Bounds of the Overflow Priority Classification for Blocking Probability Approximation in OBS Networks

Shuo Li, Meiqian Wang, Eric W. M. Wong, Vyacheslav Abramov, and Moshe Zukerman

Abstract—It has been demonstrated that the overflow priority classification approximation (OPCA) is an accurate method for blocking probability evaluation for various networks and systems, including optical burst switched networks with deflection routing. OPCA is a hierarchical algorithm that requires fixed-point iterations in each layer of its hierarchy. This may imply a long running time. We prove here that the OPCA iterations alternately produce upper and lower bounds that consistently become closer to each other as more fixed-point iterations in each layer are used, and we demonstrate numerically that only a small number of iterations per layer are required for the bounds to be sufficiently close to each other. This behavior is demonstrated for various system parameters including offered load, number of channels per trunk, and maximum allowable number of deflections.

Index Terms—Deflection routing; Loss networks; Lower and upper bounds; Nonhierarchical networks; Optical burst switching (OBS); Optical packet switching (OPS); Overflow priority classification approximation; Performance analysis.

I. INTRODUCTION

Optical burst switching (OBS) [1–7] is an optical networking technique where packets are aggregated into data bursts at the network edge and each burst is transmitted as one unit. It aims to achieve faster connection time than optical circuit switching [8–11] and to avoid energy consuming processing of individual packets and excessive overhead due to guard-band provision between packets, as in optical packet switching [6,12–14].

An important consideration in OBS networks is burst contention that may lead to burst dumping and consequently loss of data [15–18]. Given that buffering data in the optical domain is difficult, especially for large bursts, one of the key contention resolution options is deflection routing [18]. As such, OBS deflection routing and its performance analysis has attracted significant attention [19–32]. However, most of the existing performance studies were either based on simulations or were limited to a single OBS node. In [32], the blocking probability of an OBS network with deflection was evaluated using the Erlang fixed-point approximation (EFPA) [33]. Then, in [34], a recently developed overflow priority classification approximation (OPCA) [35] was used in combination with the EFPA to obtain a blocking probability approximation for this problem that, as demonstrated there, is significantly more accurate than merely the approximation using the EFPA. Given the important role of the OPCA in accurately evaluating blocking probability of telecommunications networks and systems as demonstrated in [34] for OBS networks, and in [35] and [36] for other networks, the main focus of this paper is not on the accuracy and performance of the OPCA; instead, it provides new information about the properties of the OPCA algorithm applied to bufferless OBS networks based on just-enough-time (JET) signaling [37]. This new information has the potential for wider applicability, as it can be used for further development of the OPCA in other applications.

A weakness of both the EFPA and the original OPCA applied to OBS networks is that they require a fixed-point solution, which may require a large number of iterations. Because of the fixed-point iterations, analytical results for the complexities of both the EFPA and OPCA are unattainable. Moreover, numerical studies presented in this paper indicate that the OPCA consumes less time than the EFPA, and that the advantage of the OPCA increases with the capacity of the network (see Subsection V.B). For example, the EFPA requires 3006 s, compared to 397 s for the OPCA, to evaluate blocking probability for a National Science Foundation Network (NSFNET) topology with 10,000 channels per trunk (such a number is not unreasonable [38–40], especially if subwavelength channels are considered [41]). The advantage of the OPCA in running time is probably due to the fact that the OPCA is based on a hierarchical structure with a finite number of layers, where at each layer a separate set of fixed-point iterations are performed. Experience shows that this divide and rule approach tends to reduce the total number of iterations. Although accuracy is not the main topic of this paper, we do provide some new (to our knowledge) numerical results that complement the running time comparison and demonstrate that for the cases of OBS/JET considered, within a practical traffic loading range, i.e., so that an acceptable
We discover in this paper a new important property of the OPCA algorithm. In particular, we show that we can find upper and lower bounds for the blocking probability evaluated by the OPCA such that they draw nearer to each other with an increasing number of iterations. Specifically, in each iteration, the distance between them is never larger than in the previous iteration. The effects of the design parameters (such as maximum allowable number of deflections and number of channels per trunk) on the behavior of the OPCA blocking probability bounds are also discussed. It is important to clarify that the bounds discussed in this paper are always the bounds of the OPCA result and not the bounds of the exact blocking probability result.

The remainder of the paper is structured as follows. The description of the model is given in Section II. In Section III we recall basics of the OPCA method, and in Section IV we provide bounds of the OPCA results. To support the results of Section IV, in Section V numerical results for the OPCA bounds for the 13-node NSFNET are provided, as well as the effects of the number of channels per trunk and the maximum allowable number of deflections. In Section VI, we conclude the paper.

II. MODEL

We consider an OBS network described by a graph \( G(N, E) \), where \( N \) is a set of \( n \) nodes and \( E \) is the set of \( e \) arcs. The nodes are designated 1, 2, ..., \( n \), each of which is either an optical cross connect or an edge router. The \( e \) arcs represent trunks, where trunk \( i \) is composed of \( f \) fibers, each of which supports \( w \) wavelengths. We assume full wavelength conversion in this paper, so accordingly, trunk \( i \) carries \( C_i = f \times w \) unidirectional wavelength channels, which are called channels. Note that our model is also applicable for networks where OBS uses subwavelength channels [41], in which case the term channel represents a subwavelength channel. If all trunks have the same number of channels, then \( C_j = C \) for all \( j \). However, we note that the results presented in this paper are equally applicable to networks with no wavelength conversion, which have \( f_j \) instead of \( f \times w \), channels on each intermediate trunk (excluding the first trunk) in a route.

Each unique pair of origin and destination nodes forms a directional source–destination (SD) pair, \( m \). The set of all SD pairs in the network is denoted \( \beta = \{1, 2, ..., N(N - 1)/2\} \). Thus, \( m = \{x, y\} \in N \) represents traffic composed of bursts sent from node \( x \) to node \( y \). These bursts arrive at node \( x \) according to a Poisson process with parameter \( \rho_m \). For tractability, the burst lengths are assumed to be exponentially distributed with a unit mean. The effect of this exponential assumption has been numerically studied in [34], and it has been demonstrated by comparison to scenarios involving heavy tailed bursts that the performance results are only to a small extent sensitive to the shape of the burst length distribution. It is very likely that for a directional SD pair \( m \in \beta \), there is more than a single route between the source and the destination. We designate a route with the least number of hops as the primary path of the SD pair. Then all the other routes are ranked alternative paths.

It is convenient to maintain the set

\[
\{U_m(0), U_m(j_1(1), U_m(j_2(1), ..., U_m(j_n(T_m))\}
\]

deferred.

In this set, \( U_m(0) \) denotes the primary path, and \( U_m(j(d) \) denotes the alternative path with traffic deflected from trunk \( j \), which including this deflection has already been deflected \( d \) times. \( T_m \) is the maximum number of available alternative paths for the directional SD pair \( m \). Note that \( T_m \) is based on the network topology, which limits the number of available alternative paths, for example, \( T_m = 0 \) in the trivial example of a network of two nodes and two opposite-directional trunks that connect the two nodes.

In our model, the ranking of alternative paths is based on the number of hops, and in the case of equality in the number of hops, the rank is chosen randomly. However, in practice, various cost functions (e.g., geographic distance) can also be used for ranking. If capacity is available on all trunks of the primary path, then it will be used for the transmission of a burst from the source node \( x \) to the destination node \( y \). However, if all the channels are occupied on at least one of the trunks of the primary route, then a burst will be deflected to the first trunk of the first alternative path of that blocked trunk. If there is a free channel on this trunk, then the burst is transmitted on it; otherwise, the burst is deflected to the first trunk of the second alternative path.

A given burst is permitted to be deflected at most \( D \) times. A burst is blocked, that is, dumped and cleared from the network, if it arrives at a given node where all output trunks are busy or if, while trying alternative trunks, the burst reaches the limit \( D \) of allowable number of deflections. Setting the limit \( D \) implies that a burst in the directional SD pair \( m \) can be deflected no more than

\[
T(m) = \min\{T_m, D\}
\]
times.

This paper does not consider the use of trunk reservation as in [34]. Obtaining OPCA bounds for an OBS/JET network with trunk reservation is still an open problem.

III. OVERFLOW PRIORITY CLASSIFICATION APPROXIMATION FOR OBS NETWORKS

A detailed description of the OPCA is given in [34] and [35]. To be self-contained, the paper repeats this definition using the earlier notation of [34].

