utility functions in economics and the theory of social choice.

4.1 Digression: Utility Functions and Social Welfare

The use of concave increasing utility functions to represent session utility has a natural and intuitive interpretation. Utility is is a monotonically increasing function of its input when individuals prefer having as much of the input as possible. The concavity of the utility function captures the idea of diminishing marginal utility[9]. Both concavity and monotonicity are appropriate assumptions in the case of bandwidth for elastic traffic [27], where the input to the utility function is the session rate. [10]

Utility can be difficult to quantify precisely; there is no clear unit of utility and no agreed upon scale. Comparing the utility of two individuals can be tricky, particularly when they do not share the same utility function. Because of the difficulty in performing interpersonal comparisons of utility, economists customarily think of utility as an *ordinal magnitude*, meaning that the absolute magnitude of utility is meaningless, but that the

[8] We are assuming that multicast sessions have a single source.

[9] The term 'marginal utility' is used in economics to refer to the first derivative of the utility function.

[10] In this section, utility will be assumed to be a function of session rate; we do so for the sake of concreteness and continuity with the rest of the paper. It should be understood, however, that the discussion presented here applies to any utility function.

relative magnitudes of utilities at various rates for an individual session define preferences among rates and the relative differences in magnitude indicate the strength of the preferences [28]. A consequence of considering only ordinal magnitudes is that utility functions are unique only up to a linear transformation. That is, the utility maximizing behavior of an individual with utility function $u(x)$ is indistinguishable from one whose utility function is a linear transformation of $u(x)$. This restriction makes intuitive sense because a linear transformation simply represents a change in scale and a translation of the zero point of the utility function.

The notion of an aggregate utility function is a compelling extension of the concept of individual utility. Aggregate utility is defined by a *social welfare function* (SWF) that maps the vector of all session utilities to a scalar utility value representing the social desirability of the corresponding vector of rates. Since the SWF is not one-to-one, it induces a partial ordering over allocations of rates, known as the *social preference relation* (SPR). As with individual utility functions, we are primarily interested in this preference relation rather than the absolute magnitude of the SWF.

There are many ways to define the SWF, each carrying with it some subjective judgment about how individual preferences should be combined to determine a social preference. It is possible, however, to list some properties that seem reasonable for any definition of SWF [29].

- **Complete:** The SWF should be defined for all vectors of inputs.
- **Transitive:** The induced SPR should be a transitive relation.
- **Pareto Efficient:** Given a SWF $U(x)$ and two feasible vectors x and x' which differ only in one element i such that $u_i(x_i) > u_i(x'_i)$, it must be the case that $U(x) > U(x')$. In other words, if it is possible to increase one individual's utility without reducing any other's, doing so should improve the aggregate utility.
- **Independent of Irrelevant Alternatives (IIA):** Let u and v be two vectors of individual utility functions and U and V the corresponding aggregate utility functions

defined by a SWF. If x and y are two vectors of rates such that, for each individual i,

$$\begin{aligned} &(u_i(x_i) \geq u_i(y_i) \text{ and } v_i(x_i) \geq v_i(y_i)) \\ \text{or} \quad &(u_i(x_i) < u_i(y_i) \text{ and } v_i(x_i) < v_i(y_i)) \end{aligned}$$

then

$$\begin{aligned} &(U(x) \geq U(y) \text{ and } V(x) \geq V(y)) \\ \text{or} \quad &(U(x) < U(y) \text{ and } V(x) < V(y)) \end{aligned}$$

This property is somewhat subtle; notice that it is possible for two individuals to disagree in their preferences under u. However, if each individual's preference is unaffected by a change from u to v, then any such disagreements will persist. The IIA property demands that the social preference also be unaffected under these circumstances. In other words, the socially preferred allocation is invariant under a change in individual *utility functions* that leaves individuals' *preferences* unaffected.

- **Non-dictatorial:** A SWF U is said to be non-dictatorial if there exists no individual i with utility function u_i such that

$$u_i(x_i) \geq u_i(y_i) \iff U(x) \geq U(y) \ \ \forall\, x, y$$

Perhaps the most important result of social choice theory is Arrow's Impossibility Theorem, which states that no SWF can simultaneously satisfy all of the properties listed above [30]. In optimization-based congestion control, we adopt the sum of individual utilities as the SWF. It is easy to demonstrate that this SWF violates the IIA property. Consider three sessions sharing a simple network as shown in Fig. 1. We maximize the sum of utilities to get the vector of optimal rates, that is, the socially most preferred allocation. This vector depends on the functional form of session utility. One can show that the following conditions hold for the optimal vector of rates:

$$u'(x_0) = 2\,u'(x_1) \tag{19}$$

$$x_1 = x_2 \tag{20}$$

$$x_1 = 1 - x_0 \tag{21}$$

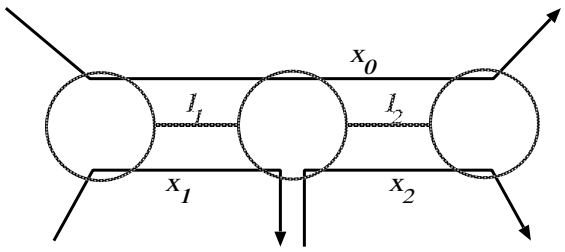

Figure 1: Three sessions with rates (x_0, x_1, x_2) in a simple two link network with unit link capacities. Each link is shared by two sessions.

Let x^* and x° be the vectors of optimal rates when $u(x) = \log(x)$ and $u(x) = -1/x$, respectively. Solving equations (19-21) for both cases, we get

$$\begin{aligned} x^* &= (1/3,\, 2/3,\, 2/3) \\ x^\circ &= (\sqrt{2}-1,\, 2-\sqrt{2},\, 2-\sqrt{2}) \end{aligned}$$

Now, each session's individual preference between x^* and x° is based on its own rate. Since utility functions are all increasing, the following inequalities hold regardless of the functional form of $u(x)$:

$$\begin{aligned} u(x_0^*) &< u(x_0^\circ) \\ u(x_1^*) &> u(x_1^\circ) \\ u(x_2^*) &> u(x_2^\circ) \end{aligned}$$

In other words, under any valid utility function, session 0 always prefers the MPD allocation to the proportionally fair allocation, and sessions 1 and 2 always prefer the proportionally fair allocation. However, the *social preference* changes depending on our choice of utility function. Indeed, it is precisely this violation of IIA that allows us to associate optimal rates under different functional forms of utility with different formal definitions of fairness.

Although the IIA property is neither required nor (in light of the Impossibility Theorem) worth pursuing for the congestion control application, a related but weaker property is still worthy of consideration.

- **Invariance Under Linear Transformation (ILT):** Let u be a vector of utility functions and v be a transformed vector such that $v_i(x) = \alpha\, u_i(x) + \beta$. Let $U(u(x))$ be a SWF, where $u(x) = (u_i(x_i))$ is the vector of session utilities for rate vector x. We

say that a SWF is *invariant under a linear transformation* if, for any two rate vectors x and y,

$$U(u(x)) \geq U(u(y)) \Rightarrow U(v(x)) \geq U(v(y))$$

for any values of α, β. In words, the SWF induces the same preference relation for u and v.

The ILT property builds on the assertion that individual utility functions are unique up to a linear transformation, saying that aggregate preferences, too, are invariant under such a transformation. We will see shortly that under some definitions of multicast session utility the ILT property is satisfied, while under others it is not.

4.2 Sender- and Receiver-Oriented Utility Functions

We now formally define sender- and receiver-oriented concepts of session utility. Consider a single-rate multicast session s with rate x and receiver set $\mathcal{R}$ with size R. In the sender-oriented approach, session utility function is a single concave increasing function of the session rate $u_s(x)$.

$$U_{snd} = u_s(x) \tag{22}$$

In the receiver-oriented approach, each receiver $i \in \mathcal{R}$ has a utility function $u_i(x)$, which is concave and increasing. The session utility function is the sum of receiver utilities.

$$U_{rcv} = \sum_{i \in \mathcal{R}} u_i(x) \tag{23}$$

We can convert these definitions into an alternate form by making two assumptions. First, we assume that all receivers have identical utility functions.

$$u_i(x) = u_r(s) \quad \forall\, i \in \mathcal{R}$$

Second, in the limit of a single receiver, both sender- and receiver-oriented utility functions should reduce to the same standard unicast utility function up to a linear transformation.

$$u_s(x) = u_r(s)$$

These assumptions allow us to express both types of session utility functions as the product of a base utility function $u(\cdot)$ and a scaling function $f(\cdot)$. The base utility function, $u(\cdot)$ depends only on the session rate and is concave and increasing. It can be thought of as the utility function of a session with a single receiver. The scaling function $f(\cdot)$ depends on the number of receivers in the session. It must be monotonic in its argument, although it need not be strictly increasing.

