Blocking probabilities of multi-layer multicast streams

Jouni Karvo, Samuli Aalto, and Jorma Virtamo
Helsinki University of Technology, P.O.Box 3000, FIN-02015 HUT, Finland.
Email: {Jouni.Karvo, Samuli.Aalto, Jorma.Virtamo}@hut.fi

Abstract

We present two new algorithms for calculating call 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. We define the state space, and give two recursive algorithms; for the general case and for the special case where all multicast channels are statistically indistinguishable. Our recursive algorithms are linear with respect to the number of links. The special case is also polynomial with respect to the number of channels.

1 Introduction

We present two new algorithms for calculating call 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 to 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. [6, 5, 16]. These models apply also for static multicast traffic. For dynamic multicast trees, however, these models do not work, since they fail to capture the network state dependent call establishment behaviour of such connections. In these networks, users at the leaf nodes can join or leave any of the several multicast channels offered by one source (i.e. the root of the tree) in the network. The users joining the channels form dynamic multicast connections that share the network resources. Note that, due to the multicast nature of traffic, the amount of resources reserved at this time depends on the network state. If there are users on that channel nearby, the reservation may concern just the access link, whereas the other extreme is that resources must be reserved along the whole route from the source to the new user. Blocking occurs when there are not enough resources available in the network to satisfy the resource requirements of a request. Since blocked calls are lost, this system is also called a "multicast loss system". The multicast loss system may be seen as a virtual network over the real one, carrying the multicast traffic of the real network.

This study was funded by the Academy of Finland. The first author was also supported by Nokia Foundation.

Audio and video streams can be coded hierarchically [7]. In hierarchical, or layered, coding, information is separated according to its importance, and then coded and transmitted separately. In this paper, we study a setting where a user may, depending on her needs and abilities, subscribe to the most important substream only, in which case we say she is on layer 1, or subscribe to any number r of the most important substreams, in which case we say she is on layer r. Our focus is on the calculation 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 substreams) due to blocking, she will not get any layer (there will be no re-negotiation).

In this paper, we give an algorithm allowing calculation of call blocking probabilities in this general setting, both in the case where the multicast loss system is in isolation (with no unicast traffic), and in the case where there is independent background traffic. The latter case refers to a scenario where multicast trees are embedded in a network with an arbitrary topology carrying also unicast traffic. The algorithm has a complexity of $O((U-1)(L+1)^{2I})$, where U denotes the number of user populations (or leaves) in the network, L the number of layers, and I the number of multicast channels in the network. Since the dependence on U is only linear, the network size will not preclude usage of the algorithm.

In addition to the general setting, we study a special case, which is applicable if channels within each layer are statistically indistinguishable. By this we mean that

- the channels are chosen with the same probabilities,
- the mean holding time for these channels is the same, and
- the capacity needed to carry a specific layer is the same for all channels on any link.

We show how to organise the problem of calculating exact time blocking probabilities for each layer under this assumption of statistical indistinguishability, yielding an algorithm with complexity $O((U-1)(I+1)^{L(L+2)})$. The algorithm is polynomial with respect to I, which is a considerable improvement from the exponential growth of the state space.

Chan and Geraniotis [4] studied the system of layered video multicasting. They gave the definition of the state space, but resorted on approximations for the actual calculations. After their work, research on multicast blocking probabilities has concentrated on non-layered multicast streams. Karvo et al. [9] studied the non-layered system under the assumption of having an infinite user population generating call requests at the leaf nodes and a single finite capacity link in the network. An exact algorithm to compute the blocking probabilities was derived. This work was extended by Boussetta and Belyot [3] by adding unicast traffic to the system. Reduced load approximations of the blocking probabilities in a network were derived in [10]. Ramanan et al. [15] have studied phase transitions in multicast calls, using simplified static multicast calls. Their study suggests, that the reduced load approximation is not adequate for multicasting, and further work is needed. An exact algorithm with complexity $O((U-1)2^{2I})$ for the network case with dynamic non-layered multicast connections has been given in Nyberg et al. [14, 13]. Efficient Monte-Carlo simulation method for dynamic multicast networks has been developed by Lassila et al. [12]. Recently, there has been slight progress in the case where the multicast streams are layered. Karvo et al. [8] developed an algorithm for calculating blocking probabilities of two-layer streams with Poisson arrivals and exponential holding times. The contribution of the present paper is a generalisation of this work to an arbitrary number of layers, and a study of insensitivity for different known user models for this system.

This paper is organised as follows. Section 2 presents the basic system model, and the basic time blocking probability calculation with exponential complexity. Section 3 presents a time blocking probability calculation algorithm with reduced complexity with respect to the network size. Section 4 shows how to efficiently calculate time blocking probabilities for statistically indistinguishable channels using a polynomial-time algorithm. Section 5 studies the insensitivity of the system, and the call blocking probabilities, when applying some basic user population models to the system. The results are summarised in section 6.

2 Multicast loss system

We study the multicast loss system using notation presented in Nyberg et al. [14], adding multilayer-specific properties to the model. Consider a network consisting of J links, indexed with $j \in \mathcal{J} = \{1, \dots, J\}$, link j having a capacity of C_j resource units. The network is organized as a tree (see Figure 1). 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 $u \in \mathcal{U} = \{1, \dots, U\}$. 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 $i \in \mathcal{I} = \{1, \dots, I\}$. The channels originating from the root node represent different multicast transmissions, from which the users may choose. There are L layers. Each layer $l \in \mathcal{L} = \{1, \dots, L\}$ has a capacity requirement $d(l)$. The capacity requirements are unique and $d(l) < d(l')$ for all $l < l'$, i.e. we assume that 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.

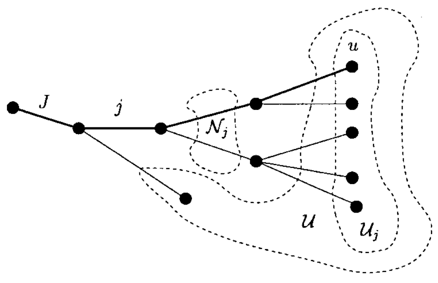

Figure 1: Network definitions. $\mathcal{R}_u$ is shown by the thick links.

2.1 State space

The state of link j is defined by the states of the channels i. Each channel is on one of the states $\{0, 1, \dots, L\}$, depending on whether the channel is *off*, or on layer $1, \dots, L$. That is, the state of channel i on link j is $Y_{j,i} \in \{0, \dots, L\}$. The state of link j is denoted by the vector $\mathbf{Y}_j = (Y_{j,i}; i \in \mathcal{I}) \in \mathcal{S}$, where $\mathcal{S}$ is the link state space $\mathcal{S} = \{0, \dots, L\}^I$. A multicast connection is defined by the tuple (u, i, l) of the user population u (leaf link), channel i and layer l. The network state $\mathbf{X}$ is defined by the states $\mathbf{Y}_u$ of the leaf links,

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

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

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

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

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

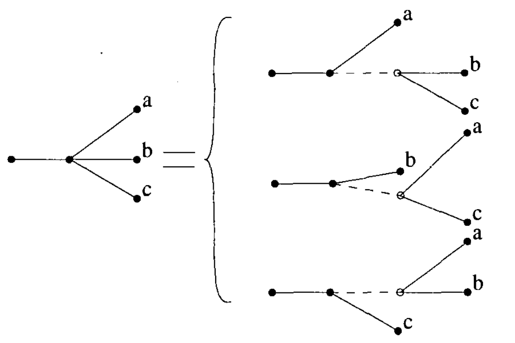

Figure 2: Original and supplemented networks. There are three possible supplemented networks, each having exactly the same states in the links from the original network (solid lines).

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

Now, in a finite capacity network, the state space is truncated by the capacity constraints of the links,

$$\tilde{\Omega} = \left\{ \mathbf{x} \in \Omega \,\middle|\, D(\mathbf{y}_j) \le C_j, \forall j \in \mathcal{J} \right\}. \tag{3}$$

Any tree network where the number of child nodes of all non-leaf nodes is greater or equal to two can be supplemented to a binary tree by adding supplemented links with infinite capacity. There are many possible ways to supplement trees, see Figure 2. The states of the original network are the same as the states in the corresponding links in the supplemented network. Because of this property, every algorithm that works properly in a binary tree-shaped networks, works in any tree network, as stated in the following lemma:

Lemma 2.1 *Given an original tree $\mathcal{J}$ and any supplemented tree $\mathcal{J}'$, the state of any link j in the original network can be calculated recursively from the leaf links $\mathcal{U}_j$ using the supplemented tree as*

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

where $\mathcal{N}_j$ is the set of downstream neighbour links of link j in the supplemented tree.

This result follows directly from the associativity and commutativity of the max(·) operation, applied recursively to the whole tree. Because of this property, we can use a supplemented tree to calculate the link probabilities instead of the original network. In the sequel we assume that the multicast loss system is first supplemented to a binary tree.

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 [13], and will be presented in section 5.

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

$$\pi(\mathbf{x}) = \mathrm{P}\{\mathbf{X} = \mathbf{x}\} = \prod_{u \in \mathcal{U}} \pi_u(\mathbf{y}_u), \tag{4}$$

since the user populations are independent.

Lemma 2.2 *Probabilities $\tilde{\pi}(\mathbf{x})$, $\mathbf{x} \in \tilde{\Omega}$, of states in a system with finite link capacities are obtained by truncation*

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

where $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\} = \sum_{\mathbf{x} \in \tilde{\Omega}} \pi(\mathbf{x})$.

Proof: By assumption, the leaf processes satisfy the detailed balance. Since the leaf link processes are independent, and the whole process is thus of product form (Eq. (4)), detailed balance holds for the network state space. Thus, truncation is allowed (see e.g. [11], [16, page 19]). ∎

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, we assume that the channels are ordered so that the channel for which we are calculating blocking has the greatest index I. Consider a 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.

$$\max\{d(r), d(y_{j,I})\} \le C_j - D'(\mathbf{y}_j) \tag{5}$$

We use the expression "link j blocks" if 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, and is defined as

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

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

$$B_{u,r} = \mathrm{P}\left\{\mathbf{X} \in \mathcal{B}_{u,r} \mid \mathbf{X} \in \tilde{\Omega}\right\} = \frac{\mathrm{P}\{\mathbf{X} \in \mathcal{B}_{u,r}\}}{\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}} \\ = 1 - \frac{\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\}}{\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}}. \tag{6}$$

Call blocking probabilities for users depend on the chosen user model, and will be discussed in section 5. Calculation of time blocking probabilities for layers is now easy, but very time consuming: the number of states in the state space is $(L+1)^{UI}$. Thus, efficient algorithms are needed. An algorithm utilizing the so-called MAX-convolution is presented in the next section.

3 MAX-convolution algorithm

As seen in Eq. (4), network state probabilities can be calculated from the leaf link state probabilities. That is, the state of any link j in the network is defined by the leaf links $\mathcal{U}_j$.

Definition 3.1 *The MAX-convolution operation $\circledast$ for any two functions $f, g : \mathcal{S} \to \mathbb{R}$ is defined as*

$$[f \circledast g](\mathbf{y}) = \sum_{\mathbf{y}',\mathbf{y}'':\ \max(\mathbf{y}',\mathbf{y}'')=\mathbf{y}} f(\mathbf{y}')g(\mathbf{y}''),$$

where $\max(\cdot,\cdot)$ denotes a componentwise MAX-operation.

Lemma 3.2 *The link probabilities $\pi_j(\mathbf{y})$ of any link j in the network can be calculated recursively as follows:*

$$\pi_j(\mathbf{y}) = \begin{cases} \pi_j(\mathbf{y}) & \text{if } j \in \mathcal{U} \\ [\pi_s \circledast \pi_t](\mathbf{y}) & \text{otherwise,} \end{cases}$$

where $\mathcal{N}_j = \{s, t\}$.

Proof: Consider three links in a branching point of the tree, such that $\mathcal{N}_j = \{s,t\}$. Since the processes in links s and t are independent,

$$\mathrm{P}\{\mathbf{Y}_j = \mathbf{y}\} = \\ \sum_{\mathbf{y}_s,\mathbf{y}_t:\max(\mathbf{y}_s,\mathbf{y}_t)=\mathbf{y}} \mathrm{P}\{\mathbf{Y}_s = \mathbf{y}_s\}\,\mathrm{P}\{\mathbf{Y}_t = \mathbf{y}_t\}.$$

This operation corresponds to the MAX-convolution operation. ■

3.1 Denominator

Definition 3.3 *Define for all $j \in \mathcal{J}$,*

$$Q_j^d(\mathbf{y}) = \mathrm{P}\{\mathbf{Y}_j = \mathbf{y}\ ;\ D(\mathbf{Y}_{j'}) \le C_{j'}, j' \in \mathcal{M}_j\}.$$

Definition 3.4 *The truncation operator T_j^d for any function $f : \mathcal{S} \to \mathbb{R}$ is defined as*

$$T_j^d f(\mathbf{y}) = f(\mathbf{y}) 1_{D(\mathbf{y}) \le C_j}.$$

Theorem 3.5 *The terms is the state sum $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}$ (Eq. (6))*

$$\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\} = \sum_{\mathbf{y}} Q_J^d(\mathbf{y}),$$

can be calculated recursively as

$$Q_j^d(\mathbf{y}) = \begin{cases} T_j^d \pi_j(\mathbf{y}) & \text{if } j \in \mathcal{U}, \\ T_j^d\left[Q_s^d \circledast Q_t^d\right](\mathbf{y}) & \text{otherwise,} \end{cases}$$

and $\mathcal{N}_j = \{s, t\}$.

Proof: Link probabilities $Q_J^d(\mathbf{y})$ are defined by pointwise disjoint sets of network states. These sets are all in the allowed state space $\tilde{\Omega}$, according to the definition of $Q_J^d(\mathbf{y})$. The union of all sets of network states defined by different $Q_J^d(\mathbf{y})$ is the allowed state space.

Lemma 3.2 states the recursive way to calculate link probabilities. The truncation constraints (Eq. (3)) are linkwise, and affect link probabilities in a similar way as the original state space probabilities. The truncation operations applied recursively guarantee that the link capacity constraints are taken into account for all links $j' \in \mathcal{M}_j$. ■

3.2 Numerator

The condition for a state in $\tilde{\Omega}$ to be in set $\tilde{\Omega} \setminus \mathcal{B}_{u,r}$ is given in Eq. (5).

Definition 3.6 *Define for $j \in \mathcal{R}_u$*

$$Q_{j,r}^n(\mathbf{y}) = \mathrm{P}\Big\{\mathbf{Y}_j = \mathbf{y}\ ;\ D(\mathbf{Y}_{j'}) \le C_{j'}, j' \in \mathcal{M}_j\ ; \\ d(r) \le C_{j'} - D'(\mathbf{Y}_{j'}), j' \in \mathcal{M}_j \cap \mathcal{R}_u\Big\}.$$

Definition 3.7 *The truncation operator $T_{j,r}^n$ for any function $f : \mathcal{S} \to \mathbb{R}$ is defined as*

$$T_{j,r}^n f(\mathbf{y}) = f(\mathbf{y}) 1_{\max\{d(r), d(y_I)\} \le C_j - D'(\mathbf{y})}.$$

Theorem 3.8 *The terms in the state sum $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\}$ (Eq. (6))*

$$\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\} = \sum_{\mathbf{y}} Q_{J,r}^n(\mathbf{y}),$$

can be calculated recursively as

$$Q_{j,r}^n(\mathbf{y}) = \begin{cases} T_{j,r}^n \pi_j(\mathbf{y}) & \text{when } j = u, \text{ and} \\ T_{j,r}^n\left[Q_{s,r}^n \circledast Q_t^d\right](\mathbf{y}) & \text{when } j \in \mathcal{R}_u \setminus \{u\}, \end{cases}$$

and $\mathcal{N}_j = \{s, t\}$, where $s \in \mathcal{R}_u$.

The only difference to Theorem 3.5 is a tighter condition for links on route $\mathcal{R}_u$. This is implemented by the new truncation operator, which guarantees simultaneously that the state is in the allowed state space, and that it is a non-blocking state.

3.3 Algorithm

To calculate the time blocking probability $B_{u,r}$, state sums $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\}$ and $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}$ are needed. These are calculated as in the previous subsections. The algorithm is summarised as follows:

1. Set $j := 1$.
2. Calculate $Q_j^d(\mathbf{y})$ for all $\mathbf{y}$ from Theorem 3.5.
3. If $j \in \mathcal{R}_u$, calculate $Q_{j,r}^n(\mathbf{y})$ for all $\mathbf{y}$ from Theorem 3.8.
4. Set $j := j + 1$. If $j \leq J$, jump to step 2.
5. Sum $Q_J^d(\mathbf{y})$ over $\mathbf{y}$ and divide it with the sum of $Q_{J,r}^n(\mathbf{y})$ to get $1 - B_{u,r}$.

Note here that our link numbering scheme ($\forall j' \in \mathcal{N}_j; j' < j$) guarantees that link probabilities for all links $j' \in \mathcal{N}_j$ have already been calculated when calculation of the link probability of link j starts.

The complexity of the MAX-convolution algorithm is $O((U-1)(L+1)^{2I})$. For each MAX-convolution operation, two leaf links are removed, leaving one leaf link. The process stops when there is only one leaf link remaining (link J). That is, the number of MAX-convolution operations equals the reduction of the number of leaf links, $U-1$.

Each convolution step requires operating each value of the I-element vector $\mathbf{y}$, where each element has $L+1$ possible values, yielding $(L+1)^I$ possible values. There are two such vectors involved, thus the number of steps is $(L+1)^{2I}$.

The complexity of the algorithm is seen to be linear with respect to the number of leaf links U. This is a significant improvement compared to the exponential complexity of the original problem.

3.4 Background traffic

So far we have treated a single multicast tree in isolation. When there are also unicast traffic and other multicast trees in the network, we need to use approximations to get efficient algorithms. The approximation used here groups all traffic except the interesting multicast tree as independent background traffic. The approach is similar to the one in [14], but note that other multicast trees may also occupy the same links.

To take the background traffic into account, we first calculate the link occupancy distribution $q_j(z)$ in the link in the case there is *only* background traffic. Then, the only remaining step is to redefine the truncation operators of the MAX-convolution algorithm to take background traffic into account:

Definition 3.9 *Define truncation operator $\widehat{T}_j^d$ for any function $f : \mathcal{S} \to \mathbb{R}$ as*

$$\widehat{T}_j^d f(\mathbf{y}) = \left(\sum_{z=0}^{C_j - D(\mathbf{y})} q_j(z) \right) f(\mathbf{y}).$$

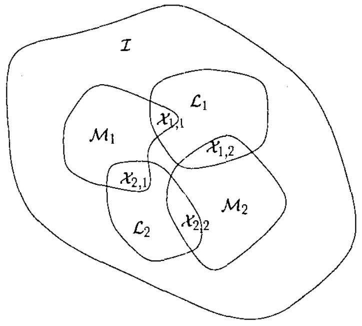

Figure 3: Combining two links when $L = 2$. All channels in $\mathcal{I}$ are equally probable. Link s has $|\mathcal{L}_1| = l_1$ channels on layer 1 and $|\mathcal{L}_2| = l_2$ channels on layer 2. Sets $\mathcal{L}_1$ and $\mathcal{L}_2$ are disjoint. Similarly, link t has m_1 and m_2 channels on layers 1 and 2, respectively.

Definition 3.10 *Define truncation operator $\widehat{T}_{j,r}^n$ for any function $f : \mathcal{S} \to \mathbb{R}$ as*

$$\widehat{T}_{j,r}^n f(\mathbf{y}) = \left(\sum_{z=0}^{C_j - D'(\mathbf{y}) - \max\{d(r), d(y_I)\}} q_j(z) \right) f(\mathbf{y}).$$

4 Combinatorial algorithm

It may happen that all channels in the network are statistically indistinguishable (as explained in section 1), to a sufficient accuracy. Thus, to describe the state of a link, it is not necessary to express the state of every individual channel i separately (as in the previous section), but just to tell the number of channels on each layer l. Then, the link state can be defined as $\mathbf{K}_j = (K_{j,l}; l \in \mathcal{L})$, where $K_{j,l}$ refers to the number of channels on layer l in link j. This results in a considerable reduction in the dimensionality of the state space, since typically $L \ll I$. In this section, we develop an algorithm with polynomial complexity for this special case.

First, assume that link j has two downstream neighbour links, $\mathcal{N}_j = \{s, t\}$. The set of channels that are on layer l on link s is denoted by $\mathcal{L}_l \subset \mathcal{I}$. These sets are disjoint, i.e. for all $l \neq l'$, $\mathcal{L}_l \cap \mathcal{L}_{l'} = \emptyset$. The set of channels that are on layer l on link t is denoted by $\mathcal{M}_l \subset \mathcal{I}$. These sets are also disjoint; for all $l \neq l'$, $\mathcal{M}_l \cap \mathcal{M}_{l'} = \emptyset$. We denote the intersections of these sets as $\mathcal{L}_1 \cap \mathcal{M}_1 = \mathcal{X}_{1,1}$, $\mathcal{L}_1 \cap \mathcal{M}_2 = \mathcal{X}_{1,2}$, $\mathcal{L}_2 \cap \mathcal{M}_1 = \mathcal{X}_{2,1}$, $\mathcal{L}_2 \cap \mathcal{M}_2 = \mathcal{Y}_{2,2}$, and so on. For a two layer example, see Figure 3. The set of channels that are on layer l on link j is denoted by $\mathcal{K}_l$. Let $l_1 = |\mathcal{L}_1|$, and correspondingly from l_2 to l_L, from m_1 to m_L, and from k_1 to k_L. Let $x_{a,b} = |\mathcal{X}_{a,b}|$ for all $a, b \in \mathcal{L}$. Let $\mathbf{l} = (l_1, \ldots, l_L)$, $\mathbf{m} = (m_1, \ldots, m_L)$, $\mathbf{k} = (k_1, \ldots, k_L)$.

First consider the case $L = 1$. Now, $x = l + m - k$, and $\mathrm{P}\{K_j = k \mid K_s = l, K_t = m\}$ is easily calculated from the following lemma.

Lemma 4.1 *Given three links j, s, and t, such that $\mathcal{N}_j = \{s,t\}$, carrying multicast traffic with a single layer and I statistically indistinguishable channels,*

$$\mathrm{P}\{K_j = k \,|\, K_s = l, K_t = m\} = \frac{\binom{l}{x}\binom{I-l}{m-x}}{\binom{I}{m}}.$$

The proof of this lemma can be found in [2, 1].

For the general case, we first give our notation for the multinomial coefficient:

$$\begin{pmatrix} & a & \\ b_1 & \cdots & b_n \end{pmatrix} = \frac{a!}{(a - \sum_{n'=1}^{n} b_{n'})! \prod_{n'=1}^{n} (b_{n'}!)}.$$

In the general case, there are L^2 possible intersections for the sets. Of these, however, L are defined by the others. We choose the intersections $x_{a,a}$ as such:

$$x_{a,a} = l_a + m_a - k_a - \sum_{a'=a+1}^{L} (x_{a,a'} + x_{a',a}).$$

The sizes of the rest of the intersections are then free parameters. From now on, we will use the short notation of style $\mathrm{P}\{\mathbf{k}\}$ for probabilities $\mathrm{P}\{\mathbf{K} = \mathbf{k}\}$. With these definitions, the probability $\mathrm{P}\{\mathbf{K}_j = \mathbf{k} \,|\, \mathbf{K}_s = \mathbf{l}, \mathbf{K}_t = \mathbf{m}\} = \mathrm{P}\{\mathbf{k} \,|\, \mathbf{l}, \mathbf{m}\}$ is easily calculated from the following lemma.

Lemma 4.2 *Given three links j, s, and t, such that $\mathcal{N}_j = \{s,t\}$, carrying multicast traffic with L layers and I statistically indistinguishable channels,*

$$\begin{aligned} &\mathrm{P}\{\mathbf{k} \,|\, \mathbf{l}, \mathbf{m}\} = \sigma(\mathbf{k}, \mathbf{l}, \mathbf{m}, I) \\ =&\sum_{x_{1,2}} \cdots \sum_{x_{1,L}} \sum_{x_{2,1}} \sum_{x_{2,3}} \cdots \sum_{x_{2,L}} \cdots \sum_{x_{L,1}} \cdots \sum_{x_{L,L-1}} \\ &\frac{\begin{pmatrix} & I - \sum_{a\geq 1} l_a & \\ m_1 - \sum_{a \geq 1} x_{a,1} & \cdots & m_L - \sum_{a\geq 1} x_{a,L} \end{pmatrix}}{\begin{pmatrix} & I & \\ m_1 & \cdots & m_L \end{pmatrix}} \prod_{a=1}^{L} \begin{pmatrix} & l_a & \\ x_{a,1} & \cdots & x_{a,L} \end{pmatrix}. \end{aligned}$$

This lemma can be proven easily by induction, but because of space restrictions, we have omitted the proof. For the two layer case, see [8].

Definition 4.3 *We define the binary operation $\otimes$ for all functions $f, g : \mathbb{Z}_+^L \to \mathbb{R}$ as*

$$[f \otimes g](\mathbf{k}) = \sum_{\mathbf{l}} \sum_{\mathbf{m}} \sigma(\mathbf{k}, \mathbf{l}, \mathbf{m}, I) f(\mathbf{l}) g(\mathbf{m}).$$

Let $\pi_j(\mathbf{k}) = \mathrm{P}\{\mathbf{K}_j = \mathbf{k}\} = \mathrm{P}\{\mathbf{k}\}$ denote the link state probability in link j corresponding to the state vector $\mathbf{k}$.

Lemma 4.4 *The link probabilities $\pi_j(\mathbf{k})$ of any link j in network can be calculated recursively as follows:*

$$\pi_j(\mathbf{k}) = \begin{cases} \pi_j(\mathbf{k}) & \text{if } j \in \mathcal{U} \\ [\pi_s \otimes \pi_t](\mathbf{k}) & \text{otherwise,} \end{cases}$$

where $\mathcal{N}_j' = \{s,t\}$.

Proof: Consider then three links, j, s, and t, such that $\mathcal{N}_j = \{s,t\}$.

$$\mathrm{P}\{\mathbf{k}\} = \sum_{\mathbf{l}} \sum_{\mathbf{m}} \mathrm{P}\{\mathbf{k} \,|\, \mathbf{l}, \mathbf{m}\} \mathrm{P}\{\mathbf{l}\} \mathrm{P}\{\mathbf{m}\}.$$

Lemma 4.2 gives the probability $\mathrm{P}\{\mathbf{k} \,|\, \mathbf{l}, \mathbf{m}\}$. Combining this with Definition 4.3 yields one step in the recursion. ■

4.1 Denominator

For this case, we define

$$D(\mathbf{k}) = \sum_{l=1}^{L} k_l d(l).$$

Definition 4.5 *Define*

$$Q_j(\mathbf{k}) = \mathrm{P}\{\mathbf{K}_j = \mathbf{k} \,;\, D(\mathbf{K}_{j'}) \leq C_{j'}, j' \in \mathcal{M}_j\}.$$

Definition 4.6 *We define the truncation operator T_j for any function $f : \mathbb{Z}_+^L \to \mathbb{R}$ as*

$$T_j f(\mathbf{k}) = f(\mathbf{k}) 1_{D(\mathbf{k}) \leq C_j}.$$

Theorem 4.7 *The terms in the state sum $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}$ (Eq. (6))*

$$\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\} = \sum_{\mathbf{y}} Q_J(\mathbf{k}),$$

can be calculated recursively as

$$Q_j(\mathbf{k}) = \begin{cases} T_j \pi_j(\mathbf{k}) & \text{if } j \in \mathcal{U} \\ T_j [Q_s \otimes Q_t](\mathbf{k}) & \text{otherwise,} \end{cases}$$

and $\mathcal{N}_j = \{s,t\}$.

Proof: Lemma 4.4 states the recursive way to calculate link probabilities. The truncation constraints (Eq. (3)) are linkwise and affect link probabilities in a similar way as the original state space probabilities. The truncation operations applied recursively guarantee that the link capacity constraints are taken into account for all links $j' \in \mathcal{M}_j$. ■

4.2 Numerator

We can calculate the numerator in the same way as the denominator. The state of the channel the request is for, however, affects the truncation condition for non-blocking states, as seen in Eq. (5). Thus, we need to consider the interesting channel I separately for links on the route $\mathcal{R}_u$. In this subsection, we use prime ($'$) for all variables which are calculated using the first $I-1$ channels, i.e. excluding the channel for which we are calculating blocking.

Let $\mathbf{K}_j'$ denote random variables in a state space where there are $I-1$ channels, corresponding to the variables $\mathbf{K}_j$ in the complete state space. L_j denotes the random variable representing the state of channel I in link j, i.e. $L_j = Y_{j,I}$.

Since the link probabilities $Q_j(\mathbf{k})$ have been defined for I channels, we define an operator E to generate link probabilities $Q_j'(\mathbf{k}', l)$, where state l of channel I is explicit.

Definition 4.8 *Define operator E for any function $f : \mathbb{Z}_+^L \to \mathbb{R}$ as*

$$Ef(\mathbf{k}',l) = \begin{cases} \left(1 - \frac{\mathbf{k}'\cdot\mathbf{1}}{I}\right) f(\mathbf{k}'), & \text{if } l = 0 \\ \left(\frac{k_l'+1}{I}\right) f(\mathbf{k}' + \mathbf{e}_l), & \text{if } l > 0 \end{cases}$$

for all $\mathbf{k}' \cdot \mathbf{1} < I$. For $\mathbf{k}' \cdot \mathbf{1} \geq I$; $Ef(\mathbf{k}',l) = 0$. Here, $\mathbf{e}_l$ denotes a vector whose l^{th} element is 1, and the others are 0.

Denote $\pi_j'(\mathbf{k}',l) = \mathrm{P}\{\mathbf{K}_j' = \mathbf{k}', L_j = l\}$.

Lemma 4.9 *Given link probability distribution $\pi_j(\mathbf{k})$, it holds that*

$$\pi_j'(\mathbf{k}',l) = E\pi_j(\mathbf{k}',l).$$

Proof: First, the probabilities $\mathrm{P}\{\mathbf{k}',l\}$ are as follows:

$$\mathrm{P}\{\mathbf{k}',l\} = \sum_{\mathbf{k}} \mathrm{P}\{\mathbf{k}',l \,|\, \mathbf{k}\}\,\mathrm{P}\{\mathbf{k}\}.$$

The conditional probability $\mathrm{P}\{\mathbf{k}',l \,|\, \mathbf{k}\}$ is then

$$\mathrm{P}\{\mathbf{k}',l \,|\, \mathbf{k}\} = \begin{cases} 1 - \frac{\mathbf{1}\cdot\mathbf{k}}{I}, & \text{if } \mathbf{k} = \mathbf{k}' \text{ and } l = 0, \\ \frac{k_l}{I}, & \text{if } \mathbf{k} = \mathbf{k}' + \mathbf{e}_l, \text{ and} \\ 0, & \text{otherwise.} \end{cases}$$

Finally, for each $(\mathbf{k}',l)$, there is one and only one $\mathbf{k}$ for which $\mathrm{P}\{\mathbf{k}',l \,|\, \mathbf{k}\} > 0$. ■

Definition 4.10 *We define $Q_{j,r}'(\mathbf{k}',l)$ for $j \in \mathcal{R}_u$ as*

$$\begin{aligned} Q_{j,r}'(\mathbf{k}',l) = \mathrm{P}\{&\mathbf{K}_j' = \mathbf{k}, L_j = l\,; \\ &D(\mathbf{K}_{j'}') + d(r) \leq C_{j'}, j' \in \mathcal{M}_j \cap \mathcal{R}_u\,; \\ &D(\mathbf{K}_{j'}) \leq C_{j'}, j' \in \mathcal{M}_j\}. \end{aligned}$$

Definition 4.11 *We define the operation $\odot$ for functions $f : \mathbb{Z}_+^L \times \mathbb{Z}_+ \to \mathbb{R}$, and $g : \mathbb{Z}_+^L \to \mathbb{R}$ as*

$$\begin{aligned} [f \odot g](\mathbf{k}',l) = &\sum_{\mathbf{l}'}\sum_{\mathbf{m}'} \sigma(\mathbf{k}',\mathbf{l}',\mathbf{m}',I-1) \\ \times &\sum_{(v_1,v_2) s.t. \max(v_1,v_2)=l} f(\mathbf{l}',v_1)Eg(\mathbf{m}',v_2). \end{aligned}$$

Definition 4.12 *The truncation operator $T_{j,r}^\circ$ for functions $f : \mathbb{Z}_+^L \times \mathbb{Z}_+ \to \mathbb{R}$ is defined as follows:*

$$T_{j,r}^\circ f(\mathbf{k}',l) = f(\mathbf{k}',l)1_{D(\mathbf{k}')+\max\{d(l),d(r)\}\leq C_j}.$$

Theorem 4.13 *The terms in the state sum $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\backslash\mathcal{B}_{u,r}\}$ (Eq. 6)*

$$\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\} = \sum_{\mathbf{k}}\sum_{l} Q_{J,r}'(\mathbf{k},l),$$

can be calculated recursively as

$$Q_{j,r}'(\mathbf{k}',l) = \begin{cases} T_{u,r}^\circ E\pi_u(\mathbf{k}',l) & \text{if } j = u \\ T_{j,r}^\circ[Q_{s,r}' \odot Q_t](\mathbf{k}',l) & \text{if } j \in \mathcal{R}_u \setminus \{u\}, \end{cases}$$

and $\mathcal{N}_j = \{s,t\}$, where $s \in \mathcal{R}_u$.

Proof: For the user leaf link u, we may apply state-dependent truncation simply as

$$Q_{u,r}'(\mathbf{k}',l) = T_{u,r}^\circ E\pi_u(\mathbf{k}',l),$$

since operator E makes dependence on channel I state l explicit.

The operation $\odot$ combines the link probabilities Q_j' with the corresponding probabilities for links not on the route, taking into account the condition for non-blocking states on the route.

Otherwise, the composition is similar to Theorem 4.7. ■

4.3 Algorithm

To calculate blocking probability $B_{u,r}$, state sums $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega} \setminus \mathcal{B}_{u,r}\}$ and $\mathrm{P}\{\mathbf{X} \in \tilde{\Omega}\}$ are needed. These are calculated as in the previous subsections. The algorithm is summarised as follows:

1. Set $j := 1$.
2. Calculate $Q_j(\mathbf{k})$ for all $\mathbf{k}$ from Theorem 4.7.
3. If $j \in \mathcal{R}_u$, calculate $Q_{j,r}'(\mathbf{k}',l)$ for all $\mathbf{k}'$ and l from Theorem 4.13.
4. Set $j := j + 1$. If $j \leq J$, jump to step 2.
5. Sum $Q_J(\mathbf{k})$ over $\mathbf{k}$ and divide it with the sum of $Q_{J,r}'(\mathbf{k}',l)$ to get $1 - B_{u,r}$.

The complexity of the combinatorial algorithm is $O((U-1)(I+1)^{L(L+2)})$. The number of links affects as for the MAX-convolution. For each combinatorial convolution, two links are combined, each having L layers. The amount of intersections is L^2, of which L intersections are defined by the other parameters. Thus, when calculating the $\sigma(\mathbf{k},\mathbf{l},\mathbf{m},I)$, there are $L^2 - L$ loops. In each convolution, there are additional $2L$ loops for each value of $\mathbf{k}$. Calculating the convolution operation for all values of $\mathbf{k}$ results in L more loops. Thus, there is a total of $L + 2L + (L^2 - L)$ sets over which summations are done. Each set has at most $I + 1$ values. Thus the complexity of the convolution step is $(I+1)^{L(L+2)}$.

A computational note: for all $l \leq r$, the link probabilities $Q'(\mathbf{k}',l)$ can be summed and treated as a single state probability, since the truncation affects them in a similar fashion.

4.4 Background traffic

Indenpendent background traffic can be treated as for the MAX-convolution algorithm, by re-defining truncation operators. Recall the link occupancy distribution $q_j(z)$ due to background traffic. Now, the truncation operators must be redefined to take background traffic into account:

Definition 4.14 *Define truncation operator $\widehat{T}_j$ for any function $f : \mathbb{Z}_+^L \to \mathbb{R}$ as*

$$\widehat{T}_j^d f(\mathbf{k}) = \left(\sum_{z=0}^{C_j - D(\mathbf{k})} q_j(z)\right) f(\mathbf{k}).$$

Definition 4.15 *Define truncation operator $\widehat{T}^{\circ}_{j,r}$ for any function $f : \mathbb{Z}_+^L \times \mathbb{Z}_+ \to \mathbb{R}$ as*

$$\widehat{T}^{n}_{j,r} f(\mathbf{k}', l) = \left(\sum_{z=0}^{C_j - D(\mathbf{k}') - \max\{d(l), d(r)\}} q_j(z) \right) f(\mathbf{k}', l).$$

4.5 Multiple groups of channels

The combinatorial convolution algorithm can be generalised to handle multiple groups of statistically indistinguishable channels. This generalisation can be done as was done in the single layer case in [1].

5 User population models

In this section, we introduce several user population models, in a similar way as Nyberg et al. [13]. The "interface" between the user population models and the presented algorithms consists of two components. First, the user populations are assumed to generate a time reversible Markov process. Second, the parameter given to the algorithms is either $\pi_u(\mathbf{y})$ for the MAX-convolution algorithm, or $\pi_u(\mathbf{k})$ for the combinatorial convolution algorithm.

For the algorithms to work, we required detailed balance to hold (Lemma 2.2). In the following subsections, we present user population models that satisfy the detailed balance condition, and give expressions for the corresponding $\pi_u(\cdot)$ probabilities.

5.1 Insensitivity

We study the insensitivity of the system using the results given in [17]. The main idea is to associate each state with clocks $s \in \mathcal{T}$, of which a part is running. When a clock runs out, a state transition is made and the clock is given a new value, taken from a distribution φ_s, if the clock is active in the new state. If the steady state distribution of the system is the same, independently of the distribution φ_s (as long as the mean remains the same), the system is said to be insensitive with respect to clock s. This happens when the states are locally balanced with respect to clock s.

The results on insensitivity are given for each user model in the following subsections.

5.2 Infinite user population

In an infinite user population model, calls to channels are generated as a Poisson arrival stream with arrival intensity λ_u, and a random sampling is done to deal the calls to different channels and layers. Call holding times (for all channels and layers) are assumed to be exponentially distributed with mean $1/\mu$. Channels and layers are selected with probabilities $\alpha_{u,i,l}$ ($\sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}} \alpha_{u,i,l} = 1$). Blocked calls are lost. The resulting traffic process is easily found to be a reversible Markov process. Since states of channels i on any leaf link u are independent,

$$\pi_u(\mathbf{y}) = \prod_{i \in \mathcal{I}} p_{u,i,y_i},$$

where $p_{u,i,y_i} = \mathrm{P}\{Y_{u,i} = y_i\}$ is the probability that channel i is on layer y_i on leaf link u.

The probabilities $p_{u,i,l}$ are found by examining the probability of state 0 of an $M/M/\infty$-queue (cf. [9]):

$$p_{u,i,0} = 1 - \sum_{l=1}^{L} p_{u,i,l},$$

$$p_{u,i,L} = 1 - e^{-\frac{\alpha_{u,i,L}\lambda_u}{\mu}},$$

and

$$p_{u,i,l} = \left(1 - e^{-\frac{\alpha_{u,i,l}\lambda_u}{\mu}}\right) \prod_{m=l+1}^{L} e^{-\frac{\alpha_{u,i,m}\lambda_u}{\mu}}.$$

In the case of statistically indistinguishable channels, the equations for $\tilde{p}_{u,l} = p_{u,i,l}$ for all i are similar, except that we need to substitute $\alpha_{u,l}/I$ for $\alpha_{u,i,l}$, where $\sum_{l \in \mathcal{L}} \alpha_{u,l} = 1$. Thus,

$$\pi_u(\mathbf{k}) = \binom{I}{k_1 \; \cdots \; k_L} \tilde{p}_{u,0}^{\,I - \sum_{l \in \mathcal{L}} k_l} \prod_{l \in \mathcal{L}} \tilde{p}_{u,l}^{k_l}.$$

Due to the PASTA property of traffic generated by the infinite user population, the call blocking probability equals the time blocking probability [18].

To study the insensitivity of the infinite user population model, the clocks associated to each state need to be defined. In order to do so, we extend all the state diagrams of the leaf links. In the extended state diagram of leaf link u, there are separate states corresponding to different numbers of actual users on each channel i and layer l. This results in a countably infinite number of states. If the call arrival and connection holding times are exponentially distributed, the resulting process is a reversible Markov chain.

Let $\mathcal{E}$ denote the extended state space. Now, for state vectors $\mathbf{n} \in \mathcal{E}$, and $\mathbf{n} + \mathbf{e}_{u,i,l} \in \mathcal{E}$, and state probabilities $\tilde{\pi}(\cdot)$, a detailed balance equation holds:

$$\tilde{\pi}(\mathbf{n})\lambda_u \alpha_{u,i,l} = \tilde{\pi}(\mathbf{n} + \mathbf{e}_{u,i,l})(n_{u,i,l} + 1)\mu.$$

For general connection holding times, we generate a scheme with relabelling for this associated Markov chain (see [17, Theorem 3.1]). We associate each channel-layer pair with a clock $s_{u,i,l}$ that generates calls to that channel-layer pair. These clocks are active in all states of the system. In addition, each channel-layer pair is associated with clocks that represent the actual users on a specific channel-layer pair. For the insensitivity property to hold, the steady-state intensity of transitions due to birth of $s_{u,i,l,m}$ and due to death of $s_{u,i,l,m}$ must equal for all $m = 1, \ldots, n_{u,i,l}$. This can be seen to hold from the detailed balance equation of the extended state space. Thus, the system is insensitive to the call holding times.

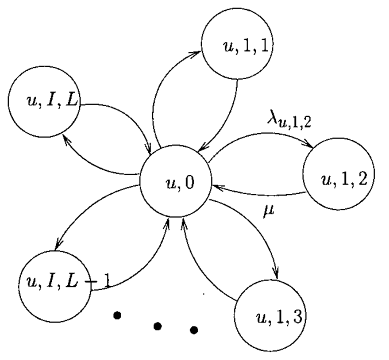

Figure 4: State transition diagram for user u (the single user model)

5.3 Single user model

Consider the case where the user population consists of a single user, selecting each channel and layer with intensity $\lambda_{u,i,l}$ and holding them with intensity μ. Both user idle and call holding times are assumed to be exponentially distributed. The resulting Markov process is reversible. Figure 4 shows the state transition diagram of the process.

The probability that the user is on channel i and layer l is

$$p_{u,i,l} = \frac{\lambda_{u,i,l}/\mu}{1 + \sum_{i' \in \mathcal{I}} \sum_{l' \in \mathcal{L}} \lambda_{u,i',l'}/\mu},$$

and the probability that the user is idle is

$$p_{u,0} = \frac{1}{1 + \sum_{i' \in \mathcal{I}} \sum_{l' \in \mathcal{L}} \lambda_{u,i',l'}/\mu}.$$

The probabilities $\pi_u(\mathbf{y})$ can be calculated from these probabilities as

$$\pi_u(\mathbf{y}) = \begin{cases} p_{u,0} & \text{when } \mathbf{y} = \mathbf{0}, \\ p_{u,i,l} & \text{when } \mathbf{y} = l\mathbf{e}_i, \\ 0 & \text{otherwise.} \end{cases}$$

For the case of statistically indistinguishable channels, use $\lambda_{u,l}/I$ instead of $\lambda_{u,i,l}$. The probabilities $\pi_u(\mathbf{k})$ are then

$$\pi_u(\mathbf{k}) = \begin{cases} p_{u,0} & \text{when } \mathbf{y} = \mathbf{0}, \\ \sum_{i \in \mathcal{I}} p_{u,i,l} & \text{when } \mathbf{k} = \mathbf{e}_l, \\ 0 & \text{otherwise.} \end{cases}$$

With the single user model, call blocking equals time blocking in a network without this user. Thus, for calculating call blocking for a specific user, it is just necessary to use for this user $p_{u,0} = 1$ and $p_{u,i,l} = 0$ for $l > 0$ and for all i.

Each user u is associated with a set of clocks: clock $s_{u,0}$ for the state where the user is not on any channel or layer, and clocks $s_{u,i,l}$ for each channel-layer pair. In each state, only one of these clocks can be on. If the user is on a specific channel i and layer l, $s_{u,i,l}$ is on. If the user is not on any channel, clock $s_{u,0}$ is on. The resulting state space is locally balanced w.r.t. all clocks $s \in \mathcal{T}$. Thus, the system is insensitive to both call holding times and user idle times.

5.4 Finite user population of independent users

We can generate finite user populations with independent users using the single user model presented. Just generate a tree with supplemented links of infinite capacity, where each leaf node contains a single user, and the root link of the tree is the actual leaf link of the original problem.

Consider, for example, a user population consisting of a finite number of independent layer-specific users. Each user is defined as a single user, as in the previous section, who only visits the states of a specific layer and the idle state.

Call blocking with finite user populations is calculated by first removing the user we are calculating blocking for (by setting $p_{u,0} = 1$), and then calculating the time blocking probability of the network.

The system with a finite user population has the insensitivity properties of the actual users used to construct the user population.

6 Summary

We presented two new algorithms for calculating call 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.

Our first algorithm, the so called MAX-convolution algorithm has linear complexity with respect to the network size (number of user populations). We also devised an algorithm for the case where all channels are statistically indistinguishable, yielding an algorithm with a computational complexity which is polynomial with respect to the number of channels and linear with respect to the number of user populations (network size). Due to this linear dependence on the number of links, network size does not prohibit usage of the algorithms.

We presented the algorithms for the cases where multicast traffic is in isolation, as well as an approximation for mixtures of unicast and multicast traffic in real networks.

An interesting subject for future work is an extensive study on the accuracy of the approximation in the case of background traffic.

References

[1] S. Aalto, J. Karvo, and J. Virtamo. Calculating blocking probabilities in multicast loss systems. *Submitted for publication.*

[2] S. Aalto and J. Virtamo. Combinatorial algorithm for calculating blocking probabilities in multicast networks. In *Proc. 15th Nordic Teletraffic Seminar, NTS-15*, pages 23–34, Lund, Sweden, Aug. 2000.

[3] K. Boussetta and A.-L. Belyot. Multirate resource sharing for unicast and multicast connections. In D. H. K. Tsang and P. J. Kühn, editors, *Proceedings of Broadband Communications'99*, pages 561–570, Hong Kong, Nov. 1999.

[4] W. C. Chan and E. Geraniotis. Tradeoff between blocking and dropping in multicasting networks. In *ICC '96 Conference Record*, volume 2, pages 1030–1034, June 1996.

[5] G. L. Choudhury, K. K. Leung, and W. Whitt. An algorithm to compute blocking probabilities in multirate multi-class multi-resource loss models. *Adv. Appl. Prob.*, 27:1104–1143, Dec. 1995.

[6] R. Fortet and C. Grandjean. Congestion in a loss system when some calls want several devices simultaneously. *Electrical Communication*, 39(4):513–526, 1964.

[7] G. Karlsson and M. Vetterli. Packet video and its integration into the network architecture. *IEEE Journal on Selected Areas in Communications*, 7(5):739–751, June 1989.

[8] J. Karvo, S. Aalto, and J. Virtamo. Blocking probabilities of two-layer statistically indistinguishable multicast streams. In J. M. de Souza, N. L. S. Fonseca, and E. A. de Souza e Silva, editors, *Proc. International Teletraffic Congress ITC-17*, pages 769–779, Salvador da Bahia, Brazil, Sept. 2001.

[9] J. Karvo, J. Virtamo, S. Aalto, and O. Martikainen. Blocking of dynamic multicast connections in a single link. In *Proc. of International Broadband Communications Conference, Future of Telecommunications*, pages 473–483, Stuttgart, Germany, Apr. 1998.

[10] J. Karvo, J. Virtamo, S. Aalto, and O. Martikainen. Blocking of dynamic multicast connections. *Telecommunication Systems*, 16(3,4):467–481, 2001.

[11] F. P. Kelly. *Reversibility and Stochastic Networks*. John Wiley & Sons, 1979.

[12] P. Lassila, J. Karvo, and J. Virtamo. Efficient importance sampling for Monte Carlo simulation of multicast networks. In *Proc. INFOCOM'01*, pages 432–439, Anchorage, Alaska, Apr. 2001.

[13] E. Nyberg, J. Virtamo, and S. Aalto. An exact end-to-end blocking probability algorithm for multicast networks. *Submitted for publication*.

[14] E. Nyberg, J. Virtamo, and S. Aalto. An exact algorithm for calculating blocking probabilities in multicast networks. In G. Pujolle, H. Perros, S. Fdida, U. Körner, and I. Stavrakakis, editors, *Networking 2000*, pages 275–286, Paris, May 2000.

[15] K. Ramanan, A. Sengupta, I. Ziedins, and P. Mitra. Markov random field models of multicasting in tree networks. *Submitted for publication*.

[16] K. W. Ross. *Multiservice Loss Models for Broadband Telecommunication Networks*. Springer Verlag, London, 1995.

[17] R. Schassberger. Two remarks on insensitive stochastic models. *Adv. Appl. Prob.*, 18:791–814, 1986.

[18] R. W. Wolff. Poisson arrivals see time averages. *Operations Research*, 30(2):223–231, Mar.–Apr. 1982.