For each SD pair \( m \), let \( \rho_m \) be the offered traffic load. The term \( k \)-deflection burst is used to represent a burst that has been deflected \( k \) times \( \{k \in \{1, ..., T(m)\}\} \). Original bursts are the bursts that have not been deflected. In other words, they are 0-deflection bursts. Let \( a^k_j(m) \) be the \( k \)-deflection bursts’ offered traffic load of SD pair \( m \) on trunk \( j \), and let \( b^k_j \) denote the probability that a \( k \)-deflection burst is
blocked on trunk \( j \). If the first trunk of the primary route between SD pair \( m \) is \( i_1 \), then the offered load to this trunk is equated to the offered load of the SD pair, i.e.,

\[
a^0_{i_1}(m) = a^0_i(m)(1 - b^0_i) = \rho_m(1 - b^0_i).
\]

When the network is congested and the trunk of a given route is fully occupied, then the bursts that initially tried to use the original trunk are deflected onto alternative trunks and routes. Let trunk \( k \) be the busy trunk that transmits \( k \)-deflection bursts. Then there is a deflection on the present route being caused by this trunk \( k \). The load offered to the first trunk \( i_1 \) of the first choice alternative route is related to the load offered to trunk \( k \) by

\[
a^k_{i_1}(m) = a^k_i(m)b^k_i.
\]

where \( k \) is the number of deflections prior to the latest deflection. Similarly, due to the deflection from \( i_1 \), the load offered to the first trunk \( i_2 \) of the second choice alternative route is

\[
a^{k+1}_{i_2}(m) = a^{k+1}_i(m)b^{k+1}_i.
\]

Let \( a^k \) be the total offered load of \( k \)-deflection bursts on trunk \( j \). The variables \( a^k \) and \( a^k_i(m) \) for \( k = 0, 1, ..., D \) are related by

\[
a^k_j = \sum_{m=0}^{\infty} a^k_i(m).
\]

Assume

\[
I(i, j; U_{m,p}(k)) = \begin{cases} 
1, & \text{if } i, j \in E \text{ and trunk } i \text{ strictly precedes (not necessarily immediately) trunk } j \text{ along } k \text{ deflection route } U_{m,p}(k) \\
0, & \text{otherwise.}
\end{cases}
\]

Equation (4) can also be written as

\[
a^k_j = \sum_{m=0}^{\infty} \rho_{m,p} \prod_{i \in E} (1 - I(i, j; U_{m,p}(k))b^k_i)
\]

where \( \rho_{m,p} \) is the offered load from trunk \( p \) to the \( k \)th deflection route of SD pair \( m \) for \( k > 1 \), and

\[
a^0_j = \sum_{m=0}^{\infty} \rho_{m,0} \prod_{i \in E} (1 - I(i, j; U_{m,0}(k))b^0_i)
\]

for the primary layer.

In addition, let \( \tilde{a}^k_j \) be the offered load of bursts that have been deflected up to \( k \) times, i.e.,

\[
\tilde{a}^k_j = \sum_{h=0}^{k} a^h_j.
\]

The averaged blocking probability \( \tilde{b}^k_j \) on trunk \( j \in E \) for bursts with deflections up to \( k \) times is equal to

\[
\tilde{b}^k_j = E(\tilde{a}^k_j, C_j).
\]

where \( E(x, C) = (x^C/C!)(\sum_{n=0}^{C} x^n/n!) \) is the Erlang-B formula with offered load \( x \) and the number of channels per trunk is \( C \). The blocking probability for \( k \)-deflection burst, \( k \in \{0, ..., D\} \) on trunk \( j \) is estimated by

\[
b^k_j = \begin{cases} 
\tilde{b}^k_j, & k = 0 \\
\frac{\tilde{b}^{k-1}_j - \tilde{b}^{k-2}_j}{\tilde{a}^k_j}, & 1 \leq k \leq D.
\end{cases}
\]

Note that the blocking probability of undeflected bursts is calculated by using the Erlang-B formula.

To obtain the OPCA blocking probability estimates, we start with the primary traffic, i.e., \( k = 0 \). Then, we solve the fixed-point equations described by Eqs. (6)–(9), with the aid of the successive substitution method in order to obtain the values \( a^k_j \) for \( j \in E \) and \( b^0_j = b^0_j = E(\tilde{a}^0_j, C_j) \). These calculations for the primary traffic and blocking probabilities are defined as layer 0 calculations.

Next, having completed the layer 0 calculations to obtain the parameters related to the primary traffic \( (k = 0) \), we progress to compute the parameters associated with the first deflection traffic \( (k = 1) \). Similarly, we solve the fixed-point equations, Eqs. (5) and (7)–(9) (existence of a solution follows from Brouwer’s theorem [42]; e.g., see [43], where Brouwer’s theorem [42] was used to prove the existence of a solution of the fixed-point equation used in that paper), to obtain the values \( a^1_j \) for \( j \in E \), as well as \( \tilde{b}^1_j \) and \( b^1_j \) using Eqs. (8) and (9), respectively, for every \( j \in E \), where \( \tilde{a}^1_j \) is given by Eq. (7).

Then, having completed the layer 0 and layer 1 calculations to obtain the parameters related to the primary and the first deflection traffic \( (k = 0 \text{ and } k = 1) \), we compute the parameters associated with the second deflection traffic \( (k = 2) \), which we call the layer 2 calculations.

The process of deriving the parameters for \( k > 1 \) repeats itself until we have all the parameter values for all \( k \in \{1, ..., D\} \) layers.

IV. BOUNDS OF TRUNK BLOCKING PROBABILITIES OF OPCA

In this section we derive the OPCA upper and lower bounds and some properties of the bounds.

Let \( b^{0,k}_j \) denote the blocking probability obtained in the \( s \)th iteration for \( k \)-deflection bursts (\( k \)th layer) on trunk \( j \in E \), and let \( a^{s,k}_j \) be the offered traffic load obtained in the \( s \)th iteration for \( k \)-deflection bursts on trunk \( j \in E \). Let \( \{a^{s,k}_j, b^{s,k}_j\} \) be a set of pairs of the fixed-point solutions of Eqs. (2)–(9). Let us denote by \( a^{s,k}_{\min} \) and \( a^{s,k}_{\max} \) the minimum and maximum values, respectively, among the set of values \( \{a^{s,k}_j\} \) and denote by \( b^{s,k}_{\min} \) and \( b^{s,k}_{\max} \) the minimum and maximum values, respectively, among the set of values \( \{b^{s,k}_j\} \).

Denote the upper and lower bounds of the set \( \{b^{s,k}_j\} \) by \( b^{u}_{\min} \) and \( b^{l}_{\min} \), respectively, and let the upper and lower bounds of the set \( \{a^{s,k}_j\} \) be \( a^{u}_{\max} \) and \( a^{l}_{\max} \), respectively.
Consider now the network model presented in Fig. 1(a). There are three SD pairs: A to D, D to H, and C to B, with traffic in the network, where only primary paths are allowed. In Fig. 1(b), triangles are used to represent a trunk, and arrows from any trunk i to trunk j are used in the cases where traffic that passes through trunk i follows directly to trunk j. Trunks from 2 to 6 in that figure form a closed loop.

For trunks that are inside a closed loop, such as trunks 2–6 in Fig. 1(b), and for trunks that receive any traffic that passed through the trunks that are inside a closed loop, such as trunk 7 in the same figure, their offered load and trunk blocking probability are calculated on the basis of the successive substitution method, leading finally to the fixed-point solution. For any layer k, we will define a set of trunks that include all such trunks.

**Definition:** A set of trunks $L_k$ is a loop based trunk (LBT) set in layer $k$ if the following hold:

1) $L_k$ has at least one closed loop of trunks in layer $k$.
2) If in layer $k$, trunk $a$ receives traffic from any trunk in $L_k$, then $a \in L_k$.

If a trunk belongs to an LBT set, the trunk is designated as an LBT.

For simplicity, in the example presented in Fig. 1, we consider only one trunk (trunk 7) that is not included in the closed loop, but it is part of the loop tree. In general, we can have a large set of trunks (which may even include other loops) that receive traffic from the LBT, and such a set of trunks is included in the LBT set.

All the trunks that are not LBTs in layer $k$ are called direct trunks in layer $k$. Following are three examples (non-exhaustive) for the direct trunks in layer $k$:

1) A direct trunk where there is no traffic in a trunk in layer $k$, so there is no offered load there. In this case, the trunk blocking probability in layer $k$ is set to zero.
2) A set of trunks that form a tree structure (namely, traffic is directed toward the leaves of the trees) in layer $k$. In this case, all the bounds of the offered load and trunk blocking probabilities for all trunks can be calculated one by one directly without iterations in that layer, from top to bottom of the tree.

3) A set of trunks that feed traffic to a closed loop in layer $k$—such trunks are not in the loop, and they do not receive traffic from the loop. For instance, trunk 1 in Fig. 1(b). In this case, the bounds of the offered load and trunk blocking probability are calculated one by one directly without iterations as well. These trunks can also form one or more tree structures.

Calculation of the bounds for the trunk blocking probabilities starts from layer 0. After $s \geq 2$ iterations, for trunk $j \in E$, we obtain the upper bound $b_{ij}^{0}$ and the lower bound $b_{ij}^{0}$ for the trunk blocking probability and the upper bound $a_{ij}^{0}$ and the lower bound $a_{ij}^{0}$ for the trunk offered load. We prove that $b_{ij}^{0} = b_{ij}^{0} = a_{ij}^{0} = a_{ij}^{0}$ for the direct trunks.