For a sender-oriented definition of session utility, $f(R) = \kappa$, where κ is a constant.

$$U_{snd}(x, R) = \kappa\, u(x) \tag{24}$$

For a receiver-oriented definition, $f(R) = \kappa\, R$, where κ is a constant.

$$U_{rcv}(x, R) = \kappa\, R\, u(x) \tag{25}$$

It is possible entertain definitions of session utility different from those we present here. We choose these because they are commonplace and mathematically tractable. One obtains equation (24) by treating all sessions equivalently, regardless of the number of receivers. Equation (25) reflects the idea that multicast session utility is itself a social welfare function, representing the aggregate utility of the receiver set. Under our assumptions, this equation is equivalent to the sum of receiver utilities—a simple and commonly used social welfare function

4.3 The Session-Splitting Problem

In Section 4.2, we identified two alternative definitions of multicast session utility based on sender- or receiver orientation. Now we consider these two definitions in more detail and determine which makes sense in the context of congestion control. We begin by attempting to capture the effect of flexible group membership using an optimization-based approach. Golestani and Sabnani [31] observe that if receivers in a session can be dropped and reassigned to a different session in response to congestion, it is often desirable to split a multicast group into subgroups with different rates. One can think of this form of congestion control as an approximation of multi-rate multicast that does not violate the constraint of having a single rate per session.

The presence of additional sessions in the network after splitting may increase contention on existing bottlenecks or even create new bottlenecks. Thus not all ways of splitting a session lead to an overall improvement in received rates. Ideally, one would like to find a way to split the session that offers higher rate to some receivers without reducing the rates of any others. A less ideal, but perhaps still tolerable split might reduce some receivers' rates but improve the utilization of the network and allow many more receivers to receive at a higher rate. In economic terms, we would like to find a way of splitting a session that maximizes utility. In this section, we consider the use of sender- and receiver-oriented social welfare functions to determine whether splitting a session will improve aggregate utility. We will show that the only way to obtain a "reasonable" solution to this optimization problem is by using a receiver oriented definition of utility.

We seek a definition of aggregate utility that captures our intuitions about when splitting the multicast session is a good idea and when it is not. More precisely, we want to know if it is reasonable to use sender- or receiver-oriented social welfare function to determine whether splitting a session will improve aggregate utility. The choice of sender- or receiver-oriented utility as well as the form of the base utility function may affect aggregate utility. However, for a fixed choice of these factors, we expect the SWF to be well-defined for all possible ways of splitting the session. Finally, the desirability of splitting should be insensitive to a linear transformation of the base utility function. If this were not the case, an arbitrary rescaling of utility could determine whether splitting a session is preferred over not splitting. We will observe that the desirability of splitting maintains this invariance in the case of a receiver-oriented SWF but not in the case of a sender-oriented one.

We begin by formalizing the session splitting problem in terms of utility maximization. In the session-splitting problem, we have a network (N, L) with link capacities $C = (c_l, l \in L)$. A set of receivers $R \subset N$ could be served by one or more multicast sessions with source $s \in N - R$. We assume that the number and rates of all other sessions in the network are fixed. Capacities in C thus represent the available capacity for multicast sessions serving receiver set R. Each session's rate is limited by its most constrained receiver, that is, by the receiver with the lowest link capacity along the path between it and the source. If this bottleneck link is not shared by all of the receivers, then it may be possible to split the

session into two or more sessions yielding a higher rate to some receivers.

Splitting the session is equivalent to partitioning the receiver set into disjoint subsets $P = \{P_1, P_2, \ldots, P_N\}$. We will use $\mathcal{P}$ to denote the set of all possible partitions of R. Each partition in $\mathcal{P}$ represents one possible way to divide the receiver set into sessions. Each element of a partition represents a subset of R to be served by a different session. Rates may vary among sessions, but all receivers within a session must receive at a single rate. Computing the rates for each session is, itself, a non-trivial problem since some links will be shared by more than one session. There are many possible mechanisms for determining session rates. One example is the greedy algorithm suggested by Rubenstein, Kurose and Towsley [32] to achieve max-min fairness among the sessions.

For our purposes, it is sufficient to assume that we have some deterministic mechanism to perform this rate assignment, which we model as a rate allocation function

$$X(P, i)$$

$$X : \mathcal{P} \times \mathbb{Z}^+ \to \mathbb{R}$$

Given a partition P and an index i, the rate allocation function returns the rate of the session serving P_i.

The session-splitting problem requires us to find a partition that maximizes the aggregate utility of the network. Recall that the optimization-based approach defines aggregate utility as the (possibly weighted) sum of all session utilities. Given a network (N, L), capacities C, receiver set $R \subseteq N$, sender s, rate allocation function $X(\cdot, \cdot)$, base utility function $u(\cdot)$, and scaling function $f(\cdot)$, the optimal splitting is a partition that solves

$$\max_{P \in \mathcal{P}} U(P; f, u, X)$$

where

$$U(P; f, u, X) = \sum_{i=1}^{|P|} f(|P_i|)\, u(X(P, i))$$

is the aggregate utility function. We can choose the scaling function from equations (24) and (25) to solve this problem for sender- and receiver-oriented definitions of session utility.

The aggregate utility function defines a partial ordering over $\mathcal{P}$. In economic terms, this ordering is the social preference relation over all possible partitions of the receiver set.

As explained in Section 4.1, it is customary to regard utility functions as unique up to a linear transformation. A reasonable restriction, therefore, is only to allow social preference relations that remain invariant under a linear transformation of the base utility function, as captured by the following axiom, similar to the ILT property in Section 4.1:

Axiom 1 *Let $f(\cdot)$ be a fixed scaling function and $X(\cdot,\cdot)$ be a fixed rate allocation function. For any base utility fuction $u(\cdot)$, let $v(\cdot)$ be another base utility function such that*

$$v(x) = \alpha\, u(x) + \beta$$

where α and β are constants. Then for all $P, Q \in \mathcal{P}$,

$$\begin{aligned} & U(P; f, u, X) \geq U(Q; f, u, X) \\ \iff \quad & U(P; f, v, X) \geq U(Q; f, v, X) \end{aligned}$$

Theorem 1 *Let $f_{snd}(R) = \kappa$ and $f_{rcv}(R) = \kappa R$ be sender- and receiver-oriented scaling functions. For any base utility function u and rate allocation function $X(\cdot,\cdot)$, the aggregate utility function $U(\cdot; f_{rcv}, u, X)$ satisfies Axiom 1, while $U(\cdot; f_{snd}, u, X)$ does not. Furthermore, Axiom 1 can only be satisfied using the scaling function $f(R) = f_{rcv}(R)$.*

Proof: For convenience in the following discussion we denote $U(P; f, u, X)$ as $U(P)$ and $U(P; f, v, X)$ as $V(P)$. Let x^P and x^Q be vectors of allocated rates for all sessions in partitions P and Q, respectively. Since the axiom must hold for any choice of P and Q, let us choose them so that $|P| \neq |Q|$. Under a sender-oriented definition, we have the following aggregate utilities

$$U(P) = \kappa \sum_{i=1}^{|P|} u(x_i^P)$$

$$U(Q) = \kappa \sum_{j=1}^{|Q|} u(x_j^Q)$$

Without loss of generality, assume that $U(P) \geq U(Q)$. Transforming $u(x)$, we obtain new values for aggregate utility.

$$V(P) = \kappa \sum_{i=1}^{|P|} [\alpha\, u(x_i^P) + \beta] = \kappa\, (\alpha\, U(P) + \beta\, |P|)$$

$$V(Q) = \kappa \sum_{j=1}^{|Q|} [\alpha\, u(x_j^Q) + \beta] = \kappa\,(\alpha\, U(P) + \beta\, |Q|)$$

It is clear that we can always chose a value for β such that $V(P) < V(Q)$. Thus, the social preference relation is not invariant under any linear transformation of the base utility function. An immediate consequence is that if one accepts that Axiom 1 is indeed an appropriate requirement for any "reasonable" definition of aggregate utility, then our sender-oriented utility definition is not "reasonable".

Under a receiver-oriented definition,

$$U(P) = \kappa \sum_{i=1}^{|P|} |P_i|\, u(x_i^P)$$

$$U(Q) = \kappa \sum_{j=1}^{|Q|} |Q_j|\, u(x_j^Q)$$

Transforming $u(x)$, we obtain

$$V(P) = \kappa \sum_{i=1}^{|P|} |P_i|\, [\alpha\, u(x_i^P) + \beta] = \kappa\,(\alpha\, U(P) + \beta\, R)$$

$$V(Q) = \kappa \sum_{j=1}^{|Q|} |Q_j|\, [\alpha\, u(x_j^Q) + \beta] = \kappa\,(\alpha\, U(Q) + \beta\, R)$$

In this case, $V(P)$ is, itself, a linear transformation of $U(P)$, which preserves the social preference relation regardless of the choice of α and β. It is straightforward to extend of this reasoning to show that any scaling function other than the receiver-oriented function fails to satisfy Axiom 1. [11] We must therefore either conclude that our receiver-oriented definition of utility is the only "reasonable" definition of session utility or reject Axiom 1 as a requirement for reasonability.

