Measurement-Based Multicast on Overlay Architecture

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

July 6,2004

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 does not assume that the gradient of an analytical cost function is known to the algorithm, but rather relies on noisy estimates from measurements. Using the analytical model presented in the paper we prove the convergence of the algorithm to the corresponding optimal solution under each network model. Simulation results are presented to demonstrate the additional benefits obtained by incrementally increasing the multicasting capabilities. We also provide a comparative study with the well-known IP multicast algorithm DVMRP.

1 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]. Many of major Internet Service Providers (ISPs) are in the process of increasing their network capacity, including providing higher node connectivity. A higher level of network connectivity typically provides multiple paths between source-destination pairs, and offers an opportunity to better utilize network resources. Our focus in this paper is to scrutinize the effects of load balancing the multicast traffic in an intradomain.

There is a limited amount of existing work on multipath multicast routing. In [3], the authors propose a solution by creating multiple trees between a source and a set of destinations and splitting the traffic optimally along the trees. However, the solution covers only the single source case. In addition, it assumes the existence of the gradient of an analytical cost function. As discussed in [4], it may not be feasible to precisely define accurate analytical cost functions due to the dynamic nature of networks. Moreover, the cost function is assumed to be continuously differentiable and strictly convex. As we will discuss later, these assumptions can be relaxed in our case. 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, these solutions suffer from the limitations inherited from network coding. First of all, network coding relies on the unrealistic assumption that the network is lossless as long as the average link rates do not exceed the link capacities. Besides, a packet loss is actually much more costly when network coding is employed since it potentially affects the decoding for 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 considerable complexity and requires high level of coordination and synchronism among the nodes. Furthermore, similar to earlier efforts, these solutions also assume that there is only one multicast source in the network.

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 existence of the gradient of an analytical cost function and is inspired from the unicast routing algorithm based on Simultaneous Perturbation Stochastic Approximation (SPSA) [10]. To the best of our knowledge, this algorithm is the first attempt to solve the optimal multipath routing problem of multiple multicast flows in a distributed fashion while relying only on network measurements.

In addition, we address the optimal multipath multicast routing problem in a more general framework than having multiple distribution 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 first look at the problem under the traditional network model without any IP multicasting functionality where multiple paths are established using a limited number of (application-layer) overlay nodes. Next, we evaluate the performance in a network model with multiple distribution trees. Finally, we relax the usual assumption that from a given multicast tree each receiver gets multicast packets at the same rate. Intuitively, relaxing this assumption may seem to create more problems than bringing any benefits. This is due to the fact that it potentially creates a complex bookkeeping problem since source node has to make sure each receiver gets a distinct set of packets from different trees while satisfying the rate constraints along each tree. However, using a specific source coding called Digital Fountain codes [11], we overcome this problem in an efficient way, which gives an opportunity to observe the potential benefits of having different receiver rates on a multicast distribution tree. In fact, our results show that the network actually benefits from such a setting by being able to balance the traffic load efficiently and achieving a better performance in terms of end-to-end throughput.

The rest of the paper is organized as follows. In Section II we give a brief overview of Digital Fountain Codes and discuss how it can help the problem of multipath multicast routing. Section III presents the network models we consider and introduces the general routing framework. We present the optimization framework and prove the optimality and stability of the routing algorithm in Section IV. Section V discusses the implementation issues. In Section VI, we describe the experimental setup used to evaluate the performance of the proposed algorithm under different network models and present the simulation results. We conclude the paper in Section VII.

2 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. Since the Internet can be modelled as an erasure channel, the idea is to use an erasure-correcting code that will eliminate retransmissions. The classic block codes for erasure correction are called Reed-Solomon codes. An $(N;K)$ Reed-Solomon code has the ideal property that if any K of the N transmitted symbols are received then the original K source symbols can be recovered. However, when using a Reed-Solomon code, as with any block code, one must estimate the erasure probability and choose the code rate $R = K/N$ before transmission. Besides, Reed-Solomon codes have the disadvantage that they are practical only for small K, N.

Fountain codes overcome all these problems. 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 on the channel, one can send as many encoded packets as needed in order for the decoder to recover the source data. The input and output symbols can be bits or more generally binary vectors of arbitrary length. Each output symbol is generated by the addition of some randomly selected input symbols where the number of input symbols to be added is decided according to a given degree distribution. It is assumed that each output symbol is equipped with information describing which input symbols it is the addition of (e.g., using a header in a packet).

A decoding algorithm for a Fountain Code is an algorithm which can recover the original K input symbols from any set of M output symbols with a high probability. For good Fountain Codes the value of 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 probability of decoding failure converges to zero polynomially fast in the number of input symbols. For instance, for $K = 64,536$ and $M = 65,552$, i.e., with a redundancy of 1.5%, it is shown in [12] that the error probability is upper bounded by 1.71×10^{-14}. So, one can easily encode a traffic source by simply dividing it into blocks whose size will only be constrained by the buffer size located at the source and apply the Raptor coding to each block.

Raptor codes have been used in commercial systems by Digital Fountain startup company. Their Raptor code implementation can encode packets at speeds of several gigabits per second on a 2.4 Ghz Intel Xeon processor with stringent error probability of decoding conditions even for small number of input symbols.

The Fountain codes are very useful in the multipath multicast routing. Since a receiver can recover the K source symbols from any M coded symbols, the source node does not need to do any bookkeeping as long as it sends distinct packets along each path. This will guarantee that each receiver successfully receives the whole multicast stream as long as each user receives packets at a sufficient rate. This gives us much flexibility to send multicast traffic to each destination at different rates along a given shared path (e.g., distribution tree) without requiring consideration of the exact packets to be sent to each destination.

3 Model

Consider a network that consists of a set of unidirectional links $\mathcal{L} = \{1, \ldots, \text{L}\}$ and a set of nodes $\mathcal{N} = \{1, \ldots, \text{N}\}$. 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 session[1] $s \in \mathcal{S}$ let D^s denote 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. These assumptions capture the performance and cost trade-off. We study the relative performance of these systems and some of the properties of their operating points.

3.1 General Routing Framework - Overlay Architecture

Before describing the network models to be used in our study, we first describe how an overlay architecture is used 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. These are called *overlay nodes*, and the set of overlay nodes is given by $\mathcal{O}$.

[1] For simplicity each source is assumed to have a single multicast session. However, the results also apply to the case where there may be multiple sessions with the same source node.

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 that are 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 at which the source node s sends packets to d through an overlay node $o \in O^s$. Also, define x^s_o to be the total rate at which an overlay node o receives packets from source s. In a unicast case this is simply the amount of rate being forwarded to the destination through the overlay node, while in the case of a multicast session, it depends on the capability of the underlying network and the implementation.

As discussed in Section 1, 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

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

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 next issue is how to forward packets from overlay nodes to destinations. The answer to this problem 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. 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

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

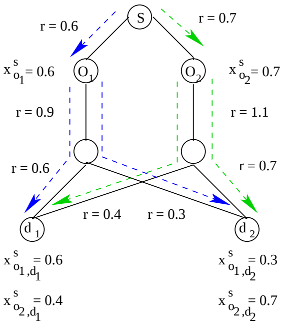

Figure 1: IP multicast not available.

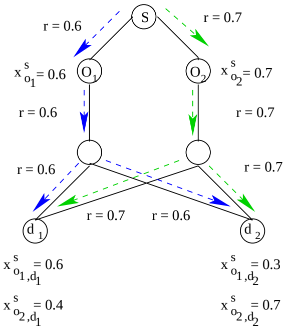

Figure 2: IP multicast available.

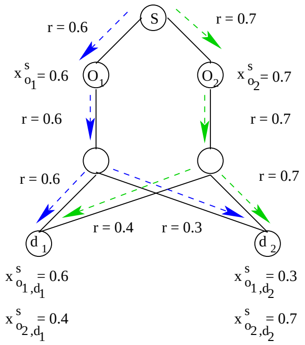

Figure 3: Smart routers available.

packets to different branches at different rates. As a result, all destination nodes will receive packets at the rate $x^s_o (= \max_{d \in D^s} x^s_{o,d})$ if there are no packet losses. Fig. 2 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 to the overlay nodes to minimize 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 shown in Fig. 3. 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.

3.2 Link Loads

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

3.2.1 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 encoded 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

$$x^l = \sum_{s \in S} \left(\sum_{o \in O^s : l \in V_o^s} x_o^s + \sum_{o \in O^s} \Big(\sum_{d \in D^s : l \in V_d^o} x_{o,d}^s \Big) \right) \tag{2}$$

This model is referred to as NM-I in Section 6.

3.2.2 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 $x_o^s = \max_{d \in D^s} x_{o,d}^s$.

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

$$x^l = \sum_{s \in \mathcal{S}} \left(\sum_{o \in O^s : l \in V_o^s} x_o^s + \sum_{o \in O^s : l \in T_o^s} x_o^s \right) \tag{3}$$

where T_o^s 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 6.

3.2.3 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. 3. Under this model a source s can select the individual rates $x_{o,d}^s$ indepedently for each destination, and each destination $d \in D^s$ will receive the intended rate $x_{o,d}^s$[3] instead of $\max_{d' \in D^s} x_{o,d'}^s$ 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.

The link rates can be written as

$$x^l = \sum_{s \in S} \left(\sum_{o \in O^s : l \in V_o^s} x_o^s + \sum_{o \in O^s} \max_{d \in D^s : l \in \hat{V}_d^o} x_{o,d}^s \right) \tag{4}$$

Here $\hat{V}_d^o$ 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 1. We assume that a suitable coding scheme is adopted. We will refer to this model as NM-III while presenting the experiments.

Due to the large gap between the level of intelligence currently available at the IP routers and the required intelligence assumed in this model, it is unlikely that this type

[3] This assumes that there are no packet losses along the path.

of network will be available in the near future. However, we consider this model for comparison purposes, and compare the performance of the other models against that of this model.

4 Optimization Framework and Stochastic Approximation

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 may not be differentiable. The optimization problem can be stated as follows:

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

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

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

where ν is an arbitrarily small positive constant[4] and ϵ^s is the required additional rate of the coding scheme for a receiver to successfully decode the incoming encoded data.

Our problem in (5) can be viewed as a natural extension of [10]. One main difference between the current problem and [10] is that the link loads $x^l, l \in \mathcal{L}$, used in the objective function depend on the adopted network model, and hence the performance of the system depends on the capability of the underlying network.

The optimization problem in (5) can be solved using a Stochastic Approximation (SA) technique. SA is a recursive procedure for finding the root(s) of equations using noisy measurements, and is useful for finding extrema of functions [14] (*e.g.,* [15, 16]). 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 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

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

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

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

Stochastic 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 other words, since each source is responsible for satisfying constraints (6) - (7) for each destination independently of others, the problem can be decomposed into several coupled distributed problems 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 denotes 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. In other words, 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:[5]

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

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 $\{\mathbf{c}_s(k), s \in \mathcal{S}\}$ s.t. $\mathbf{c}_s(k) = c_s(k) \cdot I_s$ 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. 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.

4.1 Convergence Properties

In this subsection we establish the convergence of (9) under the three network models described in subsection 3.2. In order to establish the convergence of our algorithm based on SPSA, we assume that the following conditions hold:

[5] If $C(x(k))$ is not differentiable, $\hat{g}_s(k)$ becomes an approximation to the subgradient vector $sg(x)$.

A1. $C_l\left(x^l(k)\right)$ is convex for all $l \in L$, but is not necessarily differentiable. The subdifferential of C at x [17], 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 $\{x(0), \cdots, x(k)\}$.

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) under each of three network models with link loads defined by (2)-(4) with probability one, regardless of the initial vector $(x_s(0), s \in S)$.*

We only provide the proof with the link load given by (2), which is given in the appendix. The proof for the other two models follows from the fact that the necessary convexity of the objective function can be established in a similar manner. Note here 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 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 $\max_{d\in D^s} x_{o,d}^s$. It is clear that at the solution to (5), for each $o \in O^s$, $x_{o,d}^{s\star}$ are identical for all $d \in D^s$. Hence, the rate control problem can be reduced to finding the rate allocation $x = (x_o^s, s \in \mathcal{S}, o \in O^s)$ under the assumption that all destinations receive the same rate from an overlay node.

We state this simple fact as the following corollary:

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

$$x_{o,d}^{s\star} = x_o^{s\star} \qquad \forall d \in D^s,\, o \in O^s,\, s \in S.$$

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

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

where, with a little abuse of notation, $x = (x_o^s, s \in \mathcal{S}, o \in O^s)$. Basically, the problem can be reduced to one of finding optimal overlay rates x_o^s. 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 to handle the issue of bookkeeping, and ϵ_s can be set to zero.

We refer to this formulation as NM-IIb in Section 6.

5 Implementation Issues

In this section, we will discuss the practical implementation requirements to provide the traffic engineering capabilities we have discussed.

5.1 Traffic Monitoring

The proposed algorithm relies on periodic measurements. More specifically, the model in the previous section assumes that the overall network cost is available to all sources in each period. This requires that either (i) the cost of every link is communicated to a centralized location that forwards the overall network cost to the sources, or (ii) the link costs are directly communicated to the sources. Both of these approaches are likely to incur a high communication overhead. Instead, we assume that each source obtains the costs of the links that lie in a path to at least one of its destinations. As we will see in Section 6, this appears to have only marginal effects on the performance. The required traffic monitoring process can be handled by an overlay architecture. Each link in the network is mapped to the closest overlay node with a certain tie-breaking rule that gives a unique mapping. Overlay nodes periodically poll the links for which they are responsible, process the data and forward necessary local state information to the source nodes utilizing the corresponding links in a coordinated way. (Note that this way the links are not required to be probed by each source. See [18] for details.) While sending the information to a source node, the overlay nodes also aggregate the information gathered from different links as much as possible. For instance, the cost information obtained from the links that are on a particular path of a source is aggregated by the overlay nodes, using the fact that the cost structure is additive according to the definition given in (5).

As a consequence, the overhead caused by the distribution of the link state information is minimized.

5.2 Traffic Filtering

In a scenario where multicast source does not employ Digital Fountain coding,(e.g. NM-IIb algorithm) one has to give special care while splitting the traffic at the source node to avoid the well-known reordering problem especially for the TCP traffic. The algorithm proposed in this paper calculates the rates at which traffic should be distributed along the alternative paths without requiring or specifying the exact paths that a particular packet should follow. Therefore, any existing filtering scheme that minimizes the reordering problem can be used for this purpose. A possible solution depending on hash-functions is presented in [4].

6 Experimental Setup and 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. Using simulations, we would like to verify that the algorithm is stable and robust in each case in such a way that minimizes congestion and quickly balances the load among the multiple paths of multicast source-destination pairs in a reasonable period of time. In addition, we would like to compare the performance of the algorithm under each model to identify the benefits observed as the level of intelligence increases at the routers. 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 $(x^l/c^l)^2$, 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 two different intradomain network topologies. The first topology is given in Figure 4. This topology is also used in [19], [20] and considered to be typical of a large ISP's network. (This topology closely resembles the MCI backbone topology [21].) Each link has a bandwidth of 20 Mbps. Packet size is selected to be 500 bytes. Nodes 1 and 5 are selected as multicast sources. In the first set of simulations, each source has 6 receivers. In particular, $D^1 = \{4, 7, 8, 11, 13, 16\}$ while $D^5 = \{1, 8, 9, 12, 15, 17\}$. There are three alternative paths to reach each destination via two additional overlay nodes 9 and 17; i.e. $O^1 = \{1, 9, 17\}$ and $O^5 = \{5, 9, 17\}$. Each source generates Poisson traffic with an average rate of 11.5 Mbps.[6] 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

[6] Since we focus on intradomain routing, this rate may represent an aggregate rate of multiple multicast sources having the same receiver set D^s.