For the LBTs, when $s$ increases, $b_{ij}^{0}$ and $b_{ij}^{0}$ become closer to each other and the set of fixed-point solutions for trunk blocking probability $b_{ij}^{0}$ are always between $b_{ij}^{0}$ and $b_{ij}^{0}$; $a_{ij}^{0}$ and $a_{ij}^{0}$ also become closer to each other, and the set of fixed-point solutions for trunk offered load $a_{ij}^{0}$ are always between $a_{ij}^{0}$ and $a_{ij}^{0}$. Then the upper and lower bounds for the traffic load offered from SD pair $m$ to trunk $j$ is calculated based on the bounds of the trunk blocking probabilities. After that, we calculate the upper bound $a_{ij}^{0}(m)b_{ij}^{0}$ and the lower bound $a_{ij}^{0}(m)b_{ij}^{0}$ of the overflowed traffic to layer 1 from SD pair $m$ on the congested trunk $q$ in layer 0 for $m \in \beta$, $q \in U_{m}(0)$.

After obtaining the bounds for the offered load and trunk blocking probabilities for all SDs, including direct trunks and LBTs in layer 0, we then perform the calculations for layer 1. After $s \geq 2$ iterations, for trunk $j \in E$ we obtain the upper bound $b_{ij}^{1}$ and lower bound $b_{ij}^{1}$ for the trunk blocking probability and the upper bound $a_{ij}^{1}$ and the lower bound $a_{ij}^{1}$ for the trunk offered load. For direct trunks, we obtain either $a_{ij}^{1} = a_{ij}^{1} = a_{ij}^{1}$, and $b_{ij}^{1} = b_{ij}^{1} - b_{ij}^{1} = b_{ij}^{1}$ for the trunk offered load. For direct trunks, we obtain either $a_{ij}^{1} = a_{ij}^{1} = a_{ij}^{1}$, and $b_{ij}^{1} = b_{ij}^{1} - b_{ij}^{1} = b_{ij}^{1}$. For LBTs, the bounds have the same properties as those of the LBTs in layer 1. Then we calculate the bounds of the overflowed traffic to layer 2.

The procedure repeats itself until the bounds of the offered load and blocking probabilities in each trunk for all layers are found. In each layer, we first calculate the bounds for the offered load and trunk blocking probabilities for the direct trunks, and then we iteratively calculate the bounds for the LBTs. Then, based on the bounds of the trunk offered load and blocking probabilities, we obtain the bounds of the network blocking probabilities estimated by the OPCA and prove that the fixed-point solutions are always between these upper and lower bounds.

The following subsections provide the equations used for obtaining the bounds for the blocking probability and the offered load estimated by the OPCA in each trunk for all the layers and the OPCA network blocking probability, as well as the proof of the behavior of the bounds.
A. Bounds of Trunk Blocking Probabilities for Direct Trunks

If the primary layer (layer 0) is a layer with direct trunks, then its upper and lower bounds for the offered load and trunk blocking probabilities are calculated from top to bottom of the tree by the following equations:

\[ a_j^{0k} = \sum_{m \in \mathcal{M}, j \in \mathcal{J}(0)} \rho_m \prod_{i \in \mathcal{E}} (1 - I(i, j, U_m(0)) b_i^{k0}), \tag{10} \]

\[ a_j^{0k} = \sum_{m \in \mathcal{M}, j \in \mathcal{J}(0)} \rho_m \prod_{i \in \mathcal{E}} (1 - I(i, j, U_m(0)) b_i^{k0}), \tag{11} \]

\[ b_j^{k0} = E(a_j^{k0}, C_j). \tag{12} \]

\[ b_j^{k0} = E(a_j^{k0}, C_j). \tag{13} \]

The calculations start from trunk \( j \) on the top of the tree. The offered load of this trunk does not depend on those in other trunks, so the bounds of its offered load are calculated directly using Eqs. (10) and (11). We obtain \( a_j^{0k} = a_j^{0k} = a_j^{0k} = a_j^{0k} \). This indicates uniqueness of the fixed-point solution for this trunk \( j \). Then, substituting the bounds for the offered load into Eqs. (12) and (13), we obtain the bounds of the trunk blocking probability of this trunk, \( b_j^{k0} = b_j^{k0} = b_j^{k0} = b_j^{k0} \). After that, we pass to the next trunk, that below the top trunk in the tree. The bounds of the offered load of this trunk depend only on those of the top trunk, so substituting the bounds of the trunk blocking probability of the top trunk into Eqs. (10) and (11), we obtain the bounds of the offered load for this second highest trunk. Then, substituting these bounds into Eqs. (12) and (13), we obtain the bounds of the trunk blocking probability of this second highest trunk. Repeating the steps from top to bottom in the tree, we obtain all the bounds of the offered load and trunk blocking probability one by one. They are \( a_j^{0k} = a_j^{0k} = a_j^{0k} = a_j^{0k} \) and \( b_j^{k0} = b_j^{k0} = b_j^{k0} = b_j^{k0} \).

For the same primary layer, the lower and upper bounds of the offered load for SD pair \( m \) are calculated by the formulas

\[ a_j^{0m} (m) = \rho_m \prod_{i \in \mathcal{E}} (1 - I(i, j, U_m(0)) b_i^{k0}), \tag{14} \]

\[ a_j^{0m} (m) = \rho_m \prod_{i \in \mathcal{E}} (1 - I(i, j, U_m(0)) b_i^{k0}). \tag{15} \]

Note that the notation \( a_j^{0m} (m) \) and \( a_j^{0m} (m) \) given by Eqs. (14) and (15) is not the same as the previously defined notation \( a_j^{0m} \), and the value \( a_j^{0m} \) is the lower bound of the total offered load to trunk \( j \) in layer 0, while \( a_j^{0m} (m) \) is the lower bound of the offered load to trunk \( j \) by SD pair \( m \). The difference between the notation \( a_j^{0m} \) and \( a_j^{0m} \) is explained similarly.

For layer \( k \), we assume that the offered load of bursts deflected less than \( k \) times is \( \tilde{a}_j^{k-1} \). If the layer having direct trunks is not the primary layer (that is, it is layer \( k > 0 \), then the overflowed traffic from SD pair \( m \) in the congested trunk \( q \) in layer \( k - 1 \) forms the traffic to the paths in layer \( k \). Then, the bounds of the offered load of the direct trunks in layer \( k \) are calculated by the equations

\[ a_j^{k^v} = \sum_{m \in \mathcal{M}, q \in \mathcal{J}(k)} \{ a_j^{k-1v} (m) b_q^{k-1v} \times \prod_{i \in \mathcal{E}} (1 - I(i, j, U_{m,q}(k)) b_i^{k-1v}) \}, \tag{16} \]

\[ a_j^{k^v} = \sum_{m \in \mathcal{M}, q \in \mathcal{J}(k)} \{ a_j^{k-1v} (m) b_q^{k-1v} \times \prod_{i \in \mathcal{E}} (1 - I(i, j, U_{m,q}(k)) b_i^{k-1v}) \}. \tag{17} \]

The equation for the trunk blocking probability is

\[ b_j^k = E(\tilde{a}_j^{k-1} + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \tag{18} \]

The left hand side of Eq. (18) increases when \( \tilde{a}_j^{k-1} \) or \( a_j^{k^v} \) increases (see Appendix A for the proof), and hence,

\[ E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \]

\[ \leq E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \]

\[ \leq E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \]

\[ \leq E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \]

Thus, the upper and lower bounds of the trunk blocking probabilities of direct trunks that are not in the primary layer are calculated by the formulas

\[ b_j^{k^v} = E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \tag{19} \]

\[ b_j^{k^v} = E(\tilde{a}_j^{k-1}, + a_j^{k^v}, C_j)(\tilde{a}_j^{k-1} + a_j^{k^v}) - E(\tilde{a}_j^{k-1}, C_j)(\tilde{a}_j^{k-1}) \tag{20} \]

The upper and lower bounds of the offered load to each direct trunk in SD pair \( m \) in layer \( k \) are determined by the formulas
For the kth layer, we first determine the bounds of the offered load in trunk j, which is on the top of the tree. The bounds of the offered load depend only on the overflowed traffic in the k−1 layer. From Eqs. (16) and (17) we obtain

\[ a_{j_k} = a_{j_{max}}^k = a_{j_{min}}^k \]  \( a_{j_{max}}^k = a_{j_{min}}^k \), provided that \( a_{j_k} \) is satisfied for all overflowed traffic into top trunk \( j \); in the opposite case, if \( a_{j_k} \) is not satisfied at least for one instance of the available overflowed traffic, then we obtain \( a_{j_k} > a_{j_{max}}^k \). [The equation \( a_{j_k} = a_{j_{max}}^k \) is satisfied if for the upper layer we obtain the exact value of the offered load]. Substituting these bounds of the offered load into Eqs. (19) and (20), we obtain the bounds for the blocking probability in the top trunk. If \( a_{j_k} = a_{j_{max}}^k \) for all overflowed traffic into the top trunk, then \( b_{j_k} = b_{j_{max}}^k = b_{j_{min}}^k = b_{j_k} \); otherwise, we obtain \( b_{j_k} > b_{j_{max}}^k \). In the next step, we calculate the bounds for the offered load and blocking probability for each of the trunks that are the second from the top. The calculation for each one these trunks is identical to the others and depends only on the blocking probability in the top trunk \( j \) and overflowed traffic from the k−1 layer. Therefore, without loss of generality, let \( j \) denote any of these trunks. We obtain \( a_{j_k} = a_{j_{max}}^k = a_{j_{min}}^k \) and \( b_{j_k} = b_{j_{max}}^k = b_{j_{min}}^k = b_{j_k} \) provided that \( a_{j_k} \) is satisfied for all overflowed traffic into trunk \( j \) and trunk \( j \). Otherwise, we have \( a_{j_k} > a_{j_{max}}^k \) and \( b_{j_k} > b_{j_{max}}^k \), respectively. Repeating these steps recurrently to the other direct trunks in the kth layer from top to bottom of the tree, we obtain all bounds for the offered load and trunk blocking probabilities.

B. Bounds of Trunk Blocking Probabilities for LBTs

In Proposition 1 below and in all other statements following, \( b_{j_k} \) denotes the initial (setup) value of the blocking probability for the kth layer on trunk \( j \).

**Proposition 1:** Assume that \( b_{j_k} = 0 \) (k = 0, 1, ..., D), \( j \in E \) with a nonzero offered load and trunk \( j \) is an LBT. Then, for any positive integer \( z \), we have the following inequalities for the lower and upper bounds of \( a_{j_k} \) and \( b_{j_k} \):

\[ a_{j_k}^z = a_{j_k} > a_{j_k}^z = a_{j_k}^z, \]
\[ a_{j_k}^z < a_{j_k} < a_{j_k}^z = a_{j_k}^z, \]
\[ b_{j_k}^z < b_{j_k} < b_{j_k}^z = b_{j_k}^z, \]
\[ b_{j_k}^z < b_{j_k} < b_{j_k}^z = b_{j_k}^z. \]

**Proof:** The computation starts from layer 0 of the primary path bursts. First we calculate lower and upper bounds of the offered load and trunk blocking probabilities for the direct trunks. Then, setting \( b_{j_k}^0 = 0 \), we provide calculations for other trunks, where \( j \) belongs to the LBT set and \( 0 \) denotes the first iteration in the primary path.

For the offered load on trunk \( j \), we have

\[ a_{j_k}^0 = \sum_{m \in j, E \cup \{0\}} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))) (1 - H(0, i)) b_{i_k} \]
\[ a_{j_k}^0 = \sum_{m \in E \cup \{0\}} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))) (1 - H(0, i)) b_{i_k}. \]

where

\[ I(i, j, U_m(0)) = \begin{cases} 1, & \text{if } i, j \in E \text{ and trunk } i \text{ strictly precedes (not necessarily immediately) trunk } j \text{ along primary route of SD pair } m \\ 0, & \text{otherwise}, \end{cases} \]

\[ H(k, i) = \begin{cases} 1, & \text{if trunk } i \text{ is an LBT in layer } k \\ 0, & \text{if trunk } i \text{ is a direct trunk in layer } k. \end{cases} \]

Hence, the blocking probability \( b_{j_k}^0 \) is obtained by the Erlang-B formula,

\[ b_{j_k}^0 = E(a_{j_k}^0, C_j). \]

According to assumption, \( b_{j_k}^0 = 0 \) and \( a_{j_k}^0 > 0 \) are true for all trunks \( j \) belonging to the LBT set in layer 0. Hence, comparing \( b_{j_k}^0 \) with \( b_{j_k} \) yields \( b_{j_k}^0 > b_{j_k} \) for all \( j \) belonging to the LBT set in layer 0.

By the successive substitution method, we are to replace \( b_{j_k} \) in Eq. (23) by \( b_{j_k} \), and then the load is updated by the formula

\[ a_{j_k}^0 = \sum_{m \in E \cup \{0\}} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))) (1 - H(0, i)) b_{i_k} \]
\[ a_{j_k}^0 = \sum_{m \in E \cup \{0\}} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))) (1 - H(0, i)) b_{i_k}. \]

Taking into account \( b_{j_k}^0 > b_{j_k} \) for all \( j \) belonging to the LBT set in layer 0 and \( b_{j_k}^0 = b_{j_k} \) when \( H(0, i) = 1 \), by comparing Eqs. (23) and (25), we arrive at the inequality \( a_{j_k}^0 < a_{j_k} \) for all \( j \) belonging to the LBT set in layer 0.

Repeating the steps above, for the value \( b_{j_k}^0 \), we obtain

\[ b_{j_k}^0 = E(a_{j_k}^0, C_j). \]

Since \( E(x, C) \) is an increasing function in \( x \), we have

\[ 0 = b_{j_k}^0 < b_{j_k} < b_{j_k}^0. \]
In the third iteration, we have the formula
\[
\begin{align*}
a_j^{0_3} &= \sum_{m \in \beta_i, j \in U_m(0)} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))(1 - H(0, i))b_i^{0_0}) \\
&\quad \times (1 - I(i, j, U_m(0))H(0, i))b_i^{0_0},
\end{align*}
\]
and then we arrive at
\[
b_j^{0_0} = E(a_j^{0_0}, C_j).
\] (28)

Since \(0 < b_j^{0_0} < b_j^{0_1} < b_j^{0_2}\), we have the property \(a_j^{0_0} < a_j^{0_1} < a_j^{0_2}\) and \(0 = b_j^{0_1} < b_j^{0_0} < b_j^{0_2}\) for all \(j \in L\).

Since \(a_j^{0_3} = \sum_{m \in \beta_i, j \in U_m(0)} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))(1 - H(0, i))b_i^{0_3}),\) we obtain \(a_j^{0_0} < a_j^{0_1} < a_j^{0_2} < a_j^{0_3}\) for all \(j\) belonging to the LBT set in layer 0.

In a similar way, we define the recurrent relations for \(s > 1:\)
\[
b_j^{0_0} = E(a_j^{0_{s-1}}, C_j).
\] (29)

\[
a_j^{0_0} = \sum_{m \in \beta_i, j \in U_m(0)} \rho_m \prod_{i \in E} (1 - I(i, j, U_m(0))(1 - H(0, i))b_i^{0_0}) \\
&\quad \times (1 - I(i, j, U_m(0))(1 - H(0, i))b_i^{0_0}).
\] (30)

Notice that \(b_j^{0_0} = b_j^{0_0}\) when \(H(0, i) = 0\).

Repeating the same procedure, we obtain all the necessary relations for \(a_j^{0_s}\) and \(b_j^{0_s}\) for given integer values \(s\). We also have the inequalities
\[
a_j^{0_0} < a_j^{0_1} < a_j^{0_2} < \ldots < \lim_{z \to \infty} a_j^{0_{s-1}} = a_j^{0_{\text{max}}},\]
\[
a_j^{0_0} < a_j^{0_1} < a_j^{0_2} < \ldots < \lim_{z \to \infty} a_j^{0_s} = a_j^{0_{\text{min}}}.\]

Let \(S\) be the number of iterations. We will write \(S = 2x\) in the case of an even number of iterations and \(S = 2x + 1\) in the case when the number of iterations is odd.

In the case of \(S = 2x\), the upper bound of the set of values \(\{a_j^{0_*}, s = 1, 2, \ldots, 2x\}\) is \(a_j^{0_*} = a_j^{0_{2x}}\) and the lower bound is \(a_j^{0_*} = a_j^{0_0}\). In the case of \(S = 2x + 1\), the upper bound of the set of these values is \(a_j^{0_*} = a_j^{0_{2x+1}}\) and the lower bound is \(a_j^{0_*} = a_j^{0_0}\).

For the set of the values \(\{b_j^{0_*}, s = 1, 2, \ldots, S\}\), the bounds are defined similarly:
\[
\begin{align*}
b_j^{0_0} > b_j^{0_1} > b_j^{0_2} > \ldots > \lim_{z \to \infty} b_j^{0_{s-1}} = b_j^{0_{\text{max}}},
\end{align*}
\] (33)
\[
\begin{align*}
b_j^{0_0} < b_j^{0_1} < b_j^{0_2} < \ldots < \lim_{z \to \infty} b_j^{0_{s-1}} = b_j^{0_{\text{min}}}.
\end{align*}
\] (34)

In the case of \(S = 2x\), the upper and the lower bounds of the set of values \(\{b_j^{0_*}, s = 1, 2, \ldots, 2x\}\) are \(b_j^{0_*} = b_j^{0_{2x}}\) and \(b_j^{0_*} = b_j^{0_{2x-1}}\), respectively. In the case of \(S = 2x + 1\), the upper and the lower bounds of the set of these values are \(b_j^{0_*} = b_j^{0_{2x}}\) and \(b_j^{0_*} = b_j^{0_{2x-1}}\), respectively.

Equations (31)–(34) follow by induction.

Indeed, we earlier proved Eqs. (31)–(34) for \(z = 1\). Hence, assuming that in the case \(z = i\) the inequalities
\[
a_j^{0_{2i-1}} > a_j^{0_{2i-2}} > a_j^{0_{2i-1}} > a_j^{0_{2i}},\]
\] (35)
are satisfied, we are to prove that the inequalities
\[
a_j^{0_{2i+1}} > a_j^{0_{2i-2}} > a_j^{0_{2i-1}} > a_j^{0_{2i+2}}\]
\] (36)
are true as well (case \(z = i + 1\)). At the next step, on the basis of Eq. (35), we prove that
\[
b_j^{0_{2i+3}} > b_j^{0_{2i-2}} > b_j^{0_{2i-1}} > b_j^{0_{2i+1}}.\]
\] (37)

Indeed, since the function \(F(x, C_j)\) is increasing in \(x\), then Eq. (37) follows from Eq. (35) by direct substitution of the values into Eq. (29).

Now on the basis of Eq. (37) we prove Eq. (36). The function \(F(x) = \prod_{i=1}^{n-1}(1-x_i)\) is a decreasing function in vector \(x = (x_1, x_2, \ldots, x_n)\). That is, for any two vectors \(x^{(1)} = (x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)})\) and \(x^{(2)} = (x_1^{(2)}, x_2^{(2)}, \ldots, x_n^{(2)})\) satisfying the componentwise inequalities \(x_i^{(1)} \leq x_i^{(2)}, i = 1, 2, \ldots, n,\) we have \(F(x^{(1)}) \geq F(x^{(2)})\). The strong inequality \(F(x^{(1)}) > F(x^{(2)})\) holds if, in addition, \(x_i^{(1)} < x_i^{(2)}\) is satisfied at least for one of indices \(i\).

Hence, substituting the values of Eq. (37) into Eq. (30), we arrive at the inequality
\[
a_j^{0_{2i+1}} > a_j^{0_{2i-2}} > a_j^{0_{2i-1}} > a_j^{0_{2i}}.\]
\] (38)

From Eq. (38), it is easy to obtain the desired inequality [Eq. (36)]. To this end, we first substitute Eq. (38) into Eq. (29). This yields
\[
b_j^{0_{2i+2}} > b_j^{0_{2i+1}} > b_j^{0_{2i-2}} > b_j^{0_{2i-1}}.\]
\] (39)

Finally, substituting Eq. (39) into Eq. (30) once again, we arrive at Eq. (36).

For layer 0, the lower and upper bounds of the offered load to each trunk of SD pair \(m\) are also calculated by Eqs. (14) and (15).

For layer 1, the bounds of overflowed traffic from SD pair \(m\) caused by the congestion in trunk \(q\) in layer 0 is calculated by the equation
\[
\begin{align*}
a_j^1 &= \sum_{m \in \beta_q, j \in U_m(1)} a_j^0(m) \prod_{i \in E} (1 - I(i, j, U_{m}(1))) \\
&\quad \times (1 - H(1, i)b_i^1)(1 - I(j, i, U_{m}(1))H(1, i)b_i^1).
\end{align*}
\] (40)

On the basis of the lower and upper values for each \(a_j^0(m)\) and \(b_i\) for direct trunks \(i\), we have the following transformed formulas of Eq. (40):
Li et al.

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))) \\
\times (1 - H(1, i)b_{i}^{1v})(1 - I(i, j, U_{m,q}(1))H(1, i)b_{i}^{1v}). \quad (41) \]

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))) \\
\times (1 - H(1, i)b_{i}^{1v})(1 - I(i, j, U_{m,q}(1))H(1, i)b_{i}^{1v}). \quad (42) \]

Since \( a_{j}^{1v} \) increases when the term \( a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \) increases, then substituting the same value of \( b_{i}^{1v} \) into Eqs. (41) and (42), we obtain the inequality \( a_{j}^{1v} > a_{j}^{1v} \).

Setting \( b_{j}^{1v} = 0 \), for upper and lower bounds of \( a_{j}^{1v} \) and \( b_{j}^{1v} \), we have the following relations:

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))) \\
\times (1 - H(1, i)b_{i}^{1v})(1 - I(i, j, U_{m,q}(1))H(1, i)b_{i}^{1v}), \quad (43) \]

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))) \\
\times (1 - H(1, i)b_{i}^{1v})(1 - I(i, j, U_{m,q}(1))H(1, i)b_{i}^{1v}), \quad (44) \]

Then we have the inequalities

\[ a_{j}^{1v} > a_{j}^{1v} > a_{j}^{1v} > \ldots \text{lima}_{z=\infty} a_{j}^{1v} \geq a_{j}^{1v}, \quad (45) \]

\[ a_{j}^{1v} < a_{j}^{1v} < a_{j}^{1v} < \ldots \text{lima}_{z=\infty} a_{j}^{1v} \leq a_{j}^{1v}, \quad (46) \]

\[ b_{j}^{1v} > b_{j}^{1v} > b_{j}^{1v} > \ldots \text{limb}_{z=\infty} b_{j}^{1v} \geq b_{j}^{1v}, \quad (47) \]

\[ b_{j}^{1v} < b_{j}^{1v} < b_{j}^{1v} < \ldots \text{limb}_{z=\infty} b_{j}^{1v} \leq b_{j}^{1v}. \quad (48) \]

In the case \( S = 2z \), the upper and lower bounds of \( a_{j}^{1v} \) are \( a_{j}^{1v} = a_{j}^{1v} \) and \( a_{j}^{1v} = a_{j}^{1v} \), respectively, and the upper and lower bounds of \( a_{j}^{1v} \) are \( a_{j}^{1v} = a_{j}^{1v} \) and \( a_{j}^{1v} = a_{j}^{1v} \), respectively. In the case \( S = 2z + 1 \), the upper bound of \( a_{j}^{1v} \) is changed to \( a_{j}^{1v} = a_{j}^{1v} \), and the lower bound of \( a_{j}^{1v} \) is changed to \( b_{j}^{1v} = b_{j}^{1v} \).

The proof of Eqs. (45)-(48) is similar to that of Eqs. (31)-(34).

The lower and upper bounds of the offered load to each trunk of SD pair \( m \) in layer 1 are determined by the equations

\[ a_{j}^{1v}(m) = \sum_{q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))b_{i}^{1v}). \quad (49) \]

\[ a_{j}^{1v}(m) = \sum_{q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(1)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(1))b_{i}^{1v}). \quad (50) \]

The upper and lower bounds of \( \tilde{a}_{j}^{1v} \) are \( \tilde{a}_{j}^{1v} = \sum_{i=0}^{\infty} a_{j}^{1v} \) and \( \tilde{a}_{j}^{1v} = \sum_{i=0}^{\infty} a_{j}^{1v} \).

The values of \( a_{j}^{1v} \) and \( b_{j}^{1v} \) are calculated by the formulas

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(k)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(k))(1 - H(i, k)b_{i}^{1v}) \times (1 - I(i, j, U_{m,q}(k))H(i, k)b_{i}^{1v}), \quad (51) \]

\[ a_{j}^{1v} = \sum_{m \in \mathcal{G}, q \in \mathcal{F}, j \in \mathcal{L}_{m,q}(k)} a_{q}^{b_{1}}(m)b_{q}^{h_{1}} \prod_{i \in E} (1 - I(i, j, U_{m,q}(k))(1 - H(i, k)b_{i}^{1v}) \times (1 - I(i, j, U_{m,q}(k)))H(i, k)b_{i}^{1v}), \quad (52) \]

\[ b_{j}^{1v} = \frac{E(a_{j}^{1v} + a_{j}^{1v}, C_{j})(a_{j}^{1v} + a_{j}^{1v}) - E(a_{j}^{1v}, C_{j})(a_{j}^{1v})}{a_{j}^{1v}}, \quad (53) \]

\[ b_{j}^{1v} = \frac{E(a_{j}^{1v} + a_{j}^{1v}, C_{j})(a_{j}^{1v} + a_{j}^{1v}) - E(a_{j}^{1v}, C_{j})(a_{j}^{1v})}{a_{j}^{1v}}, \quad (54) \]

Then we obtain the inequalities

\[ a_{j}^{1v} > a_{j}^{1v} > a_{j}^{1v} > \ldots \text{lima}_{z=\infty} a_{j}^{1v} \geq a_{j}^{1v}, \quad (45) \]

\[ a_{j}^{1v} < a_{j}^{1v} < a_{j}^{1v} < \ldots \text{lima}_{z=\infty} a_{j}^{1v} \leq a_{j}^{1v}, \quad (46) \]

\[ b_{j}^{1v} > b_{j}^{1v} > b_{j}^{1v} > \ldots \text{limb}_{z=\infty} b_{j}^{1v} \geq b_{j}^{1v}, \quad (47) \]

\[ b_{j}^{1v} < b_{j}^{1v} < b_{j}^{1v} < \ldots \text{limb}_{z=\infty} b_{j}^{1v} \leq b_{j}^{1v}. \quad (48) \]

In the case \( S = 2z \), the upper and lower bounds of \( a_{j}^{1v} \) are \( a_{j}^{1v} = a_{j}^{1v} \) and \( a_{j}^{1v} = a_{j}^{1v} \), respectively, and the upper and lower bounds of \( a_{j}^{1v} \) are \( a_{j}^{1v} = a_{j}^{1v} \) and \( a_{j}^{1v} = a_{j}^{1v} \), respectively. In the case \( S = 2z + 1 \), the upper bound of \( a_{j}^{1v} \) is changed to \( a_{j}^{1v} = a_{j}^{1v} \), and the lower bound of \( a_{j}^{1v} \) is changed to \( b_{j}^{1v} = b_{j}^{1v} \).

The proof of Eqs. (45)-(48) is similar to that of Eqs. (31)-(34).

The lower and upper bounds of the offered load to each trunk of SD pair \( m \) in layer \( k \) are determined by the formulas
\[ a_j^{ki}(m) = \sum_{q \in \mathcal{E}, \mu \in \{0, 1\}} (a_q^{k-1}(m)b_q^{k-1}) \prod_{i \in E} (1 - I(i, j, \mu, q)(b_i^j)), \quad (59) \]

\[ a_j^{k}(m) = \sum_{q \in \mathcal{E}, \mu \in \{0, 1\}} (a_q^{k-1}(m)b_q^{k-1}) \prod_{i \in E} (1 - I(i, j, \mu, q)(b_i^j)). \quad (60) \]

Corollary 1: Consider two OPA runs. Let the number of iterations in layer \(i\) in the first run be \(I_i\), and let the number of iterations in layer \(i\) in the second run be \(I'_i\). Let \(S_j^i\) and \(S_j^{i'}\) be the distance between the bounds for the blocking probability for trunk \(j\) in layer \(i\) in the first and second OPA runs, respectively. If \(I'_i > I_i\), for one layer and the numbers of iterations in other layers are the same, and there are LBTs in the layer, then \(S_j^i > S_j^{i'}\) for the trunks with the following situations:

1) if trunk \(j\) is a loop trunk, then for all \(k \geq i\), \(S_j^i > S_j^{k+1}\);  
2) if in a layer \(k' > i\), a trunk \(j\) that receives traffic that passes through or overflows from a trunk \(j\) in the upper layer satisfies the inequality \(S_j^{k-1} > S_j^{k+1}\), then for all \(k' < k \leq D\), \(S_j^i > S_j^{k+1}\).

Proof: According to the construction, for layers \(k < i\), the bounds of the trunk blocking probability for \(j \in E\) are the same in two OPA runs.

Since there are LBTs, then for layer \(i\) by Eqs. (57) and (58), for LBTs we obtain inequalities \(S_j^i < S_j^{i'} < \cdots < S_j^{i'}\), in which \(S_j^i = |b_j^{i+1} - b_j^{i+2}|\). Thus, if \(I'_i > I_i\), then \(S_j^i > S_j^{i'}\).

Also, if \(I'_i > I_i\), then by Eqs. (21), (22), and (55)–(58), we obtain \(a_j^{qi}(m) < a_j^{q}(m), a_j^{qi}(m) > a_j^{q}(m), b_j^{qi} > b_j^{q}, \) and \(b_j^{qi} > b_j^{q}\) for \(j \in E\) and \(m \in \mathcal{E}\).

For layer \(i + 1\), let us first consider a trunk \(j\) for which \(S_j^i > S_j^{i'}\) in layer \(i\) is satisfied. Then we have \(a_j^{q} < a_j^{qi}\) and \(a_j^{qi} < a_j^{q}\). Substituting these inequalities into Eqs. (19)–(22), we obtain the inequalities \(b_j^{qi+1} < b_j^{q+1i}, b_j^{qi+1} < b_j^{q+1i}, a_j^{qi+1}(m) < a_j^{q+1i}(m)\), and \(a_j^{qi+1}(m) > a_j^{q+1i}(m)\). Thus, if \(j\) is a loop trunk in layer \(i\), then we have \(S_j^{i+1} < S_j^{i+1}\).

Consider now a direct trunk \(j\) receiving traffic that overflowed from trunks \(j_1\), for which \(S_j^i > S_j^{i'}\) in layer \(i\), or passed through a trunk \(j_2\) in layer \(i + 1\), for which \(S_j^{i+1} > S_j^{i+1}\). Substituting the inequalities \(a_j^{qi}(m) < a_j^{q}(m)\) and \(a_j^{qi}(m) > a_j^{q}(m)\), \(b_j^{qi} < b_j^{q}\) and \(b_j^{qi} > b_j^{q}\), or \(b_j^{qi+1} < b_j^{q+1i}\) and \(b_j^{qi+1} > b_j^{q+1i}\) into Eqs. (17) and (16), we obtain \(a_j^{qi+1} < a_j^{q+1i}\) and \(a_j^{qi+1} > a_j^{q+1i}\). Then, substituting the inequalities into Eqs. (19)–(22), we obtain the inequalities \(b_j^{qi+1} < b_j^{q+1i}, b_j^{qi+1} > b_j^{q+1i}, a_j^{qi+1}(m) < a_j^{q+1i}(m)\), and \(a_j^{qi+1}(m) > a_j^{q+1i}(m)\). Thus, the inequality \(S_j^{i+1} > S_j^{i+1}\) is satisfied if trunk \(j\) is a direct trunk receiving traffic that overflowed from LBTs in layer \(i\).

Let us consider now LBTs receiving traffic that overflowed from the LBTs in layer \(i\) or passing through a direct trunk for which the inequality \(S_j^{i+1} > S_j^{i+1}\) is satisfied in layer \(i + 1\). We start from the setup value \(b_j^{i+1} = 0\). According to Eq. (51) for layer \(i + 1\), we obtain the inequality \(a_j^{i+1} > a_j^{i+1}\). Substituting this inequality into Eq. (53), we in turn obtain \(b_j^{i+2} > b_j^{i+1}\). Repeating the procedure, we again substitute the inequality obtained for Eq. (52) and now obtain \(a_j^{i+2} > a_j^{i+1}\). Further substitution of the inequality obtained for Eq. (54) yields \(b_j^{i+2} > b_j^{i+1}\). Repeating the same steps for any integer \(z\), we finally arrive at the following inequalities:

\[ a_j^{i+1z-1} > a_j^{i+1z-1}, \]
\[ b_j^{i+1z} > b_j^{i+1z-1} > b_j^{i+2z-1}. \]

By the same method in layer \(i\), we obtain \(S_j^{i+1} > S_j^{i+1}\). In addition, we obtain \(a_j^{i+1}(m) > a_j^{i+1}(m) > a_j^{i+1}(m)\) and \(a_j^{i+1}(m) > b_j^{i+1}(m)\). Repeating the same steps as in layer \(i + 1\), we obtain the same solution for layer \(k > i + 1\).

C. Bounds for Network Blocking Probability of the OPA

Proposition 2: Let \(B_U(m)\) and \(B_L(m)\) denote the upper and the lower bounds, respectively, of the network blocking probability for SD pair \(m\). We have

\[ B_U^\prime(m) = 1 - \frac{\rho_m \prod_{\mu \in \{0, 1\}} (1 - b_j^{\mu}(m))}{\rho_m} - \frac{\sum_{q \in \mathcal{E}} \sum_{i=1}^T (m) \mu_{\mathcal{U}^q}(m) \prod_{\mu \in \{0, 1\}} (1 - b_j^{\mu}(m))}{\rho_m}, \quad (61) \]

where

\[ \mu_{\mathcal{U}^q}(m) = \begin{cases} \rho^{q-1}(m) \rho^q(m), & \text{if path } U_{m, q}(h) \text{ exists}, \\ 0, & \text{otherwise}. \end{cases} \]

and we have

\[ B_L^\prime(m) = 1 - \frac{\rho_m \prod_{\mu \in \{0, 1\}} (1 - b_j^{\mu}(m))}{\rho_m} - \frac{\sum_{q \in \mathcal{E}} \sum_{i=1}^T (m) \mu_{\mathcal{U}^q}(m) \prod_{\mu \in \{0, 1\}} (1 - b_j^{\mu}(m))}{\rho_m}, \quad (62) \]

where

\[ \mu_{\mathcal{U}^q}(m) = \begin{cases} \rho^{q-1}(m) \rho^q(m), & \text{if path } U_{m, q}(k) \text{ exists}, \\ 0, & \text{otherwise}. \end{cases} \]

Proof: In order to calculate the blocking probability \(B(m)\) for SD pair \(m\), we are required first to calculate the received load from every path by the destination node. Then, we calculate the probability that a message will be served and the blocking probability.
\[ B(m) = 1 - \frac{\rho_m \prod_{i \in U_m(0)} (1 - b_i^0)}{\rho_m} - \sum_{q \in E} \sum_{k=1}^T (m) \rho_{U_m(k)} \prod_{p \in U_m(p)} (1 - b_p^k) \frac{\rho_m}{\rho_m}, \]  

where \( \rho_{U_m(k)} \) is the offered load to the path \( U_{m,q}(k) \), and it is calculated by the formula

\[ \rho_{U_m(k)} = \begin{cases} a_m^{-1}(m)b_q^{-1}, & \text{if path } U_{m,q}(k) \text{ exists} \\ 0, & \text{otherwise}. \end{cases} \]

After calculation of the lower and upper bounds for \( a_m^k \) and \( b_q^k \) (\( k = 0, 1, \ldots, T(m) \)), we derive Eqs. (61)–(63). For more iterations, the sequence \( B^U(m) \) is not increasing. If there is at least one layer with LBTs, then for more iterations the sequence \( b_i^k \) is decreasing and \( \rho_{U_m(k)} \) is a decreasing sequence in \( m \in \beta \) and \( k = 0, 1, \ldots, T(m) \) as well. Hence, \( B^U(m) \) is a decreasing sequence. Unlike \( B^U(m) \), \( B^L(m) \) is not an increasing sequence. When there is at least one layer with LBTs, the sequence \( B^L(m) \) is decreasing for more iterations of the OPCA algorithm.

Substituting \( a_m^k \) and \( b_q^k \) into Eqs. (61)–(63), we arrive at the inequality \( B^U_0 \geq B_{max}(m) \geq B_{min}(m) \geq B^L_0 \), where \( B^U_{max}(m) \) and \( B^U_{min}(m) \) denote the maximum and minimum value in the set \( \{B^U(m)\} \) of the fixed-point solutions.

**Proposition 3:** Let \( \{B^*\} \) be the set of the network blocking probabilities obtained by the fixed-point solutions for trunk blocking probabilities in trunks \( j \in E \). Let \( B^*_{min} \) and \( B^*_{max} \) denote the minimum and maximum values among the set of values \( \{B^*\} \). Let \( B_U \) and \( B_L \) denote the upper and lower bounds of the network blocking probabilities. We have

\[ B_U \geq B^*_{max} \geq B^*_m \geq B_L. \]

\( B_U \) and \( B_L \) are calculated by the equations

\[ B_U = \frac{\sum_{m \in \mathbb{R}} B^U(m)}{\#SD}, \]  

\[ B_L = \frac{\sum_{m \in \mathbb{R}} B^L(m)}{\#SD}, \]

where \#SD is the number of SD pairs in the network.

**Proof:** Substituting \( B^U_{min} \geq B^U_{max}(m) \geq B^*_min(m) \geq B^L_0 \) into Equations (64) and (65), we obtain the bounds for the network blocking probability

\[ B_U \geq B^*_{max} \geq B^*_m \geq B_L. \]

**Corollary 2:** Consider an OPCA run. Let \( M \) be the distance between the bounds of the network blocking probability. Increasing the number of iterations in one or more layers, among which at least one of those layers has LBTs, and obtaining the new distance between the bounds of the network blocking probability \( M' \), then \( M' < M \).

**Proof:** Corollary 1 shows that when the number of iterations increases in one layer having trunk loops, then the distance between the upper and lower bounds of the trunk blocking probability for some trunks decreases. If there is at least one trunk \( j \), in which the distance between the upper and lower bounds of the trunk blocking probability decreases in layer \( i \), then for SD pair \( m \) having traffic passing through trunk \( j \) in layer \( i \) we have the inequalities \( B^U(m) > B^U(m) \) and \( B^L(m) < B^L(m) \). Then, we arrive at \( B_U > B_U \) and \( B_L < B_L \). Hence, for \( M = B_U - B_L \) and \( M' = B_U' - B_L' \), we arrive at \( M' < M \).

If the trunks in all layers are direct trunks, then the solution obtained by the OPCA is not a fixed-point solution, and it is obtained in only a finite number of steps. This number of steps is bounded by \( J \times (D + 1) \), where \( J \) is the total number of trunks and \( D \) is the maximum number of allowable deflections in the network. If there are LBTs at least in one layer, then the bounds of the OPCA fixed-point solutions always become closer when the numbers of iterations in those layers are increased.

Thus, either the OPCA finds its solution in a finite number of steps, or the bounds of its fixed-point solutions always become closer with increasing number of iterations.

**V. NUMERICAL RESULTS**

In this section we provide numerical results for OBS/JET without trunk reservation over a 13-node NSFNET in order to illustrate the behavior of the bounds of the OPCA algorithm. In particular, we focus on illustrating how the bounds become closer to each other with an increase of the number of iterations. The numerical results in this section demonstrate that the bounds become closer to each other even after a small number of iterations (per layer). We will illustrate here the behavior of the OPCA blocking probability bounds for the aforementioned network considering a wide range of parameters and design factors, such as the number of channels per trunk and the maximum allowable number of deflections. We will also compare the running time and accuracy of the EPCA and OPCA algorithms.

In all the scenarios considered, the arrival process of calls for each directional SD pair follows a Poisson process. The shortest path is set to be the primary route for each SD pair, and the alternative routes are preassigned, ordered by their length. For those routes with the same lengths, the order is chosen randomly and remains unchanged afterward. All the results in this section are obtained using MATLAB software executed on a laptop with an Intel Core i7-3520M CPU at 2.96 GHz with 8 GB RAM and a 64 bit operating system.

**A. Network Topology and SD Pairs**

We now consider the NSF network with 13 nodes and 30 directional trunks. The topology of the NSF network is...
shown in Fig. 2. We randomly select a set of 12 SD pairs shown in Table I.

### B. Comparison of the Computational Time of the EFPA and OPCA

Table II provides the running times used to calculate the blocking probability in NSFNET for different C values by the EFPA and OPCA. The offered load to each SD pair is 0.5C. In each iteration, we only consider four significant digits of the fixed-point solutions for the trunks with blocking probabilities larger than 10^{-50}, whereas trunk blocking probabilities lower than 10^{-50} are set equal to 0. From Table II, we observe that the EFPA consumes much more time than the OPCA for all three different C values, and when the C value increases, the computational time of the EFPA grows faster than that of the OPCA. To gain some insight into the reason why the EFPA consumes much more time than the OPCA, we count the total number of iterations required by the EFPA, and by each layer of the OPCA, with C = 10,000. The results are shown in Table III. We observe that the EFPA requires 78 iterations to converge, but the first layer of the OPCA only requires 6 iterations, and the other layers require even fewer iterations. Layer 3 for the OPCA algorithm consumes only 0.0024 s, because there is no overflowed traffic from layer 2; therefore, the offered load to each trunk in layer 3 is 0, in which case all the layer 3 trunk blocking probabilities equal to 0 without the need to run the Erlang-B formula.

### C. Accuracy of OPCA and EFPA for the NSFNET

Having demonstrated that the OPCA converges faster than the EFPA, it is important to evaluate the accuracy of the two algorithms to see whether the longer running time enables the EFPA to provide more accurate results than the OPCA. Note also that the results presented here are for an OBS network without trunk reservation, so these results also complement those in [34], which apply to networks with trunk reservation.

Results for the comparison of the accuracy of the OPCA and EFPA for the case C = 50 are presented in Fig. 3. The results are limited to the case C = 50, because simulations for large C values are computationally prohibitive. The results are based on a comparison of the two approximations with simulation results for the case of the NSFNET example of Fig. 2, with the 12 SD pairs shown in Table I setting the maximum allowable number of deflections to 3. Error bars for 95% confidence intervals based on Student’s t-distribution are provided for all the simulation results, although in many cases the intervals are too small to be clearly visible. We observe that the OPCA generally slightly overestimates the blocking probability for this example, while the EFPA underestimates it when the traffic is low; however, the OPCA turns to underestimate the blocking probability when the traffic is high. Notice that when the offered load is within 40–50, the results of the EFPA are missing. This is because we cannot achieve a convergence in cases where the offered load to each SD pair draws near the number of channels per trunk, which is equal to 50. It is known that evaluating blocking probability by the EFPA for OBS may fail to converge in certain instances for unprotected deflection routing, as shown in [32].

It is observed from Fig. 3 that in the practical loading range, the EFPA does not perform better than the OPCA. The EFPA is only more accurate than the OPCA when the offered load is within 35–40, where the blocking probability is above 3\times10^{-4}, which is way above what is considered an acceptable grade of service. Notice that these results are consistent with the results in [34] (for cases with trunk reservations), where we observed that the EFPA is more accurate than the OPCA for high loads. Thus we observed in the present example that in the practical traffic loading range, normally used for dimensioning purposes, the longer running time does not enable the EFPA to provide more accurate results than OPCA.

### TABLE I

<table>
<thead>
<tr>
<th>Ingress and Egress SD Pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ingress WA CA1 CA1 CA2 TX GA</td>
</tr>
<tr>
<td>Egress MD IL MA MA CD MA</td>
</tr>
<tr>
<td>Ingress MD IL MA MA CD MA</td>
</tr>
<tr>
<td>Egress WA CA1 CA1 CA2 TX GA</td>
</tr>
</tbody>
</table>

### TABLE II

<table>
<thead>
<tr>
<th>Calculation Task</th>
<th>Running Time of EFPA in Seconds</th>
<th>Running Time of OPCA in Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking probability of the whole network and C = 50</td>
<td>0.271</td>
<td>0.197</td>
</tr>
<tr>
<td>Blocking probability of the whole network and C = 2000</td>
<td>64.45</td>
<td>12.91</td>
</tr>
<tr>
<td>Blocking probability of the whole network and C = 10,000</td>
<td>3006</td>
<td>397</td>
</tr>
<tr>
<td>Blocking probability of the whole network and C = 20,000</td>
<td>13,665</td>
<td>1232</td>
</tr>
</tbody>
</table>
D. OPCA Bounds Behavior

In Fig. 4 we present results for the bounds of the OPCA result iterations as a function of the number of iterations in each layer for different offered loads to each directional SD pair in the NSF network. The maximum number of deflections is set to 3, and each trunk has 50 channels. We observe in Fig. 4(a) that the lower and upper bounds of the OPCA results for overall network blocking probability become closer to each other, and when six iterations are made in each layer, the distance between the lower and upper bounds is less than $10^{-5}\%$ of the lower bound value. Similar results are observed in all four cases of different offered load alternatives presented in Fig. 4. Notice, however, that as the offered load increases, the rate at which the bounds become closer to each other is somewhat reduced. Still, in the cases presented in Figs. 4(a) and 4(b), the distance between the bounds is less than $10^{-5}\%$ of the lower bound value when six iterations per layer have been completed. However, when the offered load is 30 for each SD pair, which is the case presented in Fig. 4(c), seven iterations per layer are required to achieve a distance between the bounds of $10^{-5}\%$ of the lower bound value, and when the offered load increases to 55 [presented in Fig. 4(d)], eight iterations per layer are required to achieve the same accuracy. This is due to the fact that in the NSF network, there are LBTs in each layer. For the LBTs, when the offered load increases in all layers, the first upper bound of trunk blocking probability obtained by the first iteration also increases. This fact can be observed by Eqs. (24) and (53), which show that the trunk blocking probability increases in offered load. Since the first lower

\begin{table}[h]
\centering
\caption{Comparison of the Times Used by the EFPA and OPCA in Each Layer to Calculate the Blocking Probabilities in NSFNET with 10,000 Channels per Trunk}
\label{tab:times_used}
\begin{tabular}{llll}
\hline Algorithm & Layer & Number of & Total Running \noalign{\smallskip}\hline
EFPA & only 1 layer & 78 & 3006 \\
OPCA & layer 0 & 6 & 177.9 \\
 & layer 1 & 5 & 119.7 \\
 & layer 2 & 4 & 99.7 \\
 & layer 3 & 1 & 0.0024 \\
\hline
\end{tabular}
\end{table}

Fig. 3. Blocking probabilities in the NSFNET.

Fig. 4. Bounds of OPCA blocking probabilities in the NSFNET with a different offered load to each directional SD pair.
Fig. 5. Bounds of OPCA blocking probabilities in the NSFNET with a different number of channels per trunk (C), in which the offered load to each directional SD pair is 0.4C.

Fig. 6. Bounds of OPCA blocking probabilities in the NSFNET with a different maximum allowable number of deflections (D), in which the offered load to each directional SD pair is 20 erlangs.
bound is 0 for all LBTs, and the first upper bound increases as the traffic increases, the distance between the first upper and lower bounds of trunk blocking probability obtained by the first iteration is larger when the offered load is larger. These general assertions are consistent with observations in [34]. We also observe in Fig. 4 that as the offered load to the network increases, which implies more primary and deflected bursts in the network, it apparently makes it somewhat more difficult for the bounds to become closer to each other.

1) Effect of the Number of Channels per Trunk: Figure 5 shows the bounds of the OPCA blocking probability results for the NSFNET considering four scenarios where in each scenario there are different channels per trunk. In all the scenarios, the offered load to each directional SD pair is 0.4C, where C is the number of channels per trunk and the maximum number of deflections is set to 3. We observe from the figure that when the number of channels per trunk increases, the lower and upper bounds become closer to each other faster. In particular, when there are 20 channels per trunk, six iterations per layer are required to achieve a distance between the lower and upper bounds of around 10^{-6}\% of the lower bound value, but when there are 100 channels per trunk, in five iterations per layer we achieve a much lower distance between the bounds, namely, approximately 10^{-8}\% of the lower bound value. The results are related to the fact that with a larger number of channels per trunk, the variance of the number of busy channels is lower, which implies less deflected bursts in the networks, as we have already observed. This makes it easier for the bounds to become closer to each other.

2) Effect of the Maximum Allowable Number of Deflections: Figure 6 shows the bounds of the OPCA blocking probability results for the NSFNET considering four scenarios where in each scenario we set a different value for the maximum number of deflections (D). In all scenarios, the offered load to each directional SD pair is 20 erlangs, and each trunk has 50 channels. We observe from the figure that when D increases, the lower and upper bounds become closer to each other slightly more slowly, since the overflowed traffic increases with an increasing D. However, this effect does not seem to be significant when D ≥ 2, because the traffic in layers k (for k ≥ 2) is very small and its effect on the end-to-end blocking probability is negligible. In particular, when D = 0 and six iterations per layer are made, the distance between the lower and upper bounds is around 10^{-6}\% of the lower bound value, but when D = 2 and D = 3, in six iterations per layer the distance between the lower and upper bounds is only around 10^{-6}\% of the lower bound value.

VI. Conclusions

In this paper, we have presented the bounds of the OPCA for blocking probability approximation in OBS networks with deflection routing. The bounds obtained by the iterations of the OPCA algorithm consistently become closer to each other, and after a small number of steps yield a satisfactory blocking probability approximation. We have observed that the speed of the bounds moving closer decreases when the proportion of the overflowed traffic increases, due to the growth of the offered load or the maximum allowable number of deflections, as well as the reduction of the number of channels per trunk. We have also demonstrated in the case of NSFNET with 50 channels per trunk that the OPCA is faster and at least as accurate as the EFPA.

APPENDIX A

Let F(x) = xP_k(x), where P_k(x) = (x^k/k!)/(x_{k=0}^\infty x^k/k!) for positive integer k and x ≥ 0. The meaning of P_k(x) is the loss probability in the M/G/k/k queueing system with the offered traffic load x. P_k(x) is an increasing function in x.

The challenge is to prove that

Q(x, a) = \frac{F(x + a) - F(x)}{a} \quad (A1)

is an increasing function in a.

We can write Eq. (A1) as

Q(x, a) = \frac{F(x + a) - F(x)}{a} = \frac{1}{a} \int_x^{x+a} F(y) dy. \quad (A2)

The function F(x) = xP_k(x) is a convex function (for direct proof, see, e.g., [44,45]). Using the first mean value theorem for integration, we have

Q(x, a) = F'(x + \theta a),

where 0 < \theta < 1.

Using the fact that F'(x) is a convex function, we have the following properties:

1) Q(x, a) increases when x increases.

2) Let a_1 < a_2. Then, because F'(x) is an increasing function, we have the inequality a_1 \theta_1 \leq a_2 \theta_2, where \theta_1 and \theta_2 are the constants belonging to the interval (0,1), for which Q(x, a_1) = F'(x + \theta_1 a_1) and Q(x, a_2) = F'(x + \theta_2 a_2) (as shown in Fig. 7).

ACKNOWLEDGMENTS

The work on this paper was done when V. Abramov was with the Department of Electronic Engineering, City University of Hong Kong.
REFERENCES


Shuo Li received a B.Sc. degree in electronic and communication engineering from City University of Hong Kong, Hong Kong, in 2009. She is currently working toward a Ph.D. degree in the Department of Electronic Engineering at City University of Hong Kong. Her research interests are performance evaluation of telecommunication networks.

Meiqian Wang received her bachelor’s degree and master’s degree in the Department of Electronic Engineering at Harbin Institute of Technology in 2006 and 2009, respectively. She is now working toward a Ph.D. degree at City University of Hong Kong. Her research interest lies in performance evaluation in circuit switching and burst switching networks.

Eric W. M. Wong (S’87–M’90–SM’00) received B.Sc. and M. Phil. degrees in electronic engineering from the Chinese University of Hong Kong, Hong Kong, in 1988 and 1990, respectively, and a Ph.D. degree in electrical and computer engineering from the University of Massachusetts, Amherst, in 1994. In 1994, he joined the City University of Hong Kong, where he is now an Associate Professor with the Department of Electronic Engineering. His research interests include the analysis and design of telecommunications networks, optical switching, and video-on-demand systems. His most notable research work involved the first accurate and workable model for state-dependent dynamic routing based on the Erlang fixed-point approximation, presented at INFOCOM ’90. Since 1991, the model has been used by AT&T to design and dimension its telephone network that uses real-time network routing.

Vyacheslav Abramov graduated from Tajik State University (former USSR) in 1977. During 1977–1992 he worked at the Research Institute of Economics under the State Planning Committee of Tajikistan. In 1992 he repatriated to Israel, where he worked in different institutions and software companies. He received a Ph.D. degree in mathematics from Tel Aviv University in 2005. During 2005–2011 he worked at the School of Mathematical Sciences in Monash University (Australia). During 2011–2012 he worked at the Department of Electronic Engineering in City University of Hong Kong as a Senior Research Fellow. Since August 2012 he has been working at the Center of Advanced Internet Architectures (CAIA) in Swinburne University of Technology. He is the author of two books and about 40 research papers in the areas of applied probability, statistics, queuing theory, and financial engineering.

Moshe Zukerman (M’87–SM’91–F’07) received a B.Sc. degree in industrial engineering and management and a M.Sc. degree in operations research from the Technion—Israel Institute of Technology, Haifa, Israel, and a Ph.D. degree in engineering from the University of California, Los Angeles, in 1985. He was an Independent Consultant with the IRI Corporation and a Postdoctoral Fellow with the University of California, Los Angeles, in 1985–1986. In 1986–1997, he was with the Telstra Research Laboratories (TRL), first as a Research Engineer and, in 1988–1997, as a Project Leader. He also taught and supervised graduate students at Monash University in 1990–2001. During 1997–2008, he was with the University of Melbourne, Victoria, Australia. In 2008 he joined City University of Hong Kong as a Chair Professor of Information Engineering and a group leader. He has served on various editorial boards such as Computer Networks, IEