Measurement-Based Multipath Multicast

Tuna Güven, Richard J. La, Mark A. Shayman, Bobby Bhattacharjee
University of Maryland, College Park, MD 20742, USA
Email: {tguven@eng, hyongla@eng, shayman@eng, bobby@cs}.umd.edu

***Abstract*— We propose a measurement-based routing algorithm to load balance intradomain traffic along multiple paths for multiple multicast sources. Multiple paths are established using application-layer overlaying. The proposed algorithm is able to converge under different network models, where each model reflects a different set of assumptions about the multicasting capabilities of the network. The algorithm is derived from simultaneous perturbation stochastic approximation and relies only on noisy estimates from measurements. Simulation results are presented to demonstrate the additional benefits obtained by incrementally increasing the multicasting capabilities.**

I. Introduction

Multicast traffic over the Internet is growing steadily with increasing number of demanding applications including Internet broadcasting, video conferences, data stream applications [1] and web-content distributions. Many of these applications require certain rate guarantees, and demand that the network be utilized more efficiently than with current approaches to satisfy the rate requirements. Traffic mapping (load balancing) is one particular method to carry out traffic engineering, which deals with the problem of assigning the traffic load onto pre-established paths to meet certain requirements [2]. Our focus in this paper is to scrutinize the effects of load balancing the multicast traffic in an intradomain network.

There is a limited amount of existing work on multipath multicast routing. In [3], the authors propose a solution to optimally distribute the traffic along multiple multicast trees. However, the solution covers the case when there is only one active source in the network. In addition, it is assumed that the gradient of an analytical cost function is available, which is continuously differentiable and strictly convex. As discussed in [4], these assumptions may not be reasonable due to the dynamic nature of networks. As discussed later, we will relax all these assumptions in our solution. In another set of work, solutions based on network coding [5], [6], [7], are proposed [8], [9]. Even though they approach the problem under a more general architecture, practicality of these solutions is limited due to the unrealistic assumption that the network is lossless as long as the average link rates do not exceed the link capacities. Moreover, a packet loss is actually much more costly when network coding is employed since it potentially affects the decoding of a large number of other packets. In addition, any factor that changes the min-cut max-flow value between a source and a receiver requires the code to be updated at every node simultaneously, which brings high level of complexity and coordination.

In this paper, we propose a distributed optimal routing algorithm to balance the load along multiple paths for multiple multicast sessions. Our measurement-based algorithm does not assume the existence of the gradient of an analytical cost function and is inspired by the unicast routing algorithm based on Simultaneous Perturbation Stochastic Approximation (SPSA) [10]. In addition, we address the optimal multipath multicast routing problem in a more general framework than having multiple trees. We consider different network models with different functionalities. With this generalized framework, our goal is to examine the benefits observed by the addition of new capabilities to the network beyond basic operations such as storing and forwarding. In particular, we will first analyze the traditional network model without any IP multicasting functionality where multiple paths are established using (application-layer) overlay nodes. Next, we consider a network model in which multiple trees can be established. Finally, we will look at the generalized model by allowing receivers to receive multicast packets at arbitrarily different rates along a multicast tree. Such an assumption potentially creates a complex bookkeeping problem since source nodes have to make sure each receiver gets a distinct set of packets from different trees while satisfying the rates associated with each receiver along each tree. However, using a specific source coding called Digital Fountain codes [11], we show that this problem can be overcome in an efficient way, and allows us to have an additional degree of freedom in the optimization problem.

II. Digital Fountain Coding

The original application area of Fountain codes [11], [12] is the reliable transmission of data over the Internet as an alternative to the TCP/IP retransmissions as the Internet can be modelled as an erasure channel. The rationale behind using Digital Fountain codes as opposed to classic block codes (*e.g.,* Reed-Solomon codes) for erasure correction is that in an (N,K) Reed-Solomon code one must estimate the erasure probability and choose the code rate $R = K/N$ before transmission. Furthermore, Reed-Solomon codes have the disadvantage that they are practical only for small K, N. On the other hand, Digital Fountain codes are rateless in the sense that the number of encoded packets that can be generated from the source message is potentially limitless; the number of encoded packets generated can be determined on the fly. Regardless of the statistics of the erasure events in the channel, one can send as many encoded packets as needed in order for the decoder to recover the source data.

A decoding algorithm for a Digital Fountain Code is an algorithm that can recover the original K input symbols from any set of M output symbols with a high probability where M is very close to K and the decoding time is close to linear in K. Raptor codes [12] are examples of such Fountain Codes with linear time encoders and decoders for which the probability of decoding failure converges to zero polynomially fast in the number of input symbols.

The Fountain codes are quite useful in the context of multipath multicast routing in the sense that a source node can generate as many distinct encoded symbols as required and forward packets along multiple paths according to rate requirements, and this will guarantee that each receiver successfully receives the whole multicast stream from any distinct set of M coded symbols. This allows us to send multicast traffic to each destination at different rates along different paths (e.g., along different branches of a multicast tree) without having to keep track of which packets are sent along which path.

III. Model

Consider a network that consists of a set of unidirectional links $\mathcal{L} = \{1, \ldots, \mathrm{L}\}$ and a set of nodes $\mathcal{N} = \{1, \ldots, \mathrm{N}\}$. There are S sessions. Each session can be either a unicast session or a multicast session. The set of source nodes is denoted by $\mathcal{S} \subset \mathcal{N}$, and for each source $s \in \mathcal{S}$ let D^s be the set of destination nodes for the session.

We consider several network models based on different sets of assumptions on the capability of the underlying network to capture the performance and cost trade-off.

A. General Routing Framework - Overlay Architecture

We use an application overlay architecture to create multiple paths between a source node and either a unicast destination node or a multicast receiver node. We refer to them as destination nodes. In all models considered we assume that simple device(s) (e.g., hosts with network processors) are attached to a subset of network routers that are carefully selected inside an intradomain network.[1] These are called *overlay nodes*, and the set of overlay nodes is given by $\mathcal{O}$.

In order to reach a destination node through an overlay path, its source node attaches an additional IP header to the packet and forwards the packet to the selected overlay node using the underlying routing protocol. The overlay node strips the extra IP header used by the application overlay from the packet and forwards it to the destination node utilizing the underlying routing protocol. In principle a source node can forward any fraction of packets to a destination node through any of the available overlay nodes, creating multiple paths to a destination node. Note that this approach does not require any changes to the underlying IP routing protocol.

Denote the set of overlay nodes used to create *alternate* paths between a source $s \in \mathcal{S}$ and its destination nodes in D^s by $O^s \subset \mathcal{O}$. Assuming every overlay node in O^s is used to create an alternate path to every destination node $d \in D^s$, there are $|O^s|$ paths available to each destination node, where $|O^s|$ denotes the cardinality of O^s.[2] Define $N_s = |O^s|$. For each $s \in \mathcal{S}$ and $d \in D^s$ let $x^s_{o,d}$ be the rate the source node s sends packets to d through an overlay node $o \in O^s$. Also, let x^s_o be the total rate at which an overlay node o receives packets from source s.

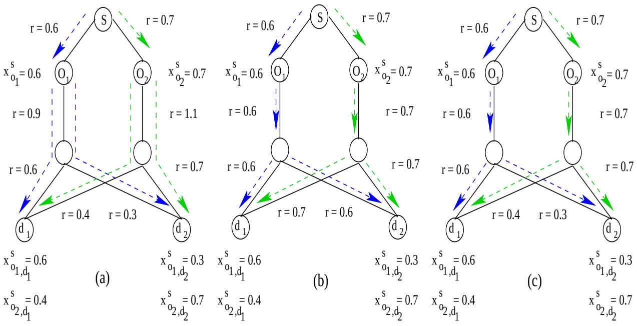

Fig. 1. The link rates under various network models.

As discussed in Section I, without adopting a special coding scheme, if the rates $x^s_{o,d}$ are not identical for all destinations, the source must not only ensure that each destination receives packets at the intended rate, but also maintain careful bookkeeping to prevent delivery of duplicate packets to a destination. This problem can be solved by using, for example, a Digital Fountain code. This allows us to reduce our problem to that of rate assignment $x = (x^s_{o,d}, s \in \mathcal{S}, o \in O^s, d \in D^s)$, which is the focus of this paper. We assume that the overlay nodes can copy packets. Hence, the sources need to deliver only a single copy of the packets to an overlay node, and the overlay node acts as the surrogate source for those packets. Under this assumption, the rate x^s_o to an overlay node o satisfies

$$x^s_o = \max_{d \in D^s} x^s_{o,d} \ . \qquad (1)$$

This means that, depending on the assumed network model, an overlay node forwards all or a portion of the packets from the source to each of the destinations at the specified rate $x^s_{o,d}$.

The answer to the question of how to forward packets from overlay nodes to destinations depends on the network model adopted. For instance, if it is assumed that the network does not have any IP multicast functionality (Network Model-I), overlay nodes should copy the packets for each destination and forward them in a unicast manner as shown in Fig. 1(a). On the other hand, if IP multicasting is available, then packets are forwarded to the destination nodes through a multicast tree rooted at the overlay node which is created by an intradomain multicast algorithm such as DVMRP [13]. Under this network model, without additional intelligence at the IP routers (Network Model-II), even when $x^s_{o,d}$ are not identical, all destinations $d \in D^s$ will be forced to receive packets at the same rate due to the fact that ordinary IP multicast routers can only copy and forward the packets. Hence, they are not capable of forwarding packets to different branches at different rates. As a result, the path rates for each destination will be $x^s_o (= \max_{d \in D^s} x^s_{o,d})$. Fig. 1(b) explains this situation. Clearly, this may cause a receiver to receive packets at a rate larger than the intended rate. However, as we will show shortly, our algorithm can observe this through measurements and lead to a rate allocation minimizing such redundancy. In fact, at the operating point $x^\star$ we have ${x^s_{o,d}}^\star = x^{s\star}_o$ for all $d \in D^s$.

Suppose that the routers possess additional intelligence and are capable of forwarding packets to downstream branches at different rates that are specified by the network (Network Model-III). Then, it is possible to forward packets to each destination d at the selected rate $x^s_{o,d}$ as

[1] Note that this is similar to regular application-layer overlays with the exception that overlay nodes are not necessarily located at the end-hosts.

[2] Note that source node itself is in the set O^s, denoting the default path.

shown in Fig. 1(c). This allows source nodes to exercise more fine-grained control over the rates $x_s = (x^s_{o,d}, o \in O^s, d \in D^s)$.

Note that under these models, overlay nodes can be viewed as content delivery servers that store a portion of the original content to be distributed. The objective is to distribute the content to these servers in such a way that the usage of network resources is optimized. Our goal is to minimize the total network cost defined to be the summation of all link costs in the network, by balancing the traffic load among multiple paths. However, the relationship between the rate assignments and the link loads depends on the adopted network model, which effectively alters behavior of the algorithm.

B. Link Loads

In this subsection we describe how the link loads are computed based on the rate allocations $x = (x_s, s \in \mathcal{S})$.

1) Network Model-I: This model represents the traditional IP network with routers without IP multicast functionality. We assume that packets are encoded using a Digital Fountain code at the source. A source node forwards the packets to overlay nodes at the required rate, and overlay nodes create a unicast session and forward packets to each destination at the specified rate $x^s_{o,d}$.

Let $V^{n_1}_{n_2} \subset \mathcal{L}$ be the set of links in the default path from node n_1 to node n_2. Given a rate assignment x, the link loads $x^l, l \in \mathcal{L}$, are given by

$$x^l = \sum_{s \in S} \left(\sum_{o \in O^s : l \in V^s_o} x^s_o + \sum_{o \in O^s} \left(\sum_{d \in D^s : l \in V^o_d} x^s_{o,d} \right) \right) \quad (2)$$

This model is referred to as NM-I in Section V.

2) Network Model-II: Under Network Model-II the routers are IP multicast capable. We assume that each overlay node $o \in O^s$ creates a multicast tree for forwarding packets. However, due to the lack of additional required intelligence the data rate to all receivers is the same and is given by $x^s_o = \max_{d \in D^s} x^s_{o,d}$.

Under this model the load of link l can be written as

$$x^l = \sum_{s \in \mathcal{S}} \left(\sum_{o \in O^s : l \in V^s_o} x^s_o + \sum_{o \in O^s : l \in T^s_o} x^s_o \right) \quad (3)$$

where T^s_o is set of links in the multicast tree rooted at overlay node o and serving destination nodes in D^s.

This model is referred to as NM-IIa in Section V.

3) Network Model-III: In this model, in addition to the IP multicast capability we also assume that each router is capable of forwarding packets onto each branch at a different rate. We refer to these routers as "smart" routers to distinguish them from the routers used in the previous model. This is shown in Fig. 1(c). Under this model a source s can select the individual rates $x^s_{o,d}$ independently for each destination, and each destination $d \in D^s$ will receive the intended rate $x^s_{o,d}$ instead of $\max_{d' \in D^s} x^s_{o,d'}$ as under Network Model-II. This allows the network operator more flexibility in rate assignment and to better exploit the existence of multiple paths through overlay nodes, while making use of multicast nature of the traffic at the same time. Hence, the link rates can be written as

$$x^l = \sum_{s \in S} \left(\sum_{o \in O^s : l \in V^s_o} x^s_o + \sum_{o \in O^s} \max_{d \in D^s : l \in \hat{V}^o_d} x^s_{o,d} \right) \quad (4)$$

Here $\hat{V}^o_d$ denotes the set of links along the path from overlay node o to destination d in the multicast tree, which may be different from the path provided by the underlying routing protocol. Under this model it is necessary to adopt a special coding scheme, such as Digital Fountain codes, in order to ensure that all destinations can recover the transmitted data as explained in Section I. We assume that a suitable coding scheme is adopted. We will refer to this model as NM-III while presenting the experiments.

IV. Optimization Framework

We formulate the problem of rate assignment x as an optimization problem, where the objective function is the sum of link costs. Link cost is a function of the total rate traversing the link and is given by $C_l(x^l), l \in \mathcal{L}$, where x^l is used to denote the rate through link l. These link cost functions are assumed to be convex, but we do not require them to be differentiable. The optimization problem can be stated as follows:

$$\min_x C(x) = \min_x \textstyle\sum_{l \in \mathcal{L}} C_l(x^l) \quad (5)$$

$$\text{s.t. } \textstyle\sum_{o \in O_s} x^s_{o,d} = r^s + \epsilon^s, \forall s \in \mathcal{S}, d \in D^s \quad (6)$$

$$x^s_{o,d} \geq \nu, \ \forall d \in D^s, o \in O^s, s \in \mathcal{S} \quad (7)$$

where r^s is the total input traffic rate of source s, ν is an arbitrarily small positive constant [3] and ϵ^s is the required additional rate of the coding scheme for a receiver to successfully decode the encoded data.

The optimization problem in (5) can be viewed as a natural generalization of [10] from unicast traffic sources to multicast sources. We can use a Stochastic Approximation (SA) (*e.g.,* [14], [15]) technique to solve (5). The general constrained SA is similar to the well-known gradient projection algorithm, in which at each iteration $k = 0, 1, \ldots$, the variables are updated based on the gradient. However, with an SA method the gradient vector $\nabla C(k)$ is replaced by its approximation $\hat{g}(k)$. The approximation is typically obtained through noisy measurements of $C(x)$ around $x(k)$. Under appropriate conditions, $x(k)$ can be shown to converge to the solution of (5), denoted by $x^\star$, as will be shown in the next subsection.

One particular method used for gradient estimation is called *Simultaneous Perturbation* (SP). When SP is employed, all elements of $x(k)$ are randomly perturbed simultaneously to obtain two measurements $y(\cdot)$. The i-th component of $\hat{g}(k)$ is computed from

$$\hat{g}_i(k) = \frac{y(x(k) + c(k)\Delta(k)) - y(x(k) - c(k)\Delta(k))}{2c(k)\Delta_i(k)} \quad (8)$$

where $c(k)$ is some positive scalar, and the vector $\Delta(k) = (\Delta_1(k), \Delta_2(k), ..., \Delta_m(k))$ of random perturbations for SP needs to satisfy certain conditions to be specified shortly. SA algorithms that use SP for gradient estimation are called Simultaneous Perturbation Stochastic

[3] For instance, some of the control packets may be routed along different paths available between the source and destination nodes.

Approximation (SPSA). As shown in [10], SPSA has significant advantages over traditional gradient estimation methods such as Finite Difference Stochastic Approximation (FDSA).

Due to the nature of the problem, the multicast routing problem given by (5) - (7) can be decomposed into several subproblems at the sources. In order to find the solution to (5) we propose to run an SPSA algorithm at each source node independently in a distributed fashion. Let Θ_s denote the feasible set that satisfies (6) and (7), and let $\Pi_{\Theta_s}[\zeta]$ denote the projection of a vector ζ onto the feasible set Θ_s using the Euclidean norm. At time $k = 0, 1, \ldots$, each source s updates its rate $x_s(k)$ according to

$$x_s(k+1) = \Pi_{\Theta_s}[x_s(k) - a_s(k)\hat{g}_s(k)] \tag{9}$$

where $a_s(k) > 0$ is the step size, and $\hat{g}_s(k)$ is the approximation to the gradient vector $\nabla C_s(k) = (\partial C(x(k))/\partial x^s_{o,d}, o \in O^s, d \in D^s)$ given by the SPSA algorithm with the following form:

$$\begin{aligned} &\hat{g}_{s,i}(k) \\ &= \frac{N_s}{N_s - 1} \frac{y_s(\Pi_\Theta[x(k) + \mathbf{c}(k)\Delta(k)]) - y_s(x(k))}{c_s(k)\Delta_{s,i}(k)} \quad (10) \\ &= \frac{N_s}{N_s - 1} \frac{(C^+(k) + \mu_s^+(k)) - (C^-(k) - \mu_s^-(k))}{c_s(k)\Delta_{s,i}(k)} , \end{aligned}$$

where $C^-(k) = C(x(k))$, $C^+(k) = C(\Pi_\Theta[x(k) + \mathbf{c}(k)\Delta(k)])$, $c_s(k)$ is a positive scalar used for perturbation, and $\mathbf{c}(k)$ is a diagonal matrix composed of block diagonal entries $\{\mathbf{c}_s(k), s \in \mathcal{S}\}$ where $\mathbf{c}_s(k) = c_s(k) \cdot I_s$ with I_s being the $(N_s \cdot |D^s|) \times (N_s \cdot |D^s|)$ identity matrix. The measurement noise terms $\mu_s^+(k)$ and $\mu_s^-(k)$, and the value of $c_s(k)$ can be different for each source. Due to this reason $\mathbf{c}(k)$ is a diagonal matrix as opposed to a scalar. In addition, we have an extra multiplicative factor $\frac{N_s}{N_s-1}$ in (10) compared to the standard SA. This is due to the projection of $x_s(k)+c_s(k)\Delta_s(k)$ to Θ_s for all $s \in S$ using L_2 projection while calculating $\hat{g}_s(k)$.

Note that each source node may have different step sizes $a_s(k)$. This allows sources to respond to the network state in an independent manner. For instance, this formulation allows the case where source nodes start running the algorithm at different times. However, we assume that sources update their rates every iteration once they start running the algorithm. This assumption is reasonable within a single domain as assumed in this paper.

A. Convergence Properties

We assume that the following conditions hold:

A1. $C_l(.)$ is convex for all $l \in L$, but is not necessarily differentiable. The subdifferential of C at x [16], denoted by $\partial C(x)$, is bounded for all $x \in \Theta$, where Θ is the feasible set of x.

A2. $\Delta_{s,i}(k)$ are (i) mutually independent with zero mean for all $s \in S$ and $i \in \{1, 2, \cdots, N_s \cdot |D^s|\}$, (ii) uniformly bounded by some constant $\alpha < \infty$, (iii) independent of $(x(l), l = 0, 1, \cdots, k)$, and (iv) $E[(\Delta_{s,i}(k))^{-1}]$, $E[(\Delta_{s,i}(k))^{-2}]$ are bounded $\forall\, k$.

A3. $E[\mu_s^+(k) - \mu_s^-(k)|\Delta(k), \mathcal{F}_k] = 0$ almost surely and $E[\mu_s^{(\pm)^2}(k)]$ are bounded for all k, where $\mathcal{F}_k$ is the σ-field generated by $\{x(0), \cdots, x(k)\}$ [17].

A4. (i) $\sum_{k=1}^{\infty} a_s(k) = \infty$, (ii) $a_s(k) \to 0$ as $k \to \infty$, (iii) $\sum_{k=1}^{\infty} \frac{a_s^2(k)}{c_s^2(k)} < \infty$, (iv) $c_s(k) \to 0$ as $k \to \infty$, and (v) $\lim_{k\to\infty} \left(\frac{c_s(k)}{c_{s'}(k)}\right) = 1$ for all $s, s' \in \mathcal{S}$.

A5. There exists a positive constant M such that

$$\frac{1}{M} \leq \frac{a_s(k)}{a_{s'}(k)} \leq M \tag{11}$$

for all s, $s' \in S$ and for all k.

A6. (i) $\sum_{k=1}^{\infty}(\hat{a}(k) - a_s(k)) < \infty$ for all $s \in \mathcal{S}$, and (ii) $\lim_{k\to\infty} \frac{a_s(k)}{\hat{a}(k)} = 1$, where $\hat{a}(k) = \max_{s\in S} a_s(k)$.

Proposition 4.1: Under Assumptions A1 - A6, the sequence $x(k) = (x_s(k),\, s \in S)$ generated by the algorithm defined by (9) converges to the solution of (5) with probability one under each of the three network models with link loads defined by (2)-(4), regardless of the initial vector $(x_s(0), s \in S)$.

Due to the space constraints, we do not present the proof here. The complete proof can be found in [18].

An important remark we would like to make is that the proposed algorithm does not require any modifications in order to converge under different network models. This allows us to compare different network models using the same optimal routing algorithm and identify the benefits obtained by each additional multicasting capability.

Under Network Model-II, the problem (5) can be simplified based on the following observation. Recall that an overlay node $o \in O^s$ forwards packets to all destinations at the same rate $\max_{d\in D^s} x^s_{o,d}$. It is clear that at the solution to (5), for each $o \in O^s$, $x^{s\star}_{o,d}$ are identical for all $d \in D^s$. Hence, the rate control problem can be reduced to finding the rate allocation $x = (x^s_o, s \in \mathcal{S}, o \in O^s)$ under the assumption that all destinations receive the same rate from an overlay node. We state this simple fact as follows:

Corollary 4.2: Let $x^\star$ be the solution to (5) under Network Model-II with link loads defined by (3). Then,

$$x^{s\star}_{o,d} = x^{s\star}_o \qquad \forall d \in D^s,\, o \in O^s,\, s \in S.$$

This observation allows us to reformulate the optimization problem (5) as the following simpler problem:

$$\begin{aligned} \min_x C(x) &= \min_x \sum_l C_l(x^l) \\ \text{s. t.} \sum_{o\in O_s} x^s_o &= r^s,\ \forall\, s \in S \end{aligned} \tag{12}$$

where, with a little abuse of notation, $x = (x^s_o, s \in \mathcal{S}, o \in O^s)$. Basically, the problem can be reduced to one of finding optimal overlay rates x^s_o. When the number of receivers is large, this could lead to much lower computational requirement.

Note that in (12) the term ϵ_s is removed. This is due to the fact that, at a feasible solution, the source node delivers packets to the overlay nodes, and each overlay node forwards every packet to all destinations. As a result, under this network model, source coding is not required

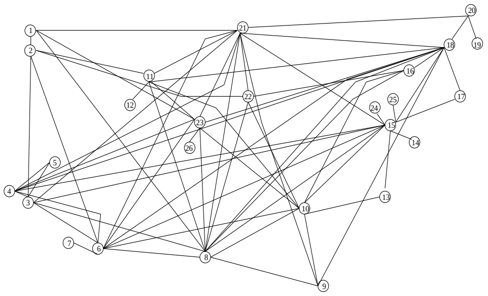

Fig. 2. Network Topology

to handle the issue of bookkeeping, and ϵ_s can be set to zero. We refer to this formulation as NM-IIb in Section V.

V. Simulation Results

The purpose of this section is to identify the characteristics of the proposed routing algorithm and evaluate its performance under various network conditions. We will use DVMRP as a benchmark while presenting the results.

We wrote a packet level discrete-event simulator. Each plot presented below illustrates the average of 10 independent runs that are initiated with different random seeds. For the optimization algorithm, the link cost function is selected as $\left(x^l/c^l\right)^2$, where c^l is the link capacity and x^l is the link rate as defined before. In all simulations, the period of link state measurements is selected as one second. As a consequence, source nodes can update their rates at best approximately every two seconds since we require two measurements for estimating the gradient vector according to the SPSA. For simplicity we set ϵ_s, the rate of redundancy due to source coding, to zero.

Experiments are conducted with the intradomain network topology given in Fig. 2[4]. It is a close approximation of Sprint's backbone topology as reported in [19]. It is of interest to analyze how our routing algorithm performs under these conditions since, as mentioned in Section I, recent findings suggest that many ISPs are in the process of increasing the node connectivity of their networks. Each link has a bandwidth of 20 Mbps. We have 3 sources that simultaneously send multicast traffic, where each source has 18 receivers and nodes 10 and 23 are selected as additional overlay nodes. Specifically, $\mathcal{S} = \{1, 9, 22\}$ and $O^1 = \{1, 10, 23\}$, $O^9 = \{9, 10, 23\}$ and $O^{22} = \{22, 10, 23\}$. Each source-destination pair has three paths including the min-hop path starting at the source node and each source generates Poisson traffic with an average rate of 10 Mbps.[5] The routing algorithm starts from the setting that all overlay rates other than the source nodes are set to zero (i.e., $x^s_{o,d} = 0$ if $o \neq s$, $x^s_{s,d} = r_s$). Hence, in NM-I model, the algorithm starts with basic unicast routing to reach each destination, while in NM-IIa, NM-IIb and NM-III models it starts with a single shortest path multicast tree (e.g., DVMRP tree) rooted at each source node and gradually shifts traffic to alternative trees rooted at overlay nodes 10 and 23. Fig. 3 illustrates the variation of total network cost and loss rate for different models. We have also computed the optimal cost values of network models NM-II and NM-III using MATLAB. The optimal values for NM-II and NM-III turn out to be close (12.5875 versus 11.7612) suggesting to us that the complexity of having smart routers that are able to forward packets onto each branch at a different rate offers only a marginal benefit in this scenario. However, it is hard to draw any further conclusions as this result may depend on the specific topology and source-destination pair selections. Also, our algorithm does better than DVMRP under NM-IIa, NM-IIb and NM-III models as a consequence of the availability of multiple trees to distribute the traffic load. However, while under NM-I model the algorithm is able to minimize the cost to a certain level, it cannot eliminate the packet losses and has a much higher overall cost compared to DVMRP.[6] The reason behind this result is the lack of multicast functionality. Since we cannot create multicast trees, the only savings due to multicasting occurs between the sources and overlay nodes. Once multicast packets reach the overlays, overlay nodes need to create independent unicast sessions for each destination ignoring the multicast nature of the traffic, and this creates a high level of link stress as multiple copies of the same packets are generated. One important observation is that the algorithm is able to converge faster in network model NM-IIb than all other models. This is due to the fact that, as a consequence of Corollary 4.2, we only need to optimize the overlay rates x^s_o instead of individual receiver rates $x^s_{o,d}$. Hence, the number of parameters to be calculated is much smaller than the other two cases (9 versus 162).

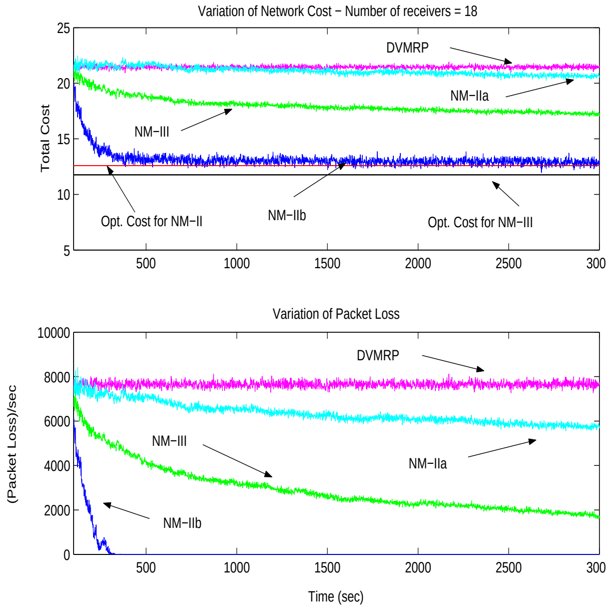

Fig. 3. Variation of total cost and packet loss - Poisson source

[4] We present a limited set of simulation results due to page limits. A detailed simulation study under different network topologies and source models can be found in [18].

[5] Since we focus on intradomain, this rate may represent the overall rate of multiple multicast sources having same receiver set.

[6] For better viewing purposes we did not put the results of NM-I in the plots. Please refer to [18] for plots with NM-I model.

References

[1] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, “Scalable application layer multicast,” in *ACM SIGCOMM*, 2002.

[2] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, “Overview and principles of internet traffic engineering,” RFC 3272, 2002.

[3] K. Park and Y. Shin, “Uncapacitated point-to-multipoint network flow problem and its application to multicasting in telecommunication networks,” *European Journal of Operational Research*, vol. 147, pp. 405–417, 2003.

[4] A. Elwalid, C. Jin, S. Low, and I. Widjaja, “MATE: MPLS adaptive traffic engineering,” in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Anchorage, Alaska, Apr. 2001.

[5] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, “Network information flow,” *IEEE Transactions on Information Theory*, vol. IT-46, pp. 1204–1216, 2000.

[6] R. Koetter and M. Medard, “Beyond Routing: an algebraic approach to network coding,” in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, 2002.

[7] S.-Y. R. Li and R. W. Yeung, “Linear network coding,” *IEEE Transactions on Information Theory*, vol. 49, pp. 371–381, 2003.

[8] T. Noguchi, T. Matsuda, and M. Yamamoto, “Performance evaluation of new multicast architecture with network coding,” *IEICE Trans. Commun*, vol. E86-B, pp. 1788–1795, 2003.

[9] Y. Zhu, B. Li, and J. Guo, “Multicast with network coding in application-layer overlay networks,” *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 107–120, 2004.

[10] T. Güven, C. Kommareddy, R. J. La, M. A. Shayman, and B. Bhattacharjee, “Measurement based optimal multi-path routing,” in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Hong Kong, Mar. 2004.

[11] D. J. C. Mackay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[12] A. Shokrollahi, “Raptor codes,” preprint 2003. [Online]. Available: www.inference.phy.cam.ac.uk/mackay/DFountain.html.

[13] D. Waitzman, C. Partridge, and S. Steering, “Distance vector multicast routing protocol,” RFC 1075, 1998.

[14] J. Kiefer and J. Wolfowitz, “Stochastic estimation of a regression function,” *Ann. Math. Stat.*, vol. 23, pp. 462–466, 1952.

[15] J. R. Blum, “Multidimensional stochastic approximation methods,” *Ann. Math. Stat.*, vol. 25, pp. 737–744, 1954.

[16] Y. He, M. C. Fu, and S. I. Marcus, “Convergence of simultaneous perturbation stochastic approximation for nondifferentiable optimization,” *IEEE Transactions on Automatic Control*, vol. 48, pp. 1459–1463, 2003.

[17] J. C. Spall, “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation,” *IEEE Trans. Automat. Contr.*, vol. 37, pp. 332–341, 1992.

[18] T. Güven, R. J. La, M. A. Shayman, and B. Bhattacharjee. Measurement-based multicast on overlay architecture. Tech. Rep. UMIACS-TR# 2004-45. [Online]. Available: http://www.cs.umd.edu/Library/TRs/CS-TR-4603/CS-TR-4603.pdf

[19] N. Spring, R. Mahajan, and D. Wetherall, “Measuring ISP Topologies with Rocketfuel,” in *ACM SIGCOMM*, 2002.