Efficient Simulation of Blocking Probabilities for Multi-layer Multicast Streams

Jouni Karvo

Networking Laboratory, Helsinki University of Technology,
P.O.Box 3000, FIN-02015 HUT, Finland.
Jouni.Karvo@hut.fi

Abstract. This paper presents an efficient algorithm for Monte-Carlo simulation of time blocking probabilities for multi-layer multicast streams with the assumption that blocked calls are lost. Users may join and leave the multicast connections freely, thus creating dynamic multicast trees. The earlier published algorithms are applicable to small networks or networks with few users. The present simulation algorithm is based on the inverse convolution method, and is the only effective way to handle large systems, known to the author.

1 Introduction

This paper presents an efficient algorithm for Monte-Carlo simulation of time blocking probabilities for multi-layer multicast streams with the assumption that blocked calls are lost. Consider a network with circuit switched traffic, or packet switching with strict quality guarantees, such as the IntServ architecture in the Internet. Decisions on whether to allow a new connection in the network are made according to availability of resources.

In general, traffic is a mixture of point-to-point (unicast) and point-to-multipoint (or multicast) traffic. There are well known algorithms for calculating blocking probabilities for unicast traffic in absence of multicast traffic, see e.g. [1, 2]. Multicast, however, gives rise to a multitude of new problems, (see e.g. [3]), one of which is blocking probability calculation. A model called "multicast loss system" has been developed for calculating blocking probabilities in recent years. This system comprises a tree-structured multicast network with dynamic membership. In this network, users at the leaf nodes can join or leave any of the several multicast channels offered by one source, the root of the tree. The users joining the channels form dynamic multicast connections that share the network resources. Blocking occurs when there are not enough resources available in the network to satisfy the resource requirements of a request. Blocked calls are lost. The multicast loss system may be seen as a virtual network over the real one, carrying the multicast traffic of the real network.

The time blocking probability is the probability that the system is in a state where a call cannot be established due to unavailable resources, while the call blocking probability is the probability that a user's attempt to establish a call

© Springer-Verlag Berlin Heidelberg 2002

fails due to unavailable resources. These probabilities are intimately related, and it is possible to calculate the call blocking probability in a multicast loss system knowing the time blocking probability.

Audio and video streams can be coded hierarchically [4]. In hierarchical, or layered, coding, information is separated according to its importance, and then coded and transmitted in separate streams. In the present setting a user may, depending on her needs and abilities, subscribe to the most important sub-stream only, in which case she is said to be on layer 1, or subscribe to any number r of the most important sub-streams, in which case she is on layer r. This paper studies the effective simulation of blocking probabilities for multicasted layered streams. The assumption that blocked calls are lost implies that if a user does not get the desired layer (or number of sub-streams) due to blocking, she will not get any layer. That is, there will be no re-negotiation of lower quality transmission.

Chan and Geraniotis [5] studied the system of layered video multicasting. They gave the definition of the state space, but resorted to approximations for the actual calculations. After their work, research has concentrated on non-layered multicast streams, see e.g. [6,7]. An efficient Monte-Carlo simulation method for dynamic multicast networks with single layer multicast streams has been developed by Lassila et al. [8]. This method was based on the inverse convolution method Lassila and Virtamo published in [9]. Recently, there has been progress in the case where the multicast streams are layered. Karvo et al. [10] developed an algorithm for calculating blocking probabilities of two-layer streams with Poisson arrivals and exponential holding times. They extended their study in [11] to an arbitrary number of layers, and studied the validity of the insensitivity property for different user models. The present paper provides an efficient simulation algorithm extending the inverse convolution approach of Lassila et al. [8] to this multi-layer case.

This paper is organized as follows. Section 2 presents the basic system model, and the time blocking probability calculation with exponential computational complexity. The problem of estimating time blocking probabilities is divided into smaller sub-problems in section 3. Section 4 contains the main contribution of this paper, showing how the inverse convolution method is applied to the layered multicast case. A numerical example is given in section 5, and the results are summarised in section 6.