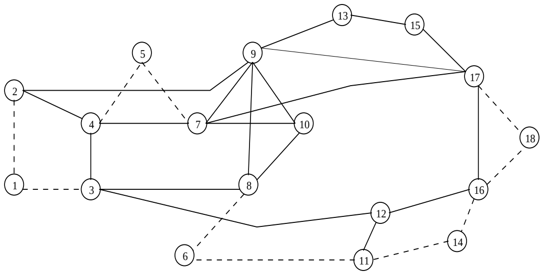

Figure 4: Network Topology 1

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 shift traffic to alternative trees that are rooted at overlay nodes 9 and 17.

Figures 5 and 6 illustrate 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 a MATLAB optimization package. The optimal value of NM-II is 4.7979 while that of NM-III is 4.3548. As we see, the optimal value of NM-III is smaller than that of all other models, which is expected as it has the highest level of multicast functionality/intelligence. Also, our algorithm do better than DVMRP under NM-IIa and NM-IIb 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. (The cost of NM-I model decreases to around 14.197 while it is around 7.45 for DVMRP.) The reason behind this result is the lack of multicast functionality. Since we cannot create multicast trees, the only savings due to multicasting occur 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 in the network as multiple copies of the same packets are generated.

One important observation is that the optimal cost values of NM-II and NM-III models are close. Hence, the additional 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.[7] In addition, 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_o^s instead of individual receiver rates $x_{o,d}^s$. Hence, the number of parameters to be calculated is much smaller than the other two cases (6 versus 36). This is clearly seen in Figures 7 and 8, which present the variation of cost and packet loss rates when the number of receivers is increased to 11 for both sources. Specifically, this time $D^1 = \{3, 4, 7, 8, 9, 10, 11, 12, 13, 16, 17\}$ and $D^5 = \{1, 2, 3, 4, 8, 9, 10, 12, 15, 16, 17\}$. We see that as the number of receivers increases, the convergence of NM-IIa and NM-III becomes considerably slower. On the contrary, the algorithm does not suffer from a slower convergence rate under the NM-IIb model

[7] It is still hard to draw any conclusions as this result may depend on the specific topology and source-destination pair selections.

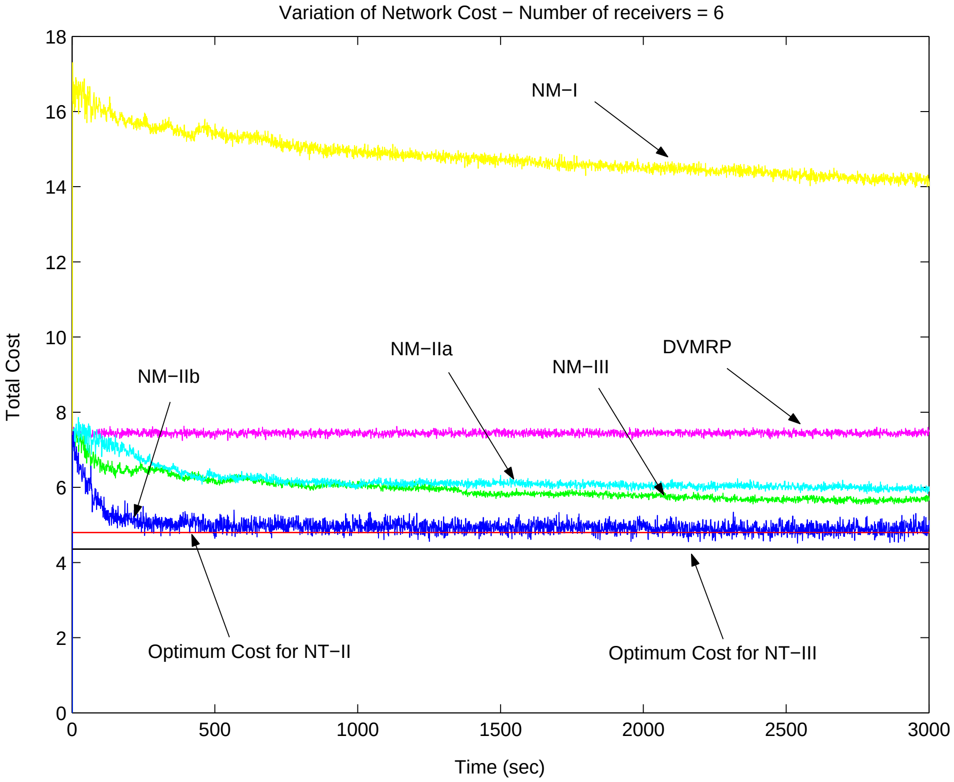

Figure 5: Variation of total cost - 6 receivers

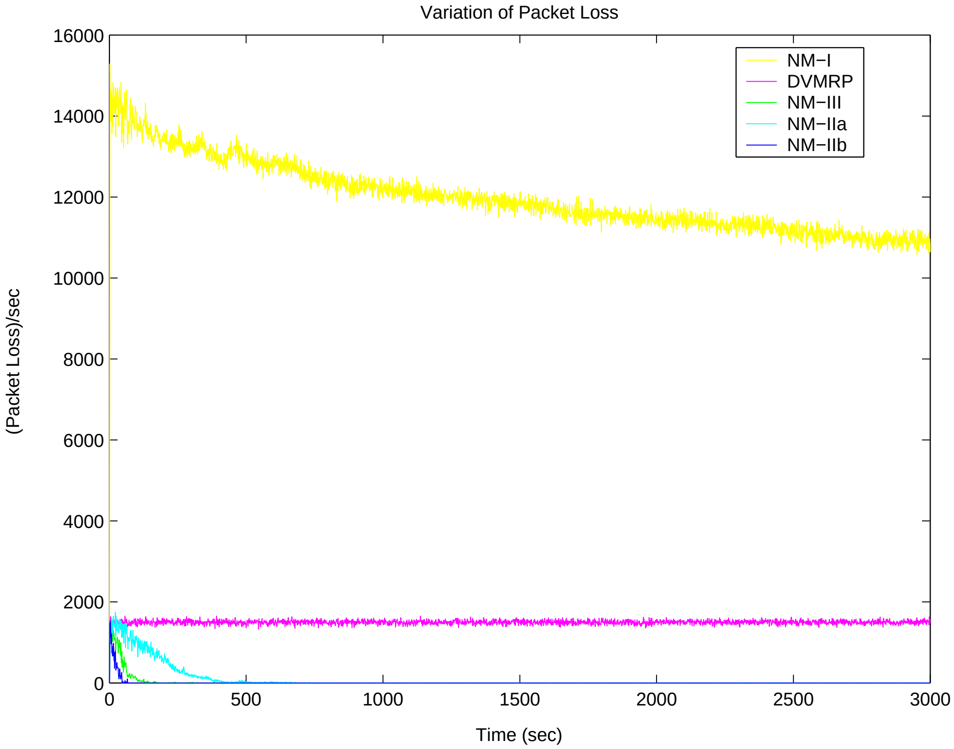

Figure 6: Variation of packet loss - 6 receivers

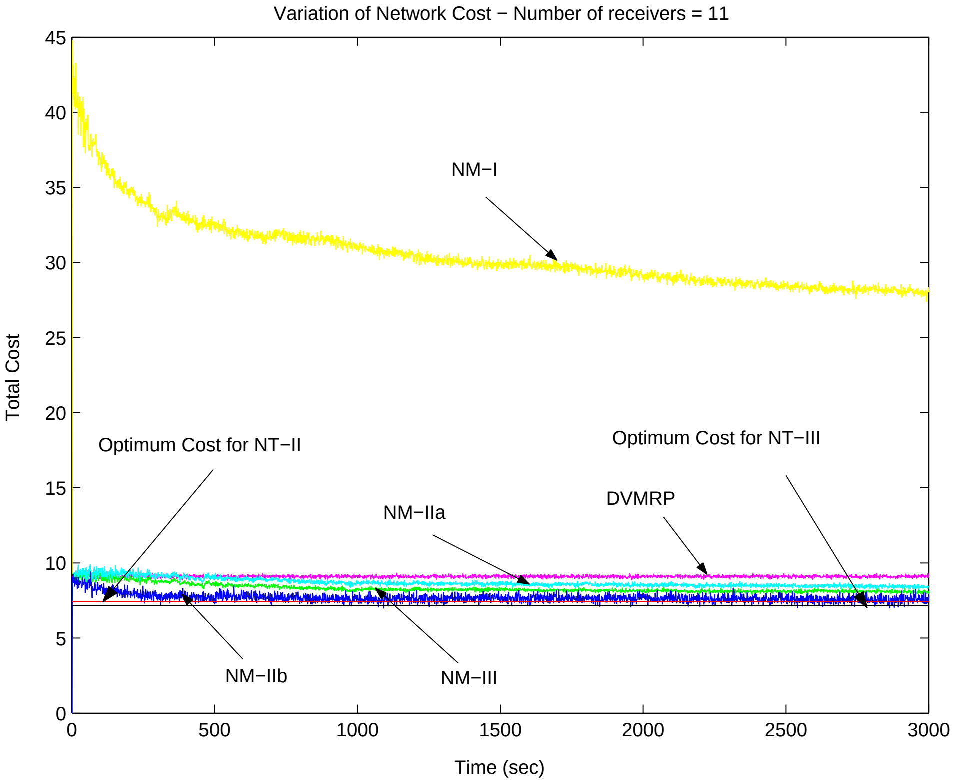

Figure 7: Variation of total cost - 11 receivers

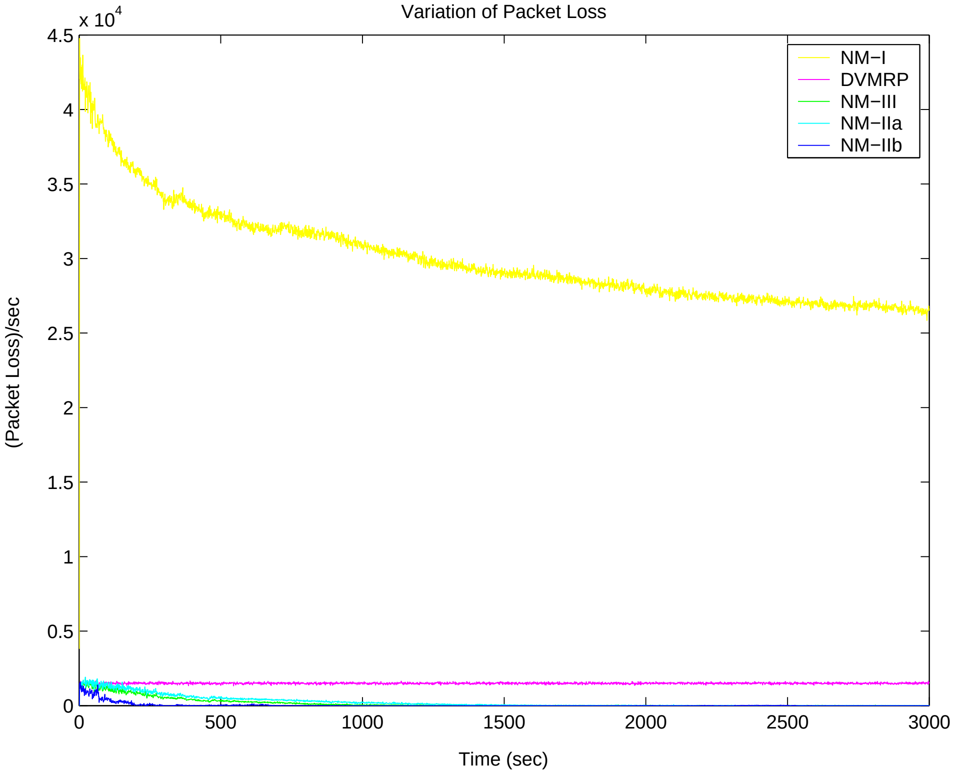

Figure 8: Variation of packet loss - 11 receivers

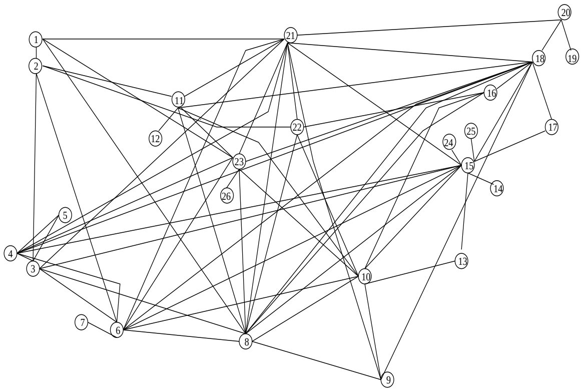

Figure 9: Network Topology 2

with increasing number of receivers. However, we would like to note that in all three models the algorithm is able to minimize packet losses within a reasonable amount of time, leading to better network performance. Moreover, we see that three paths are sufficient to effectively distribute the traffic, which suggests that a limited number of overlay nodes are able to substantially improve the network performance.

A final remark is that in NM-III setup, the optimal $x^s_{o,d}$ values turn out to be unequal for different destinations for a given overlay node as discussed earlier. For instance, under this setup, simulation results show that $x^1_{9,16}$ is roughly $0.026 \cdot r_s$, while $x^1_{9,10}$ is $0.122 \cdot r_s$.

Figure 9 represents the second topology we consider. It is a close approximation of Sprint's backbone topology as reported in [22]. It is of interest to analyze how our routing algorithm performs under these conditions since, as mentioned in Section 1, recent findings suggest that many ISPs are in the process of increasing the node connectivity of their networks. (As a comparison, the average node degree of Sprint's topology is 5.0769 while it was 3.1667 in the first topology.)

Again all links have a bandwidth of 20 Mbps. This time, we have 3 sources that simultaneously send multicast traffic, and each source has 18 receivers. Specifically, $\mathcal{S} = \{1, 9, 22\}$ and $D^1 = \{$2, 3, 4, 5, 6, 8, 9, 10, 11, 13, 15, 16, 19, 20, 21, 22, 23, 25$\}$, D^9 = $\{$1, 2, 3, 4, 6, 7, 8, 10, 11, 13, 15, 16, 17, 18, 21, 22, 23, 24$\}$ and $D^{17} = \{$1, 2, 3, 4, 6, 8, 9, 10, 11, 12, 14, 15, 16, 17, 20, 21, 23, 26$\}$. Nodes 10 and 23 are selected as additional overlay nodes, i.e. $O^1 = \{1, 10, 23\}$, $O^9 = \{9, 10, 23\}$ and $O^{22} = \{22, 10, 23\}$. Similar to previous simulation setup, each source-destination pair has three paths including the min-hop path starting at the source node. Each source generates Poisson traffic with an average rate of 10 Mbps.

Figures 10 and 11 present the results. Once again NM-I model suffers from high link stress and performs worse than DVMRP. On the other hand, since the number of receivers is relatively large, the convergence of the algorithm under NM-III and NM-IIa models are again slower than that under NM-IIb model. The optimal cost values of NM-IIa and NM-IIb models are again close (11.7612 vs. 12.5875), demonstrating that much of the benefits can be observed with the simpler network design NM-IIb. Finally, we again see that three paths per receiver is sufficient to successfully distribute the traffic.

In the last set of experiments, we have used another source model to represent VBR

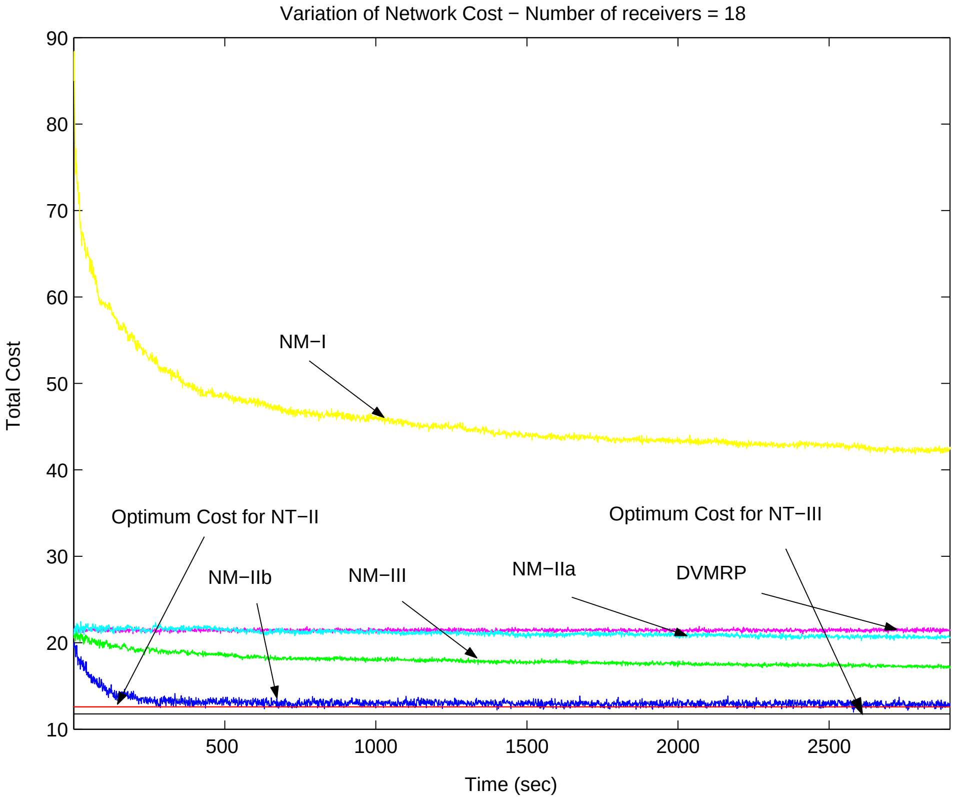

Figure 10: Variation of total cost - Poisson source

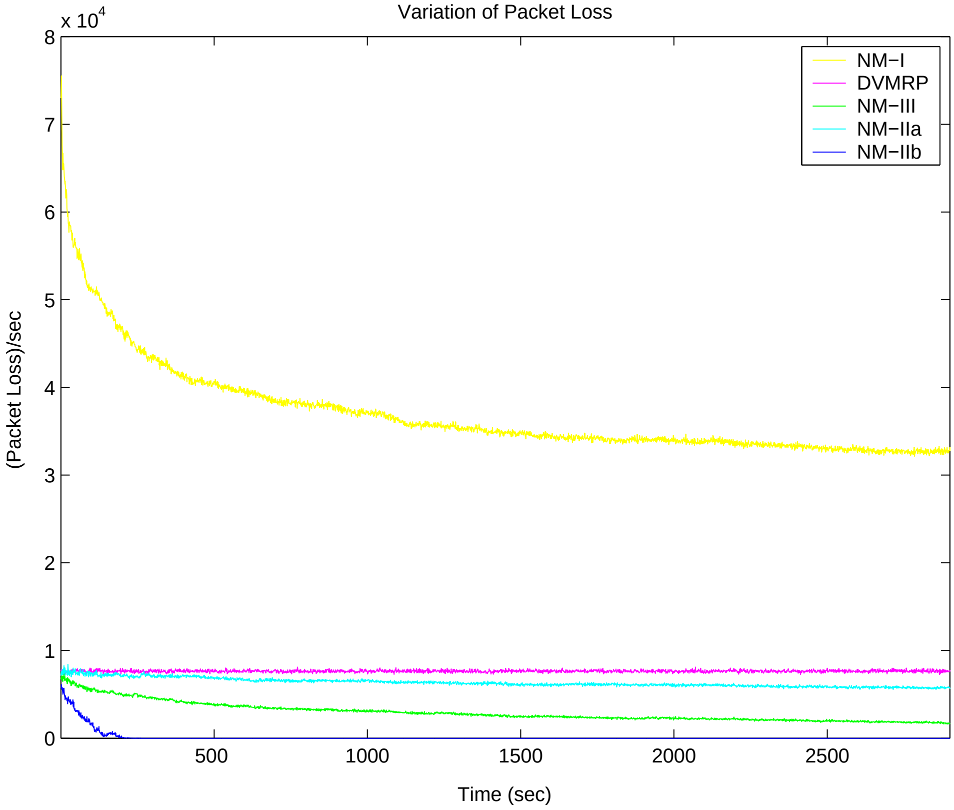

Figure 11: Variation of packet loss - Poisson source

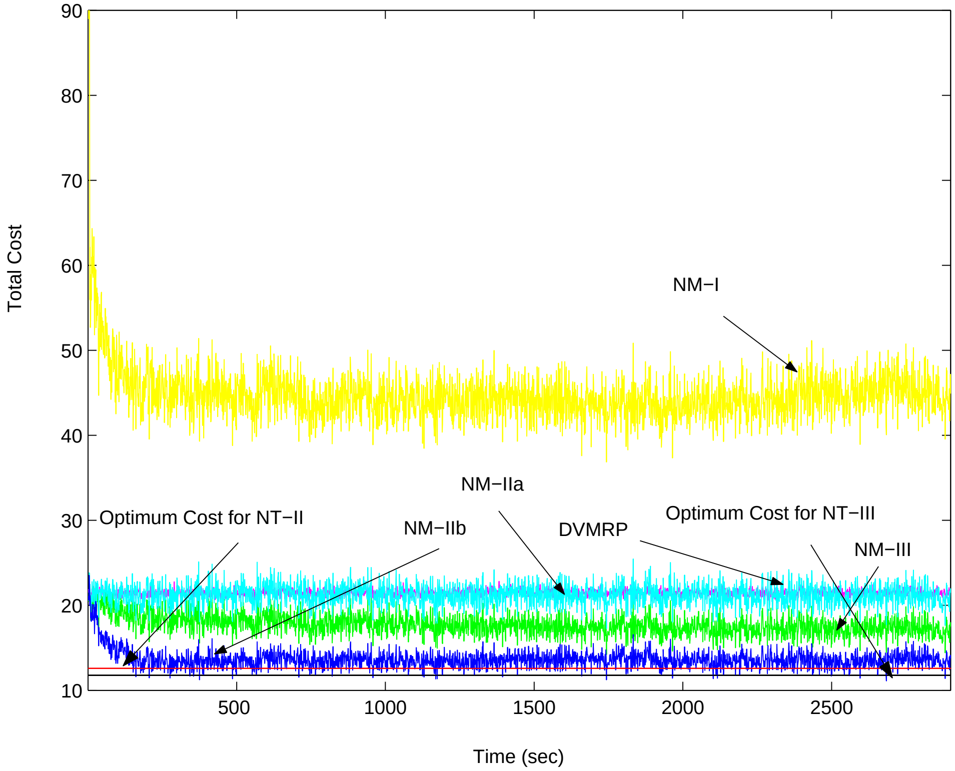

Figure 12: Variation of total cost - MMPP source

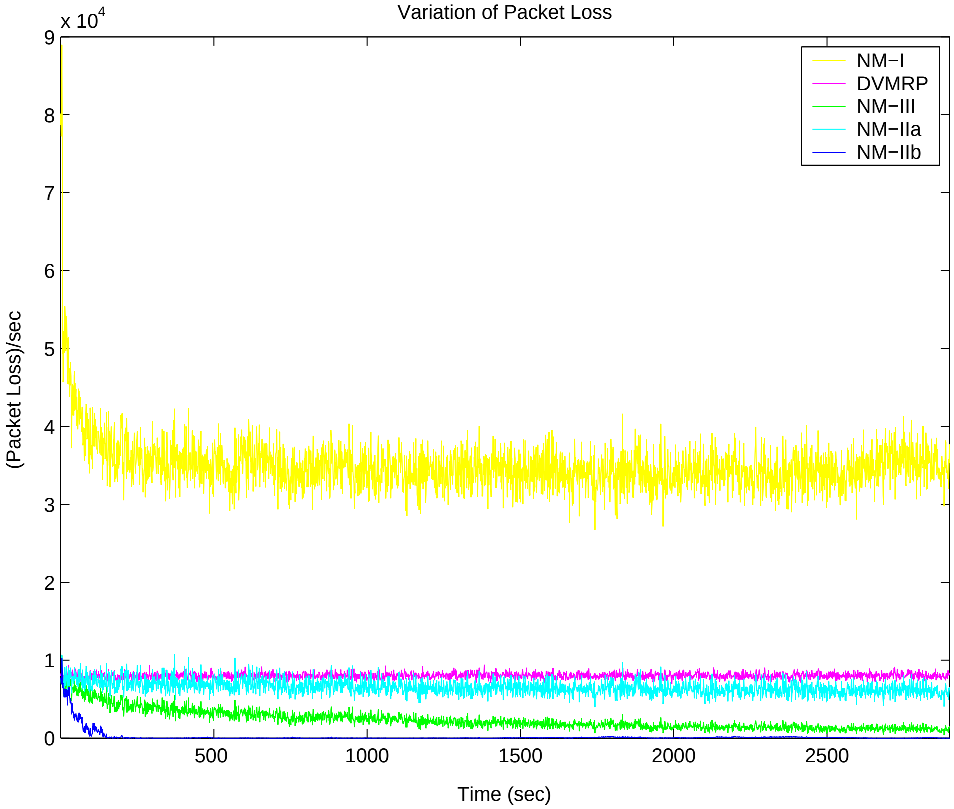

Figure 13: Variation of packet loss - MMPP source

video traffic.[8] It is shown in [23] that the long-range dependence is not a crucial property in determining the buffer behavior of VBR video traffic and a Markov chain traffic model can be used to estimate the buffer occupancy well. Following this observation, we have selected the MMPP (Markov Modulated Poisson Process) as an alternative source model. We assumed that each 10 Mbps MMPP source consists of 128 ON-OFF mini-sources where mean ON period is 350 ms and mean OFF period is 650 ms. From Figures 12 and 13, we see that the performance of the algorithm in each network model is similar to the Poisson source case except the higher variance observed around the mean values. This is expected since the packet rates at the sources fluctuate around some mean value from the adopted traffic model. Also, note that the convergence rate of the system decreases with increasing noise level.

7 Conclusion

In this paper, we have focused on the optimal multipath routing of multicast traffic where the link cost derivatives cannot be calculated analytically but can only be estimated through measurements. We have applied the SPSA idea to overcome this problem. Using Digital Fountain coding we have established a routing framework that generalizes the multiple distribution trees to a more general multiple path scenario where each destination can receive packets at a different rate from a multicast tree. In addition, we have studied the performance of optimal multipath multicast routing under different network models to estimate the level of multicasting functionality required inside the network to effectively load balance the traffic. Simulation results show that, while IP multicasting functionality is highly essential for better performance, additional functionalities, such as probabilistic splitting of multicast traffic along a multicast tree, provide only limited benefits in relation to their required complexity.