□

[11] In the general case of scaling function $f(R)$, we obtain a condition for satisfying Axiom 1 that $\sum_{i=1}^{|P|} f(P_i) = \sum_{j=1}^{|Q|} f(Q_i)$. This condition along with the constraint that the sum of session sizes is constant for any partition leads to the conclusion that $f(R)$ must be a multiple of its argument.

5 Consequences of Receiver-Oriented Utility Functions

In section 4.3 we argued that receiver oriented session utility functions are an appropriate model for multicast session utility in the session splitting problem. In this section, we return to the original congestion control problem and determine whether using receiver-oriented utility functions leads to fair sharing of bandwidth between unicast and multicast sessions. We rewrite the network optimization problem (1-2) as

$$\max_{x=(x_s,s\in S)} \sum_s \kappa R_s\, u(x_s) \tag{26}$$

$$\text{subject to } \sum_{s\in S(l)} x_s \leq c_l, \quad \forall l \in L \tag{27}$$

The Kuhn-Tucker conditions for optimality are

$$\kappa R_s \frac{du}{dx}(x_s) = \sum_{l\in L(s)} \lambda_l \tag{28}$$

$$\lambda_l(x^l - c_l) = 0,\ (x^l - c_l) \leq 0 \tag{29}$$

where the λ_l are Lagrange multipliers or link prices and $x^l = \sum_{s\in S(l)} x_s$ is the aggregate rate seen at link l. As before, we also write $\lambda^s = \sum_{l\in L(s)} \lambda_l$ as the total session price seen by session s.

From the first Kuhn-Tucker condition (28), we observe that the use of receiver-oriented utility functions creates a bias in favor of sessions with large numbers of receivers. To see this, note that

$$\frac{du}{dx}(x_s) = \frac{\lambda^s}{\kappa R_s} \tag{30}$$

The optimal rate for session s, x_s^* is given by

$$x_s^* = u'^{-1}\left(\frac{\lambda^s}{\kappa R_s}\right) \tag{31}$$

Equation (30) states that, at optimality, the a session's marginal utility should be proportional to its price divided by the number of receivers. We refer to this ratio λ^s/R_s as the *effective session price*. The optimal rate can therefore be obtained by taking the inverse of

the marginal utility function as shown in equation (31). Since U_s is concave, u' is a strictly decreasing function of x and its inverse is also a decreasing function. For a fixed session price, a session with a larger number of receivers has a lower effective session price and thus receives a higher rate. We refer to this effect as "tyranny of the majority" (ToM).

ToM is a source of unfairness against unicast flows since multicast flows with the same total session price will receive a higher rate. However, the fact that multicast sessions tend to use more links than unicast sessions, particularly as the number of receivers becomes large, means that the session price λ^s for a multicast flow is likely to be higher than that of a unicast session. To understand the fairness properties of rate allocations under receiver oriented utility functions we must determine whether the growth in price associated with the scaling of a multicast tree is sufficient to limit the effect of ToM as more receivers are added.[12]

6 Effect of Multiple Points of Congestion

In the previous section, we saw that ToM and the scaling of multicast trees have opposite effects. As we will see shortly, these effects are not equal in strength. The effect of ToM is likely to be the stronger of the two, allowing sessions with more receivers to receive a greater share of bandwidth. Whether we choose to accept this form of controlled unfairness or introduce a correction, we require a more precise understanding of the interaction of the two effects.

We begin our exploration with a simple example using a modified complete binary tree topology such as the one shown in Fig. 2. The tree is modified to include a single link from the session source to what would ordinarily be the root. This single link resides at depth 0 of the tree, with depth increasing as we move down the tree. The number of links at depth d of this tree is 2^d. Consider such a multicast tree of maximum depth D with 2^D receivers,

[12]If one holds that improving the rate of many receivers at the expense of a few is reasonable, giving a larger share of bandwidth to larger groups may not seem unfair. We take the position that a bounded bias in favor of large groups is a defensible form of "controlled unfairness" but that there must be a mechanism to prevent starvation of unicast flows.

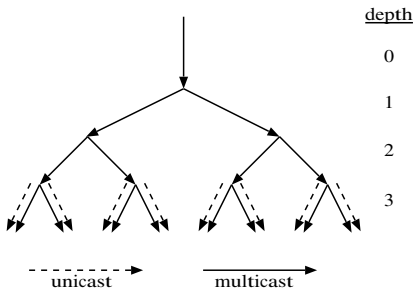

Figure 2: A modified binary multicast tree of depth 3 with a sharing depth of 3.

one at each leaf node. The total capacity available on each link is c. The tree is shared by 2^D 1-hop unicast sessions, distributed evenly across all the links at a particular depth, d. We will refer to d as the *sharing depth*. Thus, if $d = D$, then each link at the bottom of the tree will be shared with one additional session. In general, for a given choice of d, each link at the sharing depth will be shared with 2^{D-d} additional sessions. Since each link of the tree has identical capacity, each link at depth d will be a bottleneck and contribute a non-zero price to the total tree price.[13] By varying d, we can control the number of points of congestion in the tree.

For such a regular topology, the basic optimization problem (12) can be solved directly. Table 1 shows the optimal rates for the multicast and unicast sessions in a binary tree of maximum depth 3, using a receiver oriented utility function with $\kappa = 1$ and a logarithmic base utility function. For comparison, the max-min fair rates are also shown. [14] Figure 3 show the convergence of Low and Lapsley's synchronous algorithm to the optimal rates.

As we vary the sharing depth d, the total price, hence the optimal rate, of the multicast session remains constant, while the prices and rates of the unicast sessions vary. In other words, on any congested link, the multicast session receives a fixed fraction—one half—of available bandwidth with the remaining capacity shared evenly among all unicast sessions sharing that link. At smaller d, more unicast sessions share each congested link, each receiv-

[13]Since link prices are associated with Lagrange multipliers in the solution of the basic optimization problem, an equivalent interpretation is that each bottleneck link will have an active capacity constraint, hence a non-zero multiplier.

[14]In this example, the max-min fair rates for multicast and unicast sessions are equal.

Sharing Depth	Optimal Rate		Max-Min Rate
	Multicast	Unicast	
0	$c/2$	$c/16$	$c/9$
1	$c/2$	$c/8$	$c/5$
2	$c/2$	$c/4$	$c/3$
3	$c/2$	$c/2$	$c/2$

Table 1: Optimal and max-min fair rates for a binary tree of depth 3 with 8 1-hop unicast sessions sharing links at various depths. Note that the max-min fair rates are the same for both types of sessions in this example.

ing a smaller portion of a fixed available capacity. This result may seem somewhat surprising in view of earlier remarks about multiple points of congestion leading to higher session prices. The explanation for this effect is that we conserve the total number of shared sessions in this example. Thus the level of congestion on each bottleneck (i.e. the number of sessions sharing it) increases as the number of congested links decreases, resulting in a higher price per congested link. Despite this unrealistic feature of the model, it is instructive to consider the invariance of multicast session rate in greater detail.

Let us generalize our model by considering a complete tree of degree k and depth D, modified as before to include a single link at depth 0. Each link of the tree has capacity c. We retain the receiver-oriented definition of session utility, but allow an arbitrary base utility function $u(x)$. The tree has a receiver at each leaf—k^D receivers in total—and shares links with k^D unicast sessions—k^{D-d} on each of k^d links at level d. Let $\mathbf{x} = (x_s)$ be the vector of session rates, where x_0 and $x_1, \ldots, x_N$ are the rates of the multicast and unicast sessions, respectively, and $N = k^D$. Shadow prices are represented by a vector of multipliers $\lambda = (\lambda_1, \ldots, \lambda_L)$, where $L = 2^d$.

For a particular choice of sharing depth, we can now form the Lagrangian for the basic optimization problem (1).

$$\mathscr{L}_d(\mathbf{x}, \lambda) = k^D\, u(x_0) + \sum_{i=1}^{R} u(x_i) - \sum_{j=1}^{L} \lambda_j\, g_j(\mathbf{x}) \tag{32}$$

where g_j are the capacity constraints for the shared links.

$$g_j(\mathbf{x}) \;=\; x_0 + \sum_{l=L(j-1,k,d,D)}^{L(j,k,d,D)-1} x_l - c \leq 0 \tag{33}$$

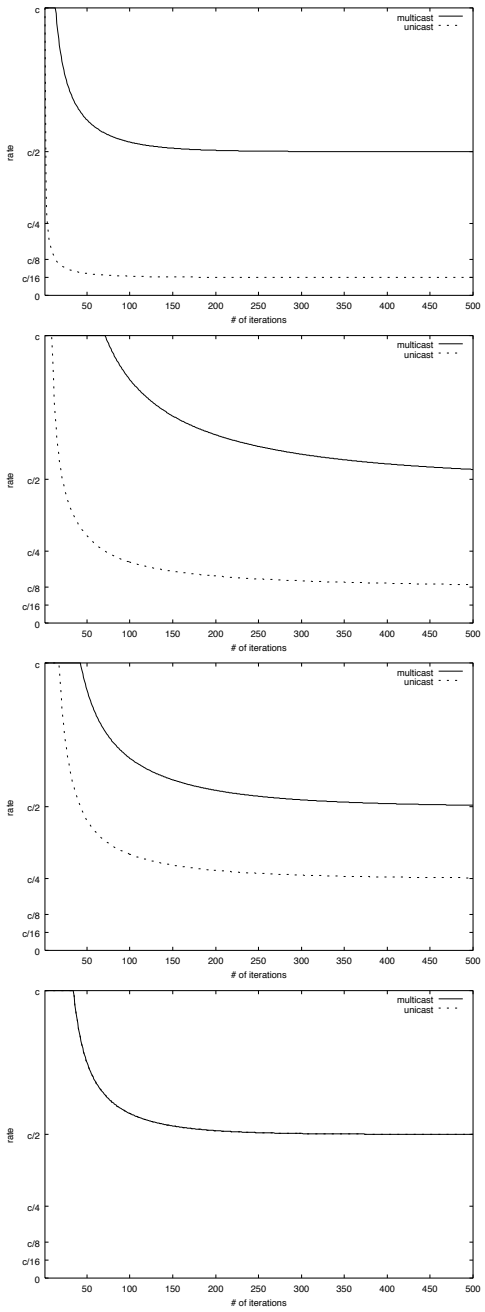

Figure 3: Convergence of Low and Lapsley's synchronous algorithm to optimal rates for the binary tree shown in Fig. 2 for sharing depths of 0, 1, 2, and 3. In each case, the algorithm is run for 500 iterations with a stepsize parameter $\gamma = 0.03$ and capacity $c = 1$.

$$L(j,k,d,D) \quad = \quad j\,k^{D-d}$$

We use the symmetry of the tree topology to reduce the problem to three variables: the multicast session rate x_m, the unicast session rate x_i, and the shadow price of a congested

link λ. We rewrite the link capacity constraint

$$g_j(\mathbf{x}) = g(\mathbf{x}) = x_m + k^{D-d}\, x_i - c \tag{34}$$

The first-derivative conditions are

$$\begin{aligned}
\frac{\partial \mathscr{L}_d}{\partial x_m} &= k^D\, u'(x_m) - k^d\, \lambda = 0 && (35)\\
\frac{\partial \mathscr{L}_d}{\partial x_i} &= u'(x_i) - \lambda = 0 && (36)\\
\frac{\partial \mathscr{L}_d}{\partial \lambda_j} &= x_m + k^{D-d}\, x_i - c = 0, && (37)
\end{aligned}$$

where we have further exploited symmetry of the capacity constraints to set $\lambda_j = \lambda$.

Solving these equations for the logarithmic base utility function $u(x) = \log(x)$ gives

$$x_m = \frac{c}{2}, \quad x_i = \frac{c}{2\, k^{D-d}}, \quad \lambda = \frac{2\, k^{D-d}}{c} \tag{38}$$

This result is a generalized version of Table 1. We observe the following facts:

- At the system optimum, the multicast session receives rate $x_m = c/2$. This result is independent of the tree depth D, the sharing depth d, and, perhaps most surprising, the tree degree k.

- The invariance of the optimal multicast rate is a direct result of the choice of a logarithmic base utility function. As we will see, this property does not hold for other utility functions.

- The remaining capacity on the shared links is split evenly among the sharing unicast sessions. Since the number of sharing sessions is k^{D-d}, the optimal unicast rate depends on D, d and k.

- The total price seen by the multicast session is

  $$\lambda\, k^d = \frac{2\, k^D}{c},$$

  which is independent of the sharing depth. Under a receiver-oriented definition of session utility, this price is divided by the number of receivers to obtain the effective session price. Thus, effective session price is independent of d, D and k under a logarithmic utility function.

Since the invariance we have observed appears to derive from a special choice of utility function, it is interesting to explore the behavior as we modify the functional form. We can combine the first-derivative conditions to derive the following optimality condition:

$$u'(x_m) = \frac{1}{k^{D-d}}\, u'\left(\frac{c - x_m}{k^{D-d}}\right), \tag{39}$$

Equation (39) relates the marginal utility function $u'(x)$ to the function $u'^*(x) = a\, u'(a\,(c - x))$ obtained when we flip u' about the line $x = c$ and scale both the argument and the result by the same factor a. Any point at which these two functions intersect satisfies the optimality condition. Note that $u'(x)$ is the derivative of a concave and strictly increasing utility function, and therefore must be strictly decreasing. Thus, $u'(x)$ and $u'^*(x)$ intersect in exactly one point, establishing the uniqueness of the solution. Observe also that the scaling factor a is of the form $1/k^{D-d} \leq 1$. Scaling the argument of $u'(c-x)$ compresses the function along the horizontal axis and moves the point of intersection to the left, while scaling its result compresses the function along the vertical axis and moves the point of intersection to the right.

Figure 4 shows how the points of intersection vary as a function of a in a binary tree for three choices of base utility function: $u(x) = \log(x)$, $u(x) = -1/x$ and $u(x) = -(-\log(x))^{\alpha}$. The first two functions are the now familiar logarithmic and MPD utility functions. The third one is shown by Kelly to yield max-min fairness in the limit as $\alpha \to \infty$ [13].[15] In all three graphs, the single decreasing functions is $u'(x)$, the first derivative of the base utility function, and the family of increasing functions are $u'^*(x)$ for decreasing a (increasing $D-d$). The points where $u'^*(x)$ intersects $u'(x)$ give the optimal rates for the multicast session as a fraction of available capacity.

As established above, the intersection point is invariant and equal to $c/2$ for logarithmic utility. The intersection point is also fixed at $c/2$ when $a = 1$ for all three functions, corresponding to a sharing depth equal to the maximum tree depth. In both the MPD and max-min fair utility functions, however, the intersection point moves to the left as a decreases, converging to the max-min fair rate. That is, as the sharing depth moves closer to the top of the tree, the number of bottleneck links *decreases* while the price on each

[15] In our experiments, we take α to a reasonably high power. ($\alpha = 250$)

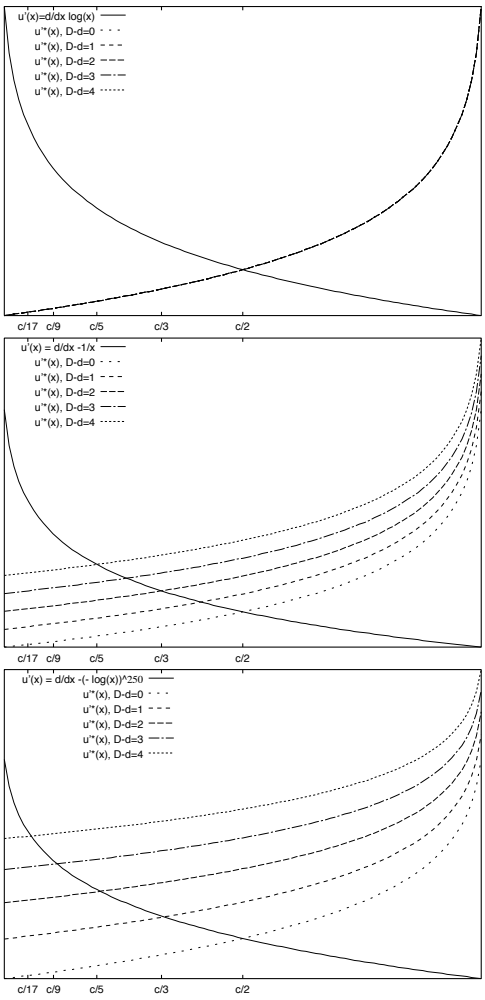

Figure 4: Figure showing the effect on the optimal allocation for a binary multicast tree as we vary the sharing depth. These graphs show three different marginal utility functions, $u'(x)$ along with their transformations $u'^*(x)$ for various choices of $D - d$ with the y-axis shown in log scale. The x-coordinate of the points of intersection give the optimal session rates as a fraction of available capacity. Max-min fair allocations for different values of $D-d$ are indicated along the x-axis. The figure shows that the logarithmic utility function (top) gives the multicast session half the available bandwidth regardless of the number of sharing unicast sessions, whereas the max-min fair utility function (bottom) splits bandwidth evenly among all sessions on the shared link regardless of the number of receivers. The MPD utility function (center) represents a compromise between these two extremes.

congested link increases and the multicast session receives a smaller fraction of the available bandwidth.

6.1 An Alternate Utility Function for Max-Min Fairness

Under the definition of max-min fairness for single-rate multicast [32], the multicast session must share bandwidth equally with all sessions on its most congested link. Thus, in the max-min fair allocation, $x_m = c/(k^{D-d}+1)$. In the case of Kelly's max-min fair utility function, we see that the optimal rates indeed coincide with the max-min fair allocations, indicated by the tick-marks along the x-axis in Fig. 4. We have observed that the points of intersection converge to these values as we transform the logarithmic utility function into max-min fair utility function by increasing the exponent α. Demonstrating this convergence formally is somewhat difficult.

We can establish a similar result for a family of utility functions that includes both the logarithmic and MPD utility functions and also yields max-min fairness as a limiting case. Consider the family of utility functions $u(x)$ with first derivatives $u'(x) = 1/x^{\alpha+1}$. Such functions include $u(x) = \log(x)$, $u(x) = -x^{-\alpha}/\alpha$. Members of this family are mathematically tractable since the functions $u'(x)$ are homogeneous, satisfying

$$u'(t\,x) = t^{-r}\,u'(x) \tag{40}$$

where $r = \alpha + 1$. We can simplify the optimality condition of (39).

$$\left(\frac{x_m}{c - x_m}\right) = a^{(1-r)/r} \tag{41}$$

As a further simplification, we can express the multicast rate as a fraction, p, of available capacity, $x_m = p\,c$. Solving for p, we get

$$p = \frac{1}{1 + a^{(1-r)/r}}. \tag{42}$$

In the limit of large α, p converges to the max-min fair allocation.

$$\lim_{r\to\infty} \frac{1}{1 + a^{(1-r)/r}} = \frac{a}{1+a} = \frac{1}{k^{D-d}+1}. \tag{43}$$

Following Kelly's example in [13], we can prove that $u(x) = -x^{-\alpha}/\alpha$ always gives max-min fairness in the limit $\alpha \to \infty$, by providing an absolute priority to smaller flows. For two rates such that $x_{s^*} < x_s$,

$$\frac{u'(x_{s^*})}{u'(x)} = \left(\frac{x_s}{x_{s^*}}\right)^{\alpha+1} \to \infty \quad \text{as } \alpha \to \infty$$

Kelly proposed the utility function $u(x) = -(-\log(x))^{\alpha}$ to establish proportional fairness and max-min a two extremes in a range of fairness definitions described by the collection of utility functions obtained for $\alpha \geq 1$. As α becomes larger, these utility functions provide an increasing priority to smaller flows. Our result is a significant extension of Kelly's observation because it locates TCP-fairness (as defined by the MPD utility function) within this range of possible fairness definitions.

7 Bounded Unicast Fairness

In Section 6, we observed that a multicast session was able to obtain a higher rate than unicast sessions sharing the same bottleneck links. We showed that this unfairness is bounded in the presence of multiple points of congestion. However, this result exploited features of an idealized multicast session topology. Adopting a somewhat more realistic model in this section, we investigate whether the same type of bounded unfairness is possible in a more general setting with receiver-oriented utility functions. We also consider whether there is any multicast utility function that allows a strictly equal split of shared bottleneck bandwidth between a multicast an unicast session.

Adopting the fairness objective proposed by Handley, Floyd and Whetten [4]—that the algorithm be provably fair relative to TCP in the steady state, we define a generalized notion of TCP fairness. We say that a multicast session utility function $U(x;r) = f(R)\,u(x)$ is *strictly unicast-fair* if the optimal rate for the multicast session is the same as would be obtained by a unicast session with utility function $u(x)$ along the most congested source-to-receiver path in the multicast tree. This definition is equivalent to TCP-fairness in the case where $u(x) = -1/x$, the MPD utility function. We will also consider a more relaxed notion of fairness, proposed by Wang and Schwartz [33] allowing bounded unfairness between multicast and unicast sessions.

We first show that neither sender nor receiver oriented multicast utility functions lead to strict unicast-fair allocations and derive a result suggesting that strict fairness is difficult to achieve under any definition of session utility. Consider the modified star network topology shown in Figure 5. A single multicast session with source node s and receivers $\{1,\ldots,R\}$ shares the network with R unicast sessions, one from s to each receiver. Link l_0 from the source to the central node is shared by all sessions and has effectively infinite capacity. Each link l_i from the center to receiver i is shared by the multicast session and one unicast session. Link l_1 is the bottleneck link, with capacity $\beta\,c$, where $\beta < 1$ and c is the capacity of all other links l_i, $i > 1$. Receiver 1 is the most congested receiver in the multicast session.

We give the unicast sessions the MPD utility function $u(x) = -1/x$. The multicast function has utility function $u(x;R) = f(R)\,u(x)$. Let x_m be the rate of the multicast

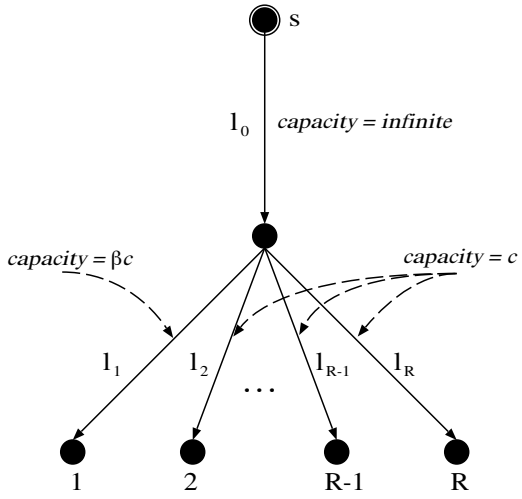

Figure 5: A multicast tree with a modified star topology. Receiver 1 is most congested.

session and x_i be the rate of the unicast session to receiver i. A strictly tcp-fair allocation would split the bandwidth on l_1 equally between x_m and x_1, $x_m = x_1 = \beta\, c/2$. We can substitute this rate into the optimality conditions of the optimization problem (26-27) to determine the appropriate scaling function $f(R)$ that will lead to the tcp-fair allocation, obtaining

$$f(R) = 1 + \frac{(R-1)\,\beta^2}{2\sqrt{2}\,(2-\beta)^2} \tag{44}$$

This result shows that tcp-fairness can be achieved in the optimization-based framework by maximizing a weighted sum of utilities with weights given by a scaling function $f(R)$. However, the presence of β, a topological parameter, in the scaling function suggests that the correct scaling function depends on topological properties of the network.

Let us consider another example, shown in Figure 6. We again have a modified star topology, but with equal capacities on all links $l_i, i > 0$. Link l_0 has finite capacity c_0, which is larger than the sum of all downstream link capacities. In addition to the R unicast sessions from the previous example, there is a single unicast session with rate x_0 using only link l_0 between the source and the center. The purpose of this one-hop session is to consume any bandwidth on l_0 not used by the other sessions, ensuring that l_0 is fully utilized and has a non-zero price λ_0. By symmetry, all two-hop unicast sessions will have the same rate x_i, and all links in $\{l_1, \ldots, l_R\}$ have the same price λ_i. We first write the conditions of optimality

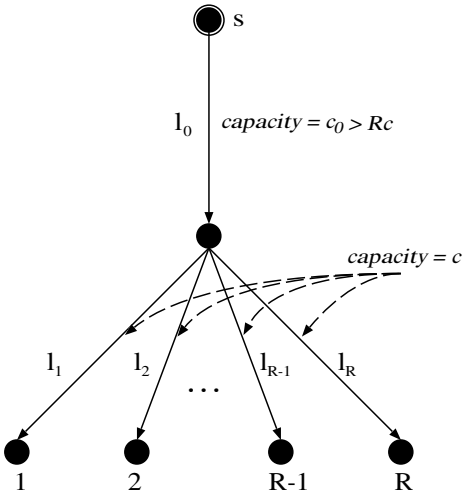

Figure 6: A multicast tree with a modified star topology. All receivers are equally congested.

for this problem for a tcp-like base utility function.

$$1/x_m^2 = \frac{R\lambda_i + \lambda_0}{f(R)} \tag{45}$$

$$1/x_0^2 = \lambda_0 \tag{46}$$

$$1/x_1^2 = \lambda_i + \lambda_0 \tag{47}$$

We define the *independent price component for receiver i* (denoted p) to be the fraction of the path price from the source to a multicast receiver that is incurred on the link not shared by any other receivers in the same session. This component is the fraction of the path cost directly attributable to the presence of receiver i.

$$p = \frac{\lambda_i}{\lambda_i + \lambda_0} \tag{48}$$

The requirement of unicast fairness places a condition on the ratio x_i/x_m. Expressing this ratio in terms of the independent price component p and the scaling function $f(R)$, we have.

$$\frac{x_i}{x_m} = \sqrt{\frac{R}{f(R)}\,p + \frac{1}{f(R)}\,(1-p)} \tag{49}$$

For strict unicast fairness, $x_i/x_m = 1$. We can easily see that neither a sender oriented nor receiver oriented scaling functions will provide tcp-fairness over all session sizes R. For example, a sender oriented scaling function $f(R) = 1$ will favor the unicast session as R

becomes large, while a receiver oriented function $f(R) = R$ will favor the multicast session. By setting $x_i/x_m = 1$ and solving for $f(R)$, we see that the unicast fair scaling function depends on a topological parameter, the independent price component.

$$f(R) = Rp + (1 - p) \tag{50}$$

We now consider a generalized version of the previous example with no explicitly defined network topology. Consider a network containing a set of links $\mathcal{L}$. The network is shared by two sessions v and w, which have rates x_v and x_w, respectively. Each session uses a subset of links in the network and session w only uses a proper subset of links that are also used by v. Formally, $L(w) \subset L(v) \subseteq \mathcal{L}$. The sessions have R_v and R_w receivers with $R_v > R_w$. We assume that the path to the most constrained receiver in both v and w is the same and is therefore entirely contained in $L(w)$. The Lagrangian for the optimization problem is.

$$\begin{aligned} \mathscr{L}(x;\lambda) &= f(R_v)\,u'(x_v) + f(R_w)\,u'(x_w) + \\ &\quad \sum_{l \in L(w)} \lambda_l (x_v + x_w - c_l) + \\ &\quad \sum_{l \in L(v) - L(w)} \lambda_l (x_v - c_l) \end{aligned} \tag{51}$$

From the Kuhn-Tucker conditions, we derive an optimality condition on the ratio of marginal utilities.

$$u'(x_v)/u'(x_w) = \frac{f(R_w)}{f(R_v)} \frac{\lambda^v}{\lambda^w} \tag{52}$$

$$\tag{53}$$

where

$$\lambda^{v|w} = \sum_{l \in L(v|w)} \lambda_l \tag{54}$$

Consider the family of base utility functions satisfying $u'(x) = -1/x^\alpha, \;\; \alpha \geq 1$, introduced in Section 6.1. Recall that this family includes both the MPD and logarithmic utility functions. The ratio of session rates is

$$x_w/x_v = \left(\frac{f(R_w)}{f(R_v)} \frac{\lambda^v}{\lambda^w} \right)^{1/\alpha} \tag{55}$$

In a strictly tcp-fair allocation, the ratio $x_w/x_v = 1$. From equation (55), it is clear that the actual value of this ratio depends on both the choice of scaling function and the ratio λ^v/λ^w.

It is also apparent that the ratio $x_w/x_v = 1$ approaches 1 in the limit as $\alpha \to \infty$. Thus, the exponent α offers one way to control unfairness for any choice of scaling function; increasing it moves the resulting rate allocation closer to max-min fairness.

Strict unicast fairness could be achieved by exploiting a scaling law relating the total price of a multicast session to its number of receivers. Chuang and Sirbu propose such a law for static multicast costs [34, 35] with the form

$$\lambda^s \propto R_s^k \tag{56}$$

The authors empirically evaluate the scaling exponent k, finding its value to be constant over a wide range of network topologies.[16] This law assumes, however, that link costs in the network are static. To be applicable for the purposes of congestion control, such a scaling law would have to be established for dynamically changing prices that reflect link congestion. If such a scaling law can be found, then strict unicast fairness would result from a multicast session utility function

$$U_s(x) \propto R_s^{-k}\, u(x).$$

We leave the search for such a scaling law as direction for future research, but note here that, as presented in Section 4.3 the sum of session utilities under such a multicast utility function would not be invariant under a linear transformation of $u(x)$.

In the absence of a scaling law, strict unicast fairness appears to be difficult to achieve in the optimization-based framework without adjusting the scaling function in response to topological parameters. It is worth considering a relaxed version of tcp fairness, such as *essential fairness* proposed by Wang and Schwartz [33]. Essential fairness is evaluated on the restricted topology shown in Figure 7. In this topology, a multicast session with R receivers shares the network with a number of tcp sessions. There are $m_i \geq 0$ unicast sessions between the sender and each receiver i. In the terminology of Wang and Schwartz, each path from the sender to a receiver is called a virtual link and denoted by L_i. Each

[16] The surprising universality of this result was subsequently explained by Phillips, Shenker and Tangmunarunkit [35]. The power law observed by Chuang and Sirbu approximates a more complex scaling behavior exhibited by graphs with exponential expansion, a property found in many real and artificially generated networks.

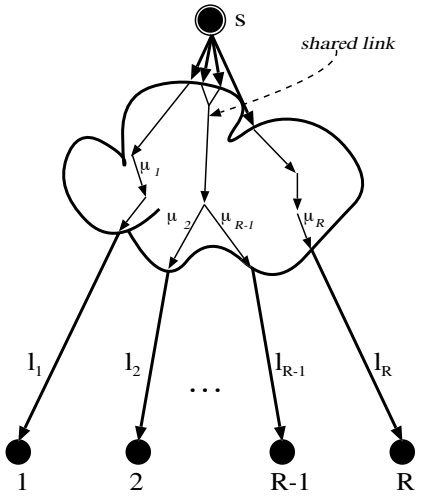

Figure 7: A multicast tree with a modified star topology. All receivers are equally congested.

virtual link has a physical bottleneck link with available capacity μ_i. The *soft bottleneck* of the multicast session, is defined as the link $L_{sb} = \arg\min_i \mu_i/(m_i + 1)$. An essentially fair rate allocation satisfies the bounds

$$a\, x_{sb} < x_m < b\, x_{sb} \tag{57}$$

$$a \leq b < R \tag{58}$$

where x_m is the rate of the multicast session and x_{sb} is the rate of a unicast session along L_{sb}.

Setting up the optimization problem for this restricted topology and taking the Kuhn-Tucker conditions yields an optimality condition relating x_m and x_{sb}.

$$x_m = \left(f(R)\, \frac{\lambda_{sb}}{\lambda^m} \right)^{1/\alpha} x_{sb} \tag{59}$$

where λ_{sb} is the price of the soft bottleneck link and λ^m is the total price seen by the multicast session.

To get the lower bound of x_m, we observe that the lowest possible price for the multicast session occurs when each virtual link has bandwidth equivalent to the soft bottleneck. In this case, $\lambda^m = R\, \lambda_{sb}$.

$$x_m \geq \left(\frac{f(R)}{R} \right)^{1/\alpha} x_b \tag{60}$$

To get the upper bound, we consider the link with the highest available capacity, L_c. Analogous to the soft bottleneck, $L_c = \arg\max_i \mu_i/(m_i+1)$. Link L_c is the lowest priced link in the multicast session, with price λ_c. The upper bound of x_m is obtained when all links other than the soft bottleneck contribute a price of λ_c.

$$x_m \leq \left(f(R) \frac{\lambda_{sb}}{(R-1)\,\lambda_c + \lambda_{sb}} \right)^{1/\alpha} x_{sb} \tag{61}$$

For fixed R, x_m increases as λ_c decreases. In the limit, we have

$$x_m \leq f(R)^{1/\alpha}\, x_{sb} \quad \text{as} \quad \lambda_c \to 0 \tag{62}$$

Putting these bounds together, we have

$$\left(\frac{f(R)}{R} \right)^{1/a} x_{sb} \leq x_m \leq f(R)^{1/\alpha}\, x_{sb} \tag{63}$$

If multicast session utility is defined using the TCP-like MPD utility functions along with a receiver oriented scaling function $f(R) = \kappa\, R$, we have

$$\kappa\, x_{sb} \leq x_m \leq \kappa \sqrt{R}\, x_{sb} \tag{64}$$

Since $\sqrt{R} < R$ for $R > 1$, this result suggests that a receiver oriented utility function with appropriately chosen $\kappa \leq 1$ combined with a minimum potential delay base utility function will lead to essentially fair sharing between multicast sessions and tcp-like unicast sessions for all R.

It is also interesting to observe that for a logarithmic base utility function ($u(x) = \log(x)$) and $\alpha = 1$, the upper bound becomes

$$\kappa\, x_{sb} \leq x_m \leq \kappa\, R\, x_{sb} \tag{65}$$

In this case, the upper bound is less fair to unicast sessions than with $a = 2$ and we must choose κ strictly less than 1 to satisfy the essential fairness property of (58).

8 Conclusion

This paper presented an optimization based scheme for multicast congestion control based on utility maximization. Appealing to the underlying economic theory behind this approach

to congestion control, we proposed the use of a receiver oriented definition of session utility. By considering the incentive to split multicast sessions into smaller sessions, we showed that only receiver oriented utility functions ensure that the optimal solution of the utility maximization problem remains invariant under a linear transformation of the utility scale. We identified two sources of unfairness that arise when maximizing the sum of receiver oriented utility functions, one favoring unicast sessions and one favoring multicast. When these two effects are combined, a net unfairness results that favors sessions with many receivers over sessions with few, with unicast sessions faring worst of all. This unfairness is bounded, however, and the tightness of the bound depends on the form of the base utility function. When comparing multicast sessions against TCP, where TCP sessions are modeled using the Minimum Potential Delay utility function, we can limit the multicast session rate to no more than $\sqrt{R}$ times the rate of a unicast session between the source and the worst-case receiver. While we have found it difficult to achieve strict fairness between unicast and multicast traffic, we argue that bounded unfairness is a reasonable goal, particularly as it provides an incentive to use multicast by rewarding larger groups.

Much future work still needs to be done in this area. Although single-rate multicast with a single source is an important class of multicast traffic, many multicast applications do not fall into this category. An important problem, therefore, is extending the optimization-based techniques to multi-rate multicast, where a single session may support receivers with different rates and to multicast sessions with multiple sources. This problem is challenging because the global optimization problem does not admit an easy decomposition into per-session optimization problems in the multi-rate case.

We also still lack a practical protocol for single-rate multicast based on our approach. While we have shown that existing distributed algorithms can be adapted to single-rate multicast with only minor modifications, there are a number of technical issues that will have to be addressed by any practical implementation. Our approach differs from many multicast congestion control schemes in that its fairness properties are not dependent on accepting feedback from a restricted set of receivers. However, other protocols restrict feedback not only for fairness, but also to eliminate feedback implosion and redundancy. An optimization-based protocol must find other mechanisms for dealing with these problems. Existing optimization-

based mechanisms for unicast rely on packet-marking techniques to communicate link prices to end hosts. The end-to-end nature of a unicast path allows the receiver to infer the total session price from the fraction of marked packets arriving in a measurement interval. In a multicast session each receiver sees only marks generated along one ene-to-end path through the tree, which only account for a fraction of the session price. Feedback is thus required from all receivers to determine the total session price. Redundancy arises when a marked packet traverses a router and is copied on more than one downstream interface, effectively multiplying the upstream path price and inflating the true cost of the tree. We believe that both redundancy and feedback problems can be addressed using general purpose network services for multicast transport [36] to provide feedback aggregation and a sophisticated mark-forwarding mechanism.

9 Acknowledgments

The authors are grateful to Steven Low and Rayadurgam Srikant for introducing us to this topic and discussing their work with us. We are particularly grateful to Steven Low for suggesting the multicast fomulation we used in this paper.

References

[1] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet," *IEEE/ACM Transactions on Networking*, August 1999.

[2] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.

[3] K. K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ecn) to ip," January 1999.

[4] M. Handley, S. Floyd, and B. Whetten, "Strawman specification for tcp friendly (reliable) multicast congestion control," Tech. Rep., Reliable Multicast Research Group, December 1998.

[5] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, "Equation-based congestion control for unicast applications," Tech. Rep. TR-00-003, International Computer Science Institute, Berkeley, CA, Mar. 2000.

[6] S. Bhattacharyya, D. Towsley, and J. Kurose, "The loss path multiplicity problem for multicast congestion control," Technical Report UM-CS-1998-076, University of Massachusetts, Amherst, Computer Science, 1998.

[7] D. F. Ferguson, *The Application of Microeconomics to the Design of Resource Allocation and Control Algorithms*, Ph.D. thesis, Columbia Univ., 1989.

[8] A. Orda, R. Rom, and N. Shimkin, "Competitive routing in multiuser communication networks," *IEEE/ACM Transactions on Networking*, vol. 1, no. 5, pp. 510–521, Oct. 1993.

[9] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang, "Pricing in computer networks: Motivation, formulation, and example," *ACM/IEEE Transactions on Networking*, vol. 1, no. 6, pp. 614–627, Dec. 1993.

[10] J. K. MacKie-Mason and H. R. Varian, "Pricing congestible network resources," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, pp. 1141–1149, September 1995.

[11] N. Semret, *Market Mechanisms for Network Resource Sharing*, Ph.D. thesis, "Columbia Univ.", 1999.

[12] S. H. Low, "Optimization flow control for the internet: a brief survey," Submitted for pulication, 1999.

[13] F.P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. volume 8, pp. 33–37, 1997.

[14] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[15] R.J. Gibbens and F.P. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, pp. 1969–1985, 1999.

[16] S. Kunniyur and R. Srikant, "End-to-end congestion control: utility functions, random losses and ecn marks," in *Proc. INFOCOM*, 2000.

[17] S. H. Low and D. E. Lapsley, "Optimization flow control, i: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, December 1999.

[18] S. Athuraliya, D. Laspsley, and S. Low, "An enhanced random early marking algorithm for internet flow control," in *Proc. INFOCOM*, 2000.

[19] Jeonghoon Mo and Jean Walrand, "Fair end-to-end window-based congestion control," IEEE/ACM Transactions on Networking, 1999.

[20] S.J. Golestani and S. Bhattacharyya, "A class of end-to-end congestion control algorithms for the internet," in *Proc. ICNP'98*, 1998.

[21] P.Key, D. McAuley, P. Barham, and K. Laevens, "Congestion pricing for congestion avoidance," Tech. Rep. MSR-TR-99-15, Microsoft Research, 1999.

[22] P. Madden, *Concavity and Optimization in Microeconomics*, Basil Blackwell, 1986.

[23] F. S. Hillier and G. J. Lieverman, *Introduction to Mathematical Programming*, McGraw-Hill, 2 edition, 1995.

[24] D. Bertsekas and R. Gallagher, *Data Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1992.

[25] L. Massoulie and J. Roberts, "Bandwidth sharing: Objectives and algorithms," in *Proc. INFOCOM*, 1999.

[26] D. Bansal and H. Balakrishnan, "Tcp-friendly congestion control for real-time streaming applications," Tech. Rep., Massachusetts Institute of Technology, May 2000.

[27] S. Shenker, "Fundamental design issues for the future internet," *IEEE J. Selected Areas Comm.*, vol. 13, pp. 1176–1188, 1995.

[28] J. Hirshleifer and D. Hirshleifer, *Price Theory and Applications*, Prentice Hall, 6 edition, 1997.

[29] T. Sandholm, *Multiagent Systems: Modern Introduction to Distributed Artificial Intelligence*, chapter Distributed Rational Decision Making, pp. 201–258, MIT Press, 1999.

[30] K. J. Arrow, *Social Choice and Individual Values*, Yale Univ. Press, 2 edition, 1963.

[31] S. J. Golestani and K. K. Sabnani, "Fundamental observations on multicast congestion control in the internet," in *Proc. INFOCOM*, 1999.

[32] D. Rubenstein, J. Kurose, and D. Towsley, "The impact of multicast layering on network fairness," in *Proc. SIGCOMM 99*, 1999.

[33] H. A. Wang and M. Schwartz, "Achieving bounded fairness for multicast and tcp traffic in the internet," in *SIGCOMM '98*, 1998.

[34] J. Chuang and M. Sirbu, "Pricing multicast communications: A cost-based approach," in *Proc. INET'98*, 1998.

[35] G. Phillips, S. Shenker, and H. Tangmunarunkit, "Scaling of multicast trees: Comments on the chuang-sirbu scaling law," in *Proc. SIGCOMM-99*, 1999.

[36] B. Cain and D. Towsley, "Generic multicast transport services: Router support for multicast applications," in *Proc. Networking 2000*, 2000.

[37] T. Speakman et al., "Pgm reliable transport protocol specification," Internet Draft.

A An ECN-Based Protocol

In this section, we briefly describe a way to modify the Random Early Marking (REM) protocol proposed by Athuraliya, Low and Lapsley to support the modified congestion control algorithm we have described. REM implements Low and Lapsley's algorithm by setting an explicit congestion notification (ECN) bit in packet headers to mark packets passing through

congested resources. An individual link marks packets with a probability m_l that increases exponentially the current link price.

$$m_l(t) = 1 - \phi^{-p_l(t)} \tag{66}$$

Because of the exponential form of m_l, the end-to-end marking probability for a a unicast session, denoted m^s, is a function of the total session price.

$$m^s(t) = 1 - \phi^{-\sum_{l \in L(s)} p_l} \tag{67}$$

The session price is thus given by

$$p^s(t) = -\log_\phi(1 - m^s(t)) \tag{68}$$

In the REM protocol, the receiver estimates m_s as the fraction of marked packets in the most recently received N packets. This estimate is periodically polled to compute a price estimate, which is then used by the source to regulate its rate.

In the following discussion, we will assume the presence of the REM packet marking discipline—that is, each link sets the ECN bit according to (66). To develop a REM-like protocol for single-rate multicast, we must find a way to estimate the total tree price using marks collected by receivers scattered throughout the tree. This problem is difficult for two reasons. First, an individual receiver will only see marks generated on links lying along an end-to-end path between it and the source. Thus no single receiver will be able to estimate the marking probability for the entire tree and the total tree price must be determined by collecting information from all of the receivers. As explained below, we advocate the use of active services within the network to collect this information while avoiding a feedback implosion at the source. A second difficulty is that marks generated on shared links can be counted by more than one receiver since a marked packet arriving at a branching point will be forwarded on more than one downstream link. This double counting must either be prevented or accounted for in the price estimate calculation. We will consider two approaches, one that requires the network to prevent double counting and one that does not.

Collecting feedback from the entire receiver set may not be an unreasonable burden for the source in small groups, but becomes impractical as the number of receivers grows large.

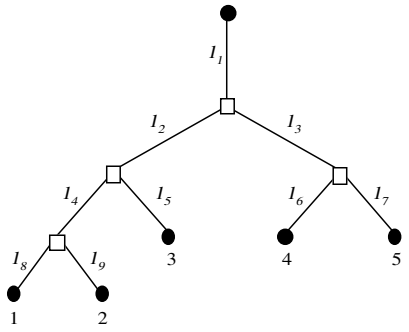

Figure 8: A multicast tree with five receivers and four internal nodes.

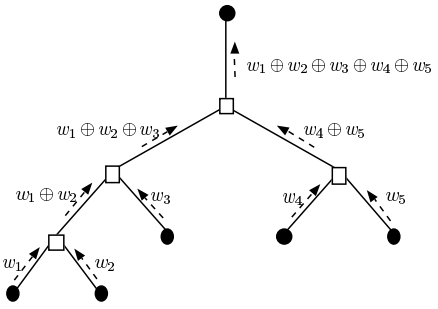

Figure 9: Example of a feedback aggregation service. Internal nodes combine messages arriving from downstream receivers into a single message sent towards the source.

We propose to take advantage of general purpose network services for multicast transport such as GMTS [36] to address this problem. The particular service we require is *feedback aggregation*. A simple form of feedback aggregation works for a set of possible feedback messages S and a binary aggregation operator $\oplus$ with the constraint that S must be closed and associative under $\oplus$. Consider a multicast group with n receivers. Each receiver i has a message $w_i \in S$ to send to the source. Suppose that the source needs to compute a value $w \in S$ such that

$$w = w_1 \oplus w_2 \oplus \ldots \oplus w_n \tag{69}$$

Routers in the multicast tree provide the feedback aggregation service by computing the "partial sums" of messages from downstream receivers and forwarding the result upstream as shown in Fig. 9. For the aggregation to work, receivers must be able to send their messages toward the source along the reverse multicast tree. This reverse route for feedback is typically established using *session path messages* [37, 36] to inform each node in the tree of the address of its nearest upstream neighbor. We will show how a general purpose aggregation service can be used by a REM-like congestion control protocol.

While a feedback aggregation service allows the source to efficiently collect information from the entire receiver set, a more fundamental problem concerns what should be computed by the individual receivers and by the sender. One possibility is for receivers to send information about the fraction of marked packets received and have the sender compute the total marking probability and the total price for the tree. Let us assume that receivers estimate the marking probability by measuring the fraction of marks seen in the N most recently received packets. It is not sufficient for receivers to simply send the fraction of marked packets. Let M_i be the event that a packet is marked by link l_i and let $\Pr(M_i)$ be the probability of this event occurring. Similarly, let M^r be the event that a packet is marked somewhere on the path between the source and receiver r, occurring with probability $\Pr(M^r)$. Since a single marking bit is available per packet, the detection of a mark by receiver r must be interpreted to mean that at least one link along the packet's path marked the packet,

$$\Pr(M^r) = \Pr\left(\bigcup_{l \in L(s,r)} M_l\right), \tag{70}$$

where $L(s,r) \subseteq L(s)$ is the set of links in the path from the source to receiver r in session s.

Consider the marking probabilities estimated by the two leftmost receivers in the tree shown in Fig. 8. Receiver 1 estimates a marking probability of

$$\Pr(M^1) = \Pr(M_1 \cup M_2 \cup M_4 \cup M_8), \tag{71}$$

while receiver 2 estimate a marking probability of

$$\Pr(M^2) = \Pr(M_1 \cup M_2 \cup M_4 \cup M_9) \tag{72}$$

Suppose both receivers send their estimated probabilities to their immediate upstream router. We would like this router to aggregate these values to yield an estimate of

$$\Pr(M^1 \cup M^2) = \Pr(M^1) + \Pr(M^2) - \Pr(M^1 \cap M^2) \tag{73}$$

Unfortunately, the router does not have enough information to perform this operation. Specifically, the router has no way to estimate the cross term in equation (73), which represents the fraction of losses on links shared by both receivers.

Suppose now that instead of sending the fraction of marked packets, each receiver sends a vector of N bits representing the N most recently received packets, indicating a marked packet by placing a 1 at the corresponding bit. Routers can easily aggregate these messages by using a bitwise-OR as the aggregation operator. As a result of this aggregation taking place at each level of the tree, the source receives a single vector with bits set for each packet that was marked on route to at least one receiver. The total marking probability for the entire tree can be estimated as the fraction of bits in this vector that are set.

While this approach can in principle use the feedback aggregation mechanism to allow the source to efficiently estimate the tree price, it suffers from a drawback as the size of the tree grows. For large trees, the total marking probability, that is, the probability that a packet is marked on at least one link in the tree, is likely to be very close to one. Since the marking probability in (67) approaches one asymptotically, it is very difficult to get a precise estimate of the tree price for values close to one. For large trees, one would have to increase the number of packets N over which the measurement is taken, leading to a less responsive protocol and a larger size for feedback messages.

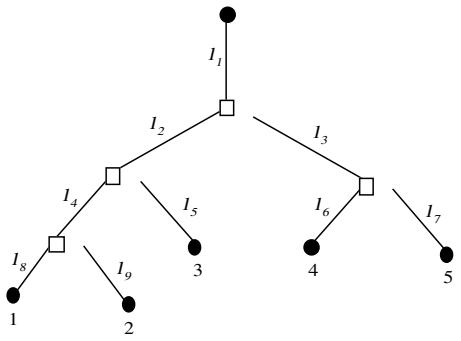

Figure 10: Partitioning the tree of Fig. 8 into disjoint paths for the purposes of price estimation. This partition is generated by a mark forwarding rule in which each internal node forwards a mark on the leftmost downstream interface only.

An alternative approach requires the network to prevent the double-counting of marked packets by ensuring that marks generated on a given link reach exactly one receiver. Conceptually, this approach divides the entire multicast tree into disjoint end-to-end paths, one for each receiver. Each receiver then estimates the price along its assigned path. The total tree price can be computed using feedback aggregation with binary addition as the aggregation operator. Routers prevent double counting by resetting the ECN bit of a marked packet for all but one of the forwarded copies. The router must use a deterministic rule, such as always preserving the mark on the lowest indexed downstream interface to ensure that the same receiver sees all marks generated on a particular link. Fig. 10 shows the resulting partition for a simple mark-forwarding rule. Because marks are forwarded deterministically along a single branch of the tree, marks generated on links that are shared by many receivers will be assigned to exactly one of those receivers. Furthermore, the longest path assigned to any receiver will be comparable in length to a unicast path, which means that marking probability used to estimate the price along this path will not be close to 1.

The choice of a mark collection scheme thus poses a tradeoff. The bit-vector approach assumes no additional capabilities inside the network other than the REM packet marking discipline, but suffers from a loss of precision and responsiveness as the multicast tree scales in size. We can address these limitations by having receivers estimate prices for disjoint paths in the tree, however, we must introduce additional complexity in routers to ensure that marks are delivered to the appropriate receiver. We leave a quantitative characterization of these

tradeoffs as an area of future work.