2 Multicast Loss System

This section presents the system model and the notation for the multicast loss system. This model is the same as in [11]. Consider a network consisting of J links, indexed with $j \in \mathcal{J} = \{1, \ldots, J\}$, link j having a capacity of C_j resource units. The network is organized as a tree. The set $\mathcal{U}$ denotes the set of user populations, located at the leaves of the tree. The leaf links and the user populations connected to them are indexed with the same index $u \in \mathcal{U} = \{1, \ldots, U\}$. The set of links on the route from user population u to the root node is denoted by $\mathcal{R}_u$. The user populations downstream link j, i.e. for which link $j \in \mathcal{R}_u$, are denoted by $\mathcal{U}_j$. The size of the set $\mathcal{U}_j$ is denoted by U_j. Let $\mathcal{M}_j$ denote the set of all links downstream link j (including link j), and $\mathcal{N}_j$ the set of neighbouring links downstream link j (excluding link j). The links of the tree are indexed so that for all $j' \in \mathcal{N}_j$, $j' < j$. Thus, the root link is denoted by J. The multicast network supports I channels, indexed with $i \in \mathcal{I} = \{1, \ldots, I\}$. The channels originating from the root node represent different multicast transmissions, from which the users may choose. There are L layers. Each layer $l \in \mathcal{L} = \{1, \ldots, L\}$ has a capacity requirement of $d(l)$ capacity units. The capacity requirements are unique and $d(l) < d(l')$ for all $l < l'$, i.e. layer L contains all hierarchically coded sub-streams, layer 2 the two most important ones, and layer 1 only contains the most important sub-stream.

2.1 State Space

The states of the channels in a link define the state of that link. Each channel is in one of the states $\{0, 1, \ldots, L\}$, depending on whether the channel is *off*, or on layer $1, \ldots, L$. That is, the state of channel i on link j is $Y_{j,i} \in \{0, \ldots, L\}$. The vector $\mathbf{Y}_j = (Y_{j,i}; i \in \mathcal{I}) \in \{0, \ldots, L\}^I$ denotes the state of link j. The tuple (u, i, l) of the user population u (leaf link), channel i and layer l defines a multicast connection. The states $\mathbf{Y}_u$ of all the leaf links define the network state $\mathbf{X}$,

$$\mathbf{X} = (\mathbf{Y}_u; u \in \mathcal{U}) = (Y_{u,i}; u \in \mathcal{U}, i \in \mathcal{I}) \in \Omega \,, \tag{1}$$

where $\Omega = \{0, \ldots, L\}^{U \times I}$ denotes the network state space. The network state determines the state of any link j as follows:

$$\mathbf{Y}_j = \begin{cases} \mathbf{Y}_u, & \text{if } j = u \in \mathcal{U}\,, \\ \max\limits_{u' \in \mathcal{U}_j} (\mathbf{Y}_{u'}), & \text{otherwise}\,, \end{cases} \tag{2}$$

where $\max(\cdot)$ denotes the component-wise max-operation. The occupancy of any link j is determined by the link state as

$$S_j = D(\mathbf{Y}_j) = \sum_{i=1}^{I} d(Y_{j,i})\,, \tag{3}$$

where $d(0) = 0$, i.e. when channel is *off*, it does not need any link capacity. The occupancy generated by all other channels but I is denoted by $S'_j = D'(\mathbf{Y}_j) = \sum_{i=1}^{I-1} d(Y_{j,i})$.

Finally, in a finite capacity network, the capacity constraints of the links truncate the state space,

$$\tilde{\Omega} = \left\{ \mathbf{x} \in \Omega \,\middle|\, D(\mathbf{y}_j) \leq C_j, \forall j \in \mathcal{J} \right\}. \tag{4}$$

2.2 Probability Distributions

Let us assume that the user populations of the leaf links are independent, and that the leaf link distributions $\pi_u(\mathbf{y}) = \mathrm{P}\{\mathbf{Y}_u = \mathbf{y}\}$, $u \in \mathcal{U}$, are known, and represent stationary distributions of reversible Markov processes satisfying the detailed balance equations. Several types of user population models of this kind have been discussed in [7], and in [11].

The steady state probabilities $\pi(\mathbf{x})$ of the network states in a system with infinite link capacities can be calculated from

$$\pi(\mathbf{x}) = \mathrm{P}\{\mathbf{X} = \mathbf{x}\} = \prod_{u \in \mathcal{U}} \pi_u(\mathbf{y}_u)\,, \tag{5}$$

since the user populations are independent. The inverse convolution approach also dictates that all channels shall be independent. Thus,

$$\pi_u(\mathbf{y}_u) = \prod_{i \in \mathcal{I}} p_{u,i}(y_{u,i})\,. \tag{6}$$

As already noted in [11], probabilities $\tilde{\pi}(\mathbf{x})$, $\mathbf{x} \in \tilde{\Omega}$, of states in a system with finite link capacities are obtained by truncation

$$\tilde{\pi}(\mathbf{x}) = \mathrm{P}\{\mathbf{X} = \mathbf{x} \,|\, \mathbf{X} \in \tilde{\Omega}\} = \frac{\pi(\mathbf{x})}{\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}}\,, \tag{7}$$

where $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\} = \sum_{\mathbf{x} \in \tilde{\Omega}} \pi(\mathbf{x})$. This follows from the assumed detailed balance. See Kelly [12] for discussion of truncation.