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.

[8] Due to source encoding, it is possible to shape the incoming multicast traffic at the source nodes. However, this issue is ignored in the simulation studies.

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

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

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

[17] 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.

[18] T. Güven, C. Kommareddy, R. J. La, M. A. Shayman, and B. Bhattacharjee. Measurement based optimal multi-path routing. Tech. Rep. UMIACS-TR# 2003-69. [Online]. Available: http://www.cs.umd.edu/Library/TRs/CS-TR-4500/CS-TR-4500.pdf

[19] S. Nelakuditi and Z. L. Zhang, “A localized adaptive proportioning approach to QoS routing,” *IEEE Commun. Mag.*, vol. 40, no. 6, pp. 66–71, 2002.

[20] G. Apostolopoulos, R. Guerin, S. Kamat, and S. Tripathi, “Quality of service based routing: A performance perspective,” in *ACM SIGCOMM*, 1998.

[21] Q. Ma and P. Steenkiste, “On path selection for traffic with bandwidth guarantees,” in *IEEE International Conference on Network Protocols*, 1997.

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

[23] D. P. Heyman and T. V. Lakshman, “What are the implications of long range dependence for VBR video traffic engineering,” *IEEE/ACM Transactions on Networking*, vol. 4(3), pp. 301–317, 1996.

[24] G.R.Grimmett and D. Stirzaker, *Probability and Random Processes.* Oxford Science Publications, 2nd edition, 1998.

[25] H. Kushner and D. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems.* Springer-Verlag, 1978.

[26] H. Kushner and G. Yin, *Stochastic Approximation Algorithms and Applications.* Springer-Verlag, 1997.

Appendix - I
Proof of Proposition 4.1

First, note that the algorithm defined in (9) has the same form as the unicast algorithm defined in [18]. In the unicast case, it was assumed that the link cost functions and consequently the overall cost function are continuously differentiable with respect to the input variables. However, as it can be seen from (2), this assumption is no longer valid due to x_o^s terms, although the convexity is preserved. Here, we will show that the proof holds even if the cost function is not differentiable using convex analysis and the concept of subgradient.[9] We will closely follow [18]. Collecting the terms of (9) for all sources, we have:

$$x(k+1) = \Pi_\Theta[x(k) - a(k)\hat{g}(k)], \tag{13}$$

where $x(k) = (x_s(k), s \in \mathcal{S})$, $g(k) = (g_s(k), s \in \mathcal{S})$, $a(k)$ is a $N \times N$ diagonal matrix, $N = \sum_{s\in\mathcal{S}}(N_s \cdot |D^s|)$, and the diagonal entries of $a(k)$ are equal to the corresponding step sizes of different sources, $a_s(k)$.

We can follow the definitions given in [18], with the exception that all gradient terms $\nabla C(x)$ are now replaced by a subgradient $sg(x)$ that satisfies certain conditions as will be specified shortly. Rewrite (13) in the following form

$$\begin{aligned} x(k+1) &= x(k) + a(k)[-sg(x(k)) + \xi(k) + b(k)] + \tau(k) + \phi(k), \\ &= v(k) + \tau(k) + \phi(k) \end{aligned}$$

[9] See [17] for details on subgradients ($sg(x)$) and subdifferentials ($\partial C(x)$).

where

$$\begin{aligned}
v(k) &= x(k) + a(k)[-sg(x(k)) + \xi(k) + b(k)], \\
\xi(k) &= E[\hat{g}(k)|x(k)] - \hat{g}(k), \\
b(k) &= sg(x(k)) - E[\hat{g}(k)|x(k)], \\
\tau(k) &= \Pi_\Theta[v^\gamma(k)] - v^\gamma(k), \\
v^\gamma(k) &= x(k) + a(k)[-sg(x(k)) + \xi(k) + b(k)]I(k), \\
\phi(k) &= v^\gamma(k) - v(k) + (\Pi_\Theta[v(k)] - x(k))(1 - I(k)),
\end{aligned}$$

$I(k)$ denotes the indicator function of the event $\{||\, a(k)\xi(k)|| \leq \gamma(k)/2\}$, and $\gamma(k)$ is a sequence of positive real numbers such that (i) $\gamma(k) \to 0$ and (ii) $||\, a(k)\xi(k)|| \leq \gamma(k)/2$ for all but a finite number of k w.p. 1. The following lemma guarantees the existence of such a sequence.

Lemma 1 *Under the assumptions A1-A4, for the SPSA gradient estimator $\hat{g}(k)$ defined in (10), the bias and error terms given by $b(k)$ and $\xi(k)$, respectively, satisfy*

(a) $b(k) \to 0$ *w.p. 1,*

(b) $\sum_{k=0}^{\infty} E[||\, a(k)\xi(k)||^2] < \infty$ *w.p. 1.*

Proof See Appendix II.

Since $\sum_{i=k}^{j} a(i)\xi(i),\ j \geq k$ is a martingale and $\sum_{i=k}^{\infty} E[||a(i)\xi(i)||^2] < \infty$, by the Doob-Kolmogorov inequality ([24, p. 309]), for each $\epsilon > 0$, we have

$$P(\sup_{j \geq k} ||\sum_{i=k}^{j} a(i)\xi(i)|| \geq \epsilon) \leq \frac{1}{\epsilon^2}\sum_{i=k}^{\infty} E[||\, a(i)\xi(i)||^2] \ . \tag{14}$$

Now since the right-hand side of (14) goes to zero as $k \to \infty$, it follows that

$$\lim_{k \to \infty} P(\sup_{j \geq k} ||\sum_{i=k}^{j} a(i)\xi(i)|| \geq \epsilon) = 0 \ ,$$

and the existence of the sequence $\gamma(k)$ is guaranteed.

Now let $X^0(t)$ be the continuous time interpolation of $x(k)$, where the trajectory of this continuous time process follows that of the differential inclusion to be defined shortly, and the path of this differential inclusion will be shown to converge to the solution of the optimization problem we consider.

$$X^0(t) := \frac{t_{k+1} - t}{\hat{a}(k)} x(k) + \frac{t - t_k}{\hat{a}(k)} x(k+1) \ , \ t \in (t_k, t_{k+1}) \tag{15}$$

where $t_k = \sum_{i=1}^{k-1} \hat{a}(i)$.[10]

Let

[10] Recall that $\hat{a}(k) = \max_{s \in \mathcal{S}} a_s(k)$.

$$
\begin{aligned}
B^0(t_k) &:= \sum_{i=1}^{k-1} a(i)b(i), \quad & M^0(t_k) &:= \sum_{i=1}^{k-1} a(i)\xi(i), \\
\tau^0(t_k) &:= \sum_{i=1}^{k-1} \tau(i), & \Phi^0(t_k) &:= \sum_{i=1}^{k-1} \phi(i)
\end{aligned}
$$

and the corresponding piecewise linear interpolations for $t \in (t_k, t_{k+1})$ are defined as

$$
\begin{aligned}
B^0(t) &:= \frac{t_{k+1}-t}{\hat{a}(k)} B^0(t_k) + \frac{t-t_k}{\hat{a}(k)} B^0(t_{k+1}), \\
M^0(t) &:= \frac{t_{k+1}-t}{\hat{a}(k)} M^0(t_k) + \frac{t-t_k}{\hat{a}(k)} M^0(t_{k+1}), \\
\tau^0(t) &:= \frac{t_{k+1}-t}{\hat{a}(k)} \tau^0(t_k) + \frac{t-t_k}{\hat{a}(k)} \tau^0(t_{k+1}), \\
\Phi^0(t) &:= \frac{t_{k+1}-t}{\hat{a}(k)} \phi^0(t_k) + \frac{t-t_k}{\hat{a}(k)} \phi^0(t_{k+1})).
\end{aligned}
$$

Using (14), $X^0(t)$ can be written as

$$
\begin{aligned}
X^0(t) &= \frac{t_{k+1}-t}{\hat{a}(k)} x(k) + \frac{t-t_k}{\hat{a}(k)} \Big(x(k) + a(k)[-sg(x(k)) + \xi(k) + b(k)] + \tau(k) + \phi(k) \Big) \\
&= x(k) + \frac{t-t_k}{\hat{a}(k)} \Big(a(k)[-sg(x(k)) + \xi(k) + b(k)] + \tau(k) + \phi(k) \Big) \\
&= x(1) + \sum_{i=1}^{k-1} \Big(a(i)[-sg(x(i)) + b(i) + \xi(i)] + \tau(i) + \phi(i) \Big) \\
&\quad + \frac{t-t_k}{\hat{a}(k)} \Big(a(k)[-sg(x(k)) + \xi(k) + b(k)] + \tau(k) + \phi(k) \Big) \\
&= x(1) + B^0(t) + M^0(t) - \sum_{i=1}^{k-1} \hat{a}(i) sg(x(i)) - (t-t_k) sg(x(k)) + Z^0(t) + \tau^0(t) + \Phi^0(t) \\
&= x(1) + B^0(t) + M^0(t) + H^0(t) + Z^0(t) + \tau^0(t) + \Phi^0(t),
\end{aligned}
$$

where

$$
\begin{aligned}
Z^0(t) &= \sum_{i=0}^{k-1} (\hat{a}(i) - a(i)) sg(x(i)) + \frac{t-t_k}{\hat{a}(k)} (\hat{a}(i) - a(i)) sg(x(k)) \ , \\
H^0(t) &= -\int_0^t sg(\bar{x}(s)) ds \ , \\
\bar{x}(s) &= x(k), \ s \in [t_k, t_{k+1}) \ .
\end{aligned}
$$

Since we are interested in the tail properties of the interpolated process, let us define the time shifted and centered process $X^k(t)$:

$$
X^k(t) := x(k) + B^k(t) + M^k(t) + H^k(t) + Z^k(t) + \tau^k(t) + \Phi^k(t) \ ,
$$

where

$$
B^k(t) = B^0(t_k + t) - B^0(t_k),
$$

$$\begin{aligned} M^k(t) &= M^0(t_k+t) - M^0(t_k), \\ \tau^k(t) &= \tau^0(t_k+t) - \tau^0(t_k), \\ \Phi^k(t) &= \phi^0(t_k+t) - \phi^0(t_k), \\ Z^k(t) &= Z^0(t_k+t) - Z^0(t_k), \\ H^k(t) &= -\int_0^t sg(\bar{x}(t_k+s)ds\ . \end{aligned}$$

To show convergence, we need to verify the equicontinuity of each term in (16). There is a null set Ω_0 such that for each outcome $\omega \notin \Omega_0$, $B^k(t)$, $M^k(t)$ and $\Phi^k(t)$ converge to zero uniformly on finite intervals as $k \to \infty$ as a consequence of Lemma 1. Under the assumption A6 and given the fact that $sg(x)$ is bounded due to A1, it is easy to verify that $Z^k(t) \to 0$ as $k \to \infty$. Hence, all the terms other than $H^k(t)+\tau^k(t)$ go to zero as $k \to \infty$. By the same argument as in [25, Theorem 5.3.1], $\tau^k(t)$ is equicontinuous under A5. The equicontinuity of $H^k(t)$ follows from Proposition 1 in [17]. Hence as a consequence of Arzela-Ascoli Theorem ([26, p. 72]), for $\omega \notin \Omega_0$, there exists a subsequence k_j such that

$$\{X^{k_j}(\omega,.), H^{k_j}(\omega,.)\}$$

converges to $\{X(\omega,.), H(\omega,.)\}$. As shown in [17], $H(\omega,t) = \int_0^t \tilde{h}(\omega,s)ds$, where $\tilde{h}(\omega,s) \in -\partial C(X(\omega,s))$. Then, one can write $X(\omega,t) = X(\omega,0) + \int_0^t \tilde{h}(\omega,s)ds + \tau(\omega,t)$, and using similar arguments in [25, Theorem 5.3.1], $\tau(\omega,t) = \int_0^t \tilde{\tau}(\omega,s)ds$, where $\tilde{\tau}(\omega,s) \in -V(x)$ for almost all s and $V(x)$ is the convex cone generated by the set of outward normals $\{y : y = \nabla q_i(x), \text{ s.t.}\, q_i(x) = 0\}$ and $q_i(\cdot)$ are the constraints of the optimization problem.

Hence, the limit $X(w,.)$ of any convergent subsequence satisfies the differential inclusion:

$$\dot{x} \in -\partial C(x) + \tau, \quad \tau(t) \in -V(x(t)). \tag{16}$$

As shown in [17], $X^k(\omega)$ converges to S_Θ w.p. 1, where S_Θ is the stationary set of points of (16) in Θ, i.e., points in Θ where $0 \in -\partial C(x) + \tau$. Since $C(\cdot)$ is a convex function and Θ is a nonempty convex set, any point in S_Θ attains the minimum of $C(\cdot)$ relative to Θ.

Appendix- II
Proof of Lemma 1

Let $\bar{\Delta}_s(k)$ be an $N \times 1$ vector, where values of entries corresponding to those of source s are $\Delta_{s,i}(k)$ and zero otherwise. Hence, $\Delta(k) = \sum_{s \in S} \bar{\Delta}_s(k) = (\Delta_{s,i}, s \in S, i \in 1, 2, \cdots, N_s * |D^s|)$. Similarly, u_s is an $N \times 1$ vector, where the values of entries corresponding to those of source s are one and zero otherwise. As shown in [18], there exists a finite K_1 such that $\forall k > K_1$

$$C^+(k) = C\left(x(k) + \sum_{s' \in S} c_{s'}(k)\tilde{\Delta}_{s'}(k)\right)\ , \tag{17}$$

where

$$\tilde{\Delta}_{s'}(k) = \bar{\Delta}_{s'}(k) - \frac{\sum_{j=1}^{N_{s'}} \Delta_{s',j}(k)}{N_{s'}} u_{s'}\ .$$

Following Lemma 1 of [17], define for $k > K_1$

$$\begin{aligned} G_k &:= \frac{C^+(k) - C^-(k)}{c_s(k)} \\ &= \frac{C\left(x(k) + \sum_{s' \in S} c_{s'}(k)\tilde{\Delta}_{s'}(k)\right) - C(x(k))}{c_s(k)} \\ &= \frac{C\left(x(k) + c_s(k)\hat{\Delta}(k)\right) - C(x(k))}{c_s(k)}, \end{aligned}$$

where

$$\hat{\Delta}(k) = \sum_{s' \in S} \frac{c_{s'}(k)}{c_s(k)} \tilde{\Delta}_{s'}(k)$$

Since (i) $\left(\frac{c_s(k)}{c_{s'}(k)}\right)^2 \to 1$ from A4, which implies $\hat{\Delta}(k)$ is bounded, and (ii) $C(\cdot)$ is convex and continuous, by Lemma 1 of [17] $\forall \epsilon > 0$, $\exists\ K_2 < \infty$ such that $\forall k \geq K_2$, w.p. 1

$$\left| C'\left(x(k); \hat{\Delta}(k)\right) - G_k \right| < \epsilon$$

where $C'\left(x(k); \Delta\hat{(k)}\right)$ is the one-sided directional derivative of C at $x(k)$ with respect to vector $\hat{\Delta}(k)$ as defined in [17]. We know that there exists a subgradient $sg(x(k)) \in \partial C(x(k))$ such that

$$\begin{aligned} C'\left(x(k); \Delta\hat{(k)}\right) &= sg^T(x(k))\hat{\Delta}(k) \\ &= sg^T(x(k)) \sum_{s' \in S} \frac{c_{s'}(k)}{c_s(k)} \tilde{\Delta}_{s'}(k) \end{aligned}$$

As $\frac{c_{s'}(k)}{c_s(k)} \to 1$, there exists a finite $K_3 \geq K_2$ such that $\forall \epsilon > 0$ and $k > K_3$, with probability one

$$\left| \frac{C^+(k) - C^-(k)}{c_s(k)} - sg^T(x(k)) \sum_{s' \in S} \tilde{\Delta}_{s'}(k) \right| < \epsilon\ . \tag{18}$$

As a consequence, for $k > \max\{K_1, K_3\}$,

$$\begin{aligned} E\left[\,\hat{g}_{s,i}(k) | x(k)\,\right] &= \frac{N_s}{N_s - 1} E\left[\frac{C^+(k) - C^-(k) + \mu_s^+(k) - \mu_s^-(k)}{c_s(k)\Delta_{s,i}(k)} \Big| x(k) \right] \\ &= \frac{N_s}{N_s - 1} E\left[\frac{E\left[C^+(k) - C^-(k) | \Delta(k)\right]}{c_s(k)\Delta_{s,i}(k)} \Big| x(k) \right] \\ &= \frac{N_s}{N_s - 1} E\left[\frac{sg^T(x(k)) \sum_{s' \in S} \tilde{\Delta}_{s'}(k)}{\Delta_{s,i}(k)} \Big| x(k) \right] + \delta(k) \\ &= \frac{N_s}{N_s - 1} \left(\frac{N_s - 1}{N_s} sg_i(x(k)) \right) + \delta(k) \\ &= sg_i(x(k)) + \delta(k) \end{aligned}$$

where

$$\delta(k) = E\left[\frac{C^+(k) - C^-(k)}{c_s(k)} - sg^T(x(k))\sum_{s' \in S} \tilde{\Delta}_{s'}(k)\right]$$

and $\lim_{k\to\infty} \delta(k) \to 0$ as a consequence (18). Hence, $b(k) \to 0$ w.p. 1.

From the assumption that $E[\mu_s^+(k) - \mu_s^-(k)|\mathcal{F}_k] = 0$ and using the independence of $\mu_s^\pm(k)$ and $\Delta_s(k)$, we can bound the second moment of $\hat{g}_s(k)$ as follows:

$$\begin{aligned} E[(\hat{g}_{s,i}(k))^2] &= E\left[\left(\frac{C^+(k) - C^-(k) + \mu_s^+(k) - \mu_s^-(k)}{c_s(k)\Delta_{s,i}(k)}\right)^2\right] \\ &= E\left[\left(\frac{C^+(k) - C^-(k)}{c_s(k)\Delta_{s,i}(k)}\right)^2 + \left(\frac{\mu_s^+(k) - \mu_s^-(k)}{c_s(k)\Delta_{s,i}(k)}\right)^2\right] \end{aligned}$$

Following a similar argument used above one can show that the first term in (19) is $O(1)$ and the second term is $O(c_s(k)^{-2})$, using the bounds on $E[(\Delta_s(k))^2]$, $E[(\Delta_s(k))^{-2}]$, and $E[(\mu_s^\pm(k))^2]$.

Therefore, $\sum_{i=k}^\infty E[||a(k)\xi(k)||^2] < \infty$ w. p. 1. since we have $\sum_{k=1}^\infty \frac{a_s^2(k)}{c_s^2(k)} < \infty$ from A4.