2.3 Blocking

In a finite capacity network, blocking occurs whenever a user tries to establish a connection for channel i and layer r, and there is at least one link $j \in \mathcal{R}_u$ where the channel is on state $l < r$ and there is not enough spare capacity for setting the channel on the requested layer. Without loss of generality, the channels are ordered so that the blocking probability is calculated for channel with index I. Consider link j. A request for layer r succeeds if there is enough capacity already reserved for the layer in link j, or there is enough free capacity in the link, i.e. $\max\{d(r), d(y_{j,I})\} \le C_j - D'(\mathbf{y}_j)$. The expression “link j blocks” means that this condition does not hold for link j. The set $\mathcal{B}_{u,r}$ consists of the states where at least one link blocks for connection (u, I, r), when layer r of channel I is requested by user u, and is defined as

$$\mathcal{B}_{u,r} = \left\{ \mathbf{x} \in \tilde{\Omega} \,\middle|\, \exists j \in \mathcal{R}_u : d(r) > C_j - D'(\mathbf{y}_j) \right\}. \tag{8}$$

Then the time blocking probability for connection (u, I, r) is

$$B_{u,r} = \mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r} \,|\, \mathbf{X} \in \tilde{\Omega}\} = \frac{\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\}}{\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}}\,. \tag{9}$$

Call blocking probabilities for users depend on the chosen user model, as discussed in [11]. Calculation of time blocking probabilities for layers is easy, but very time consuming: the number of states in the state space is $(L+1)^{UI}$. The following section attacks this problem using the inverse convolution method.

3 Divide and Conquer

This section discusses efficient estimation of time blocking probabilities by applying the algorithm developed by Lassila and Virtamo [9]. As the form of the stationary distribution $\pi(\mathbf{x})$ is known, a natural choice for simulation is the Monte Carlo method. The main problem in the simulation is to quickly get a good estimate for $\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\}$, i.e., the numerator in Eq. (9), especially in the case when the blocking probability $B_{u,r}$ is small. Note that $B_{u,r}$ also depends on $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}$ given by the denominator of Eq. (9). This probability is usually close to unity and is easy to estimate using the standard Monte Carlo method. Therefore, the rest of this paper concentrates on efficient methods for estimating $\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\}$.

First, section 3.1 divides the task of estimating $P(\mathcal{B}_{u,r})$ into simpler sub-problems. Then, each of the sub-problems is solved using importance sampling, as is described in section 3.2.

3.1 Decomposition

In order to divide the task of estimating $P(\mathcal{B}_{u,r})$ to simpler sub-problems, $\mathcal{B}_{u,r}$ is partitioned into sets $\mathcal{E}^j_{u,r}$. $\mathcal{E}^j_{u,r}$ is defined as the set of points in $\mathcal{B}_{u,r}$ where link j blocks but none of the links closer to user u block,

$$\begin{aligned}\mathcal{E}^j_{u,r} = \mathcal{B}_{u,r} \cap \Big\{ \mathbf{x} \in \tilde{\Omega} \,\Big|\, & d(r) > C_j - D'(\mathbf{y}_j) \;\wedge \\ & d(r) \le C_{j'} - D'(\mathbf{y}_{j'}), \forall j' \in \mathcal{R}^j_u \Big\}\,,\end{aligned} \tag{10}$$

where $\mathcal{R}^j_u$ denotes the set of links on the path from u to j, including link u but not link j. The $\mathcal{E}^j_{u,r}$ form a partitioning of $\mathcal{B}_{u,r}$, i.e. $\mathcal{B}_{u,r} = \bigcup_{j \in \mathcal{R}_u} \mathcal{E}^j_{u,r}$, and $\mathcal{E}^j_{u,r} \cap \mathcal{E}^{j'}_{u,r} = \emptyset$, when $j \neq j'$. From this it follows that

$$\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\} = \sum_{j \in \mathcal{R}_u} \mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}\,. \tag{11}$$

The probability $\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$ can be thought of as the blocking probability contribution due to link j. It should be noted, however, that blocking in the states where several links block can be arbitrarily attributed to any of the blocking links. I use the convention which attributes it to the blocking link closest to the user.

3.2 Conditioning of $\mathbf{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$

Equation (11) decomposes estimation of $\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\}$ into independent sub-problems of estimating the $\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$. For these estimation tasks, I introduce the superset $\mathcal{D}^j_{u,r} \supset \mathcal{E}^j_{u,r}$,

$$\mathcal{D}^j_{u,r} = \left\{ \mathbf{x} \in \Omega \,\middle|\, d(r) > C_j - D'(\mathbf{y}_j) \geq d(y_{j,I}) \right\}. \tag{12}$$

This set corresponds to blocking states in a system where link j has a finite capacity C_j but all other links have infinite capacity. Since all links have finite capacity in real systems, and several links could block simultaneously, sets $\mathcal{D}^j_{u,r}$ are not disjoint unlike their subsets $\mathcal{E}^j_{u,r}$.

The next step is to use conditional probabilities to estimate $\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$, as follows:

$$\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\} = \mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r} \,|\, \mathbf{X} \in \mathcal{D}^j_{u,r}\}\mathrm{P}\{\mathbf{X} \in \mathcal{D}^j_{u,r}\}\,. \tag{13}$$

This relation is useful from the simulation point of view since it is easy to compute $\mathrm{P}\{\mathbf{X} \in \mathcal{D}^j_{u,r}\}$ and to generate samples from the original distribution under the condition $\mathbf{X} \in \mathcal{D}^j_{u,r}$, as explained later. Monte Carlo simulation is then used to estimate the conditional probability $\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r} \,|\, \mathbf{X} \in \mathcal{D}^j_{u,r}\}$ instead of $\mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$, which is usually much more effective.

Let $\widehat{\eta}^j_{u,r}$ denote the estimator for $\eta^j_{u,r} = \mathrm{P}\{\mathbf{X} \in \mathcal{E}^j_{u,r}\}$,

$$\widehat{\eta}^j_{u,r} = \frac{v_j}{N_j} \sum_{n=1}^{N_j} 1_{\mathbf{X}^*_n \in \mathcal{E}^j_{u,r}}\,, \tag{14}$$

where $v_j = \mathrm{P}\{\mathbf{X} \in \mathcal{D}^j_{u,r}\}$ and $\mathbf{X}^*_n$ denotes samples drawn from the conditional distribution $\mathrm{P}\{\mathbf{X} = \mathbf{x} | \mathbf{X} \in \mathcal{D}^j_{u,r}\}$. Then, the estimator for $P(\mathcal{B}^j_{u,r})$ is simply

$$\widehat{P}(\mathcal{B}^j_{u,r}) = \sum_{j \in \mathcal{R}_u} \widehat{\eta}^j_{u,r}\,. \tag{15}$$

Given the total number of samples N to be used for the estimator, the number of samples N_j allocated to each sub-problem is a free parameter. This can be exploited by assigning the number of samples to different $\widehat{\eta}^j_{u,r}$ according to their estimated variance during the simulation. See e.g. [8].

4 Inverse Convolution

This section presents the inverse convolution method (IC) for sample generation. I am now only considering the estimation of one $\eta^j_{u,r}$ for fixed $j \in \mathcal{R}_u$ and traffic class (u, I, r). The method is based on generating points from the conditional distribution $\mathrm{P}\{\mathbf{X} = \mathbf{x} \,|\, \mathbf{X} \in \mathcal{D}^j_{u,r}\}$ by reversing the steps used to calculate the

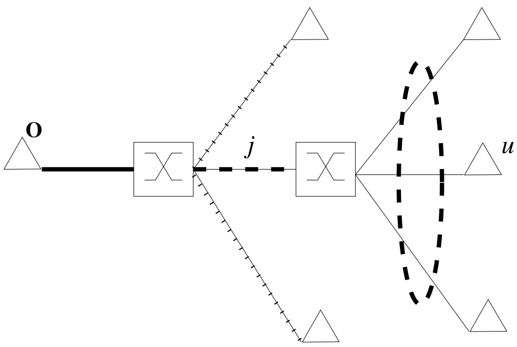

Fig. 1. Example of sample generation. A sample in the set $\mathcal{D}^j_{u,r}$ is generated for the link j (thick dashed line). States of the links marked by the dashed ellipse are generated by inverse convolution from the state of link j. States for links denoted by ticks are generated by a simple draw. The state of the link denoted by the thick line is calculated directly from the states of the other links.

occupancy distribution of the considered link. Note that the condition $\mathbf{X} \in \mathcal{D}^j_{u,r}$ is a condition expressed in terms of the occupancy, S'_j, of the considered link. The idea in the inverse convolution method is to first generate a sample of $\mathbf{Y}_j$ such that the occupancy of the link is in the blocking region. Then, given the state $\mathbf{Y}_j$, the state of the network, i.e. states of the leaf links, is generated. The mapping $\mathbf{x} \mapsto \mathbf{y}_j$ is surjective, having several possible network states $\mathbf{x}$ generating the link state $\mathbf{y}_j$, and one of them is drawn according to their probabilities.

The main steps of the simulation can be summarized as follows (See Figure 1.):

1. Generate the states for leaf links u by
   a) Generate a sample state $\mathbf{Y}_j$ under the condition $d(r) > C_j - D'(\mathbf{y}_j) \geq d(y_{j,I})$ for link j.
   b) Generate the leaf link states $\mathbf{Y}_u$, $u \in \mathcal{U}_j$, with the condition that link j state $\mathbf{Y}_j = \max_{u \in \mathcal{U}_j}(\mathbf{Y}_u)$ is given.
   c) Generate the states $\mathbf{Y}_u$, $u \in \mathcal{U} - \mathcal{U}_j$ for the rest of the leaf links as in the normal Monte Carlo simulation.
2. The sample state of the network $\mathbf{X}^*_n \in \mathcal{D}^j_{u,r}$ consists of the set of all sample states of leaf links generated with step 1.
3. To collect the statistics for estimator $\widehat{\eta}^j_{u,r}$, check if $\mathbf{X}^*_n \in \mathcal{E}^j_u$.

The above steps are repeated for generating N_j samples. Section 4.1 explains the method of generating a sample for link j (step 1a). Section 4.2 explains the method for generating the leaf link states from the link state (step 1b).

4.1 Generating a Sample for $\mathcal{D}^j_{u,r}$

As already noted, I have partitioned the set of blocking states into disjoint sets $\mathcal{E}^j_{u,r}$. It is not easy to generate samples directly to these sets, however. Instead, I generate samples to sets $\mathcal{D}^j_{u,r}$ which correspond to the states in which at least link j blocks. After that it is possible to check if the sample belongs to the set $\mathcal{E}^j_{u,r}$ to collect the sum in Eq. (14).

Convolution method for calculating $\mathbf{P\{X \in \mathcal{D}^j_{u,r}\}}$. First, the link occupancy S_j is easily calculated recursively as follows. Let $S_{j,i}$ denote link occupancy due to the first i channels,

$$S_{j,i} = \sum_{i' \le i} d(Y_{j,i'}) \,. \tag{16}$$

Then $S_j = S_{j,I}$ and $S'_j = S_{j,I-1}$. The $Y_{j,i}$ are mutually independent, and $S_{j,i} = S_{j,i-1} + d(Y_{j,i})$, where $S_{j,i-1}$ and $Y_{j,i}$ are independent.

Channel I must be dealt with differently than the other channels, since the system can be in a blocking state only if $C_j - S_{j,I-1} < d(r)$, but the channel I can be in any state $l < r$. Knowing this, the set $\mathcal{D}^j_{u,r}$ can be partitioned into r point-wise disjoint subsets:

$$\begin{aligned}\mathcal{D}^{j,l}_{u,r} = \Big\{\mathbf{x} \in \Omega \,\Big|\, & y_{j,I} = l \,\wedge \\ & d(r) > C_j - D'(\mathbf{y}_j) \ge d(l)\Big\}, \qquad l \in \{0, \dots, r-1\}\,.\end{aligned} \tag{17}$$

If a state $\mathbf{x}$ belongs to the set $\mathcal{D}^{j,l}_{u,r}$, the state is a blocking state for link j, and the channel I is on layer l. Thus, the free capacity $C_j - D'(\mathbf{y}_j)$ of the link must be at most $d(r) - 1$, for the state to be a blocking state. The other channels may, however, consume at most $C_j - d(l)$ capacity units for the state to be within the allowed states. Now, let $v_j(l)$ denote the probability $\mathrm{P}\{\mathbf{X} \in \mathcal{D}^{j,l}_{u,r}\}$:

$$v_j(l) = p_{j,I}(l) \sum_{i=C_j-d(r)+1}^{C_j-d(l)} q_{j,I-1}(i)\,, \tag{18}$$

where $q_{j,i}(x) = \mathrm{P}\{S_{j,i} = x\}$. The probability mass v_j of the set $\mathcal{D}^j_{u,r}$, can be calculated as

$$v_j = \mathrm{P}\{\mathbf{X} \in \mathcal{D}^j_{u,r}\} = \sum_{l=0}^{r-1} v_j(l)\,. \tag{19}$$

The link occupancy distribution $q_{j,I-1}(\cdot)$ can be calculated recursively by convolution:

$$q_{j,i}(x) = \sum_{y=0}^{x} q_{j,i-1}(x - d(y)) p_{j,i}(y)\,, \tag{20}$$

where the recursion starts with $q_{j,0}(x) = 1_{x=0}$. Here, $p_{j,i}(y) = \mathrm{P}\{Y_{j,i} = y\}$, and is calculated easily, as shown in section 4.2.

Inverse convolution. For interpretation of the convolution step, note that the event $\{S_{j,i} = x\}$ is the union of the events $\{Y_{j,i} = y, S_{j,i-1} = x - d(y)\}$, $y \in \{0, \ldots, L\}$. The corresponding probability is $q_{j,i-1}(x - d(y))p_{j,i}(y)$. Conversely, the conditional probability of the event $\{Y_{j,i} = y,\, S_{j,i-1} = x - d(y)\}$ given that $S_{j,i} = x$ is,

$$\mathrm{P}\{Y_{j,i} = y,\, S_{j,i-1} = x - d(y) \,|\, S_{j,i} = x\} = \frac{p_{j,i}(y)q_{j,i-1}(x - d(y))}{q_{j,i}(x)}\,. \tag{21}$$

Generating a sample state in $\mathcal{D}^j_{u,r}$ starts by drawing a value l for $Y_{j,I}$ using the distribution

$$\mathrm{P}\{Y_{j,I} = l \,|\, \mathbf{X} \in \mathcal{D}^j_{u,r}\} = \frac{\mathrm{P}\{Y_{j,I} = l,\, \mathbf{X} \in \mathcal{D}^j_{u,r}\}}{\mathrm{P}\{\mathbf{X} \in \mathcal{D}^j_{u,r}\}} = \frac{v_j(l)}{v_j}\,, \tag{22}$$

where $l \in \{0, \ldots, r-1\}$.

Then, a value for $S'_j = S_{j,I-1}$ is drawn with the condition that $Y_{j,I} = l$ that is, using the distribution

$$p(x|l) = \mathrm{P}\{S_{j,I-1} = x \,|\, Y_{j,I} = l,\, \mathbf{X} \in \mathcal{D}^j_{u,r}\} = \frac{\mathrm{P}\{Y_{j,I} = l, S_{j,I-1} = x\}}{\mathrm{P}\{Y_{j,I} = l,\, \mathbf{X} \in \mathcal{D}^j_{u,r}\}}\,, \tag{23}$$

since $\{Y_{j,I} = l \wedge S_{j,I-1} = x\} \Rightarrow \{\mathbf{X} \in \mathcal{D}^j_{u,r}\}$, restricting x to $x \in \{C_j - d(r) + 1, \ldots, C_j - d(l)\}$, and

$$p(x|l) = \frac{p_{j,I}(l)q_{j,I-1}(x)}{v_j(l)} = \frac{q_{j,I-1}(x)}{\sum_{y=C_j-d(r)+1}^{C_j-d(l)} q_{j,I-1}(y)}\,. \tag{24}$$

Then, given the value of $S_{j,I-1}$, the state $Y_{j,i}$ of each channel $(i = I-1, \ldots, 1)$ is drawn in turn using probabilities in Eq. (21). Concurrently with the state $Y_{j,i}$, the value of $S_{j,i-1}$ becomes determined. This is then used as the conditioning value in the next step to draw the value of $Y_{j,i-1}$ (and of $S_{j,i-2}$), etc. Note that for reasonable sizes of links, it is advantageous to store the probabilities for fast generation of samples.

The next subsection presents a method for drawing leaf link states $\mathbf{Y}_u$, given the state $\mathbf{Y}_j$ of link j.

4.2 Generating Leaf Link States from a Link State

Having drawn a value for state $\mathbf{Y}_j$ of link j, it is possible to draw values of the state vectors $\mathbf{Y}_u$, $u \in \mathcal{U}$, of the leaf links. For $u \in \mathcal{U}_j$, states $\mathbf{Y}_u$ are generated under the condition $\mathbf{Y}_j = \max_{u \in \mathcal{U}_j}(\mathbf{Y}_u)$ using a similar inverse convolution procedure as above. Due to the assumed independence of channels, this condition can be broken down into separate conditions, i.e. for each i there is a separate problem of generating the values $Y_{u,i}$, $u \in \mathcal{U}$, under the condition $Y_{j,i} = \max_{u \in \mathcal{U}_j}(Y_{u,i})$ with a given $Y_{j,i}$. The above conditions affect leaf links

$u \in \mathcal{U}_j$. For other links $u \in \mathcal{U} - \mathcal{U}_j$, the states $\mathbf{Y}_u$ are independently generated from the distribution $\pi_u(\cdot)$.

First, let us consider a convolutional approach for generating a link state for channel i and link j if the states for each link $u \in \mathcal{U}_j$ are already known. In this section, I use an index $u_j \in \{1, \ldots, U_j\} = \mathcal{U}_j$ for the subset of leaf links. Let $Z_{u_j,i} = x$ denote the event that the channel i is on state x on link j when $u' = 1, \ldots, u_j$ leaf links have been counted for, i.e. $Z_{u_j,i} = \max_{u' \leq u_j}(Y_{u',i})$. Note that $Y_{j,i} = Z_{U_j,i}$. Probabilities $\xi_{u_j,i}(s) = \mathrm{P}\{Z_{u_j,i} = s\}$ can be calculated recursively as follows:

$$\xi_{u_j,i}(s) = p_{u_j,i}(s) \sum_{s'=0}^{s-1} \xi_{u_j-1,i}(s') + \xi_{u_j-1,i}(s) \sum_{s'=0}^{s} p_{u_j,i}(s') \,. \tag{25}$$

The recursion starts with $\xi_{0,i}(s) = 1_{s=0}$. The probabilities $p_{j,i}(s)$ used in the previous section are then simply $p_{j,i}(s) = \xi_{U_j,i}(s)$ where all users have been taken into account. If $Z_{u_j-1,i} = s$, then necessarily $Z_{u_j,i} \geq s$ (due to the nature of max-operation).

Conversely, to generate the state for each leaf link, given the value of $Y_{j,i}$, I first generate $Z_{u_{j-1},i}$ from the distribution:

$$\mathrm{P}\{Z_{u_j-1,i} = x \,|\, Z_{u_j,i} = s\} = \begin{cases} \dfrac{\xi_{u_j-1,i}(x) \sum_{s'=0}^{x} p_{u_j,i}(s')}{\xi_{u_j,i}(s)}, & \text{when } x = s\,, \\ \dfrac{\xi_{u_j-1,i}(x) p_{u_j,i}(s)}{\xi_{u_j,i}(s)}, & \text{otherwise}\,. \end{cases} \tag{26}$$

Note that the event $Z_{u_j-1,i} < Z_{u_j,i}$ implies directly that $Y_{u_j,i} = Z_{u_j,i}$. If this is not the case, the value of $Y_{u_j,i}$ is drawn from the distribution

$$\mathrm{P}\{Y_{u_j,i} = y \,|\, Z_{u_j-1,i} = Z_{u_j,i} = s\} = \frac{p_{u_j,i}(y)}{\sum_{y'=0}^{s} p_{u_j,i}(y')}\,. \tag{27}$$

This procedure is repeated for each channel. The state vectors of each leaf link $u \in \mathcal{U}_j$ result from this procedure. The rest of the leaf link states must be generated as in the normal Monte Carlo simulation using distribution $\pi_u(\cdot)$.

5 Numerical Results

This section gives some numerical examples to illustrate the efficiency of the presented method in Monte Carlo simulation of the blocking probabilities. I consider the same network used in [7]. The network is the one shown in Figure 1. There is a root node, four channels, $I = 4$, and three layers, $L = 3$, with $d(l) = l$ for all channels. The capacity of the root link is $C_J = 6$, for the others, $C_j = 5$. Each leaf link has an infinite user population offering traffic to each channel. The probability $p_{u,i}(l)$ that a channel is on layer l is $p_{u,i}(l) = \alpha_l b$ (for all users), where $\alpha_1 = 0.3$, $\alpha_2 = 0.2$ and $\alpha_3 = 0.1$. I simulated blocking for channel I and

Table 1. The relative deviation of the estimates $\widehat{P}(\mathcal{B}_{u,r})$ for the example network

Samples	b	r / $B_{u,r}$	relative deviation		
			MC	MC-IC	MC-ICSA
10 000	0.01	1 / 0.0146%	0.6301	0.0060	0.0048
		2 / 0.0591%	0.4244	0.0063	0.0049
		3 / 0.98%	0.0948	0.0078	0.0049
	0.05	1 / 0.33%	0.1240	0.0067	0.0056
		2 / 1.28%	0.0748	0.0068	0.0055
		3 / 6.12%	0.0384	0.0076	0.0060
	0.10	1 / 1.14%	0.0564	0.0073	0.0063
		2 / 4.25%	0.0413	0.0073	0.0060
		3 / 14.0%	0.0227	0.0079	0.0068
100 000	0.01	1 / 0.0146%	0.2075	0.0019	0.0015
		2 / 0.0591%	0.1273	0.0020	0.0015
		3 / 0.98%	0.0281	0.0025	0.0016
	0.05	1 / 0.33%	0.0398	0.0021	0.0018
		2 / 1.28%	0.0227	0.0022	0.0017
		3 / 6.12%	0.0128	0.0024	0.0019
	0.10	1 / 1.14%	0.0194	0.0023	0.0020
		2 / 4.25%	0.0120	0.0023	0.0019
		3 / 14.0%	0.0073	0.0025	0.0021

user u (the longer path) with three values for b: 0.01, 0.05, and 0.1 to compare the simulation methods in light, moderate, and high load conditions.

I also estimated the relative deviation of the estimator for 10^4 samples and 10^5 samples, given by $(\mathrm{V}[\widehat{P}(\mathcal{B}_{u,r})])^{1/2}/\widehat{P}(\mathcal{B}_{u,r})$. For classic Monte Carlo (MC), these were the total numbers of samples used, while for Inverse Convolution method (MC-IC), one third of samples was used for each estimate $\widehat{\eta}^{j}_{u,r}$. For Inverse Convolution with optimal Sample Allocation (MC-ICSA) [8], the total number of samples was allocated optimally for each estimate.

The results are shown in Table 1. The table shows that the variance reductions obtained with the inverse convolution method are remarkable. For example, for light load ($b = 0.01$), the ratio between the deviations of the standard MC and the inverse convolution method (MC-ICSA) is up to 131 for 10 000 samples and 138 for 100 000 samples, corresponding to a decrease by a factor of 17 000 to 19 000 in the required sample sizes. In high load situations, the overhead in sample generation might not be justified, as the traditional Monte Carlo method gives rather good estimates, too.

6 Summary

I presented an algorithm for efficient simulation of time blocking probabilities for multi-layer multicast streams with the assumption that blocked calls are lost.

Calculating blocking probabilities for this system directly from the steady state probabilities is easy in principle, but excessively time-consuming.

The simulation algorithm presented is based on the inverse convolution algorithm. The results in the shown example network support convincingly its efficiency, yielding a decrease in sample size of up to a factor of 19 000 over the traditional Monte Carlo method.

Acknowledgement. This study was funded by the Academy of Finland, and also supported by Nokia Foundation. I thank Pasi Lassila, Jorma Virtamo and Samuli Aalto for their helpful advise.

References

1. Fortet R. and Grandjean C., "Congestion in a loss system when some calls want several devices simultaneously," *Electrical Communication*, vol. 39, no. 4, pp. 513–526, 1964.
2. Ross K. W., *Multiservice Loss Models for Broadband Telecommunication Networks*, Springer Verlag, London, 1995.
3. Diot C., Dabbous W., and Crowcroft J., "Multipoint communication: A survey of protocols, functions, and mechanisms," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 277–290, Apr. 1997.
4. Karlsson G. and Vetterli M., "Packet video and its integration into the network architecture," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 5, pp. 739–751, June 1989.
5. Chan W. C. and Geraniotis E., "Tradeoff between blocking and dropping in multicasting networks," in *ICC '96 Conference Record*, June 1996, vol. 2, pp. 1030–1034.
6. Karvo J., Virtamo J., Aalto S., and Martikainen O., "Blocking of dynamic multicast connections," *Telecommunication Systems*, vol. 16, no. 3,4, pp. 467–481, 2001.
7. Nyberg E., Virtamo J., and Aalto S., "An exact algorithm for calculating blocking probabilities in multicast networks," in *Networking 2000*, Pujolle G., Perros H., Fdida S., Körner U., and Stavrakakis I., Eds., Paris, May 2000, pp. 275–286.
8. Lassila P., Karvo J., and Virtamo J., "Efficient importance sampling for Monte Carlo simulation of multicast networks," in *Proc. INFOCOM'01*, Anchorage, Alaska, Apr. 2001, pp. 432–439.
9. Lassila P. E. and Virtamo J. T., "Nearly optimal importance sampling for Monte Carlo simulation of loss systems," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 10, no. 4, pp. 326–347, Oct. 2000.
10. Karvo J., Aalto S., and Virtamo J., "Blocking probabilities of two-layer statistically indistinguishable multicast streams," in *Proc. International Teletraffic Congress ITC-17*, de Souza J. M., Fonseca N. L. S., and de Souza e Silva E. A., Eds., Salvador da Bahia, Brazil, Sept. 2001, pp. 769–779.
11. Karvo J., Aalto S., and Virtamo J., "Blocking probabilities of multi-layer multicast streams," in *2002 Workshop on High Performance Switching and Routing (HPSR 2002) (To appear)*, Kobe, Japan, May 2002.
12. Kelly F. P., *Reversibility and Stochastic Networks*, John Wiley & Sons, 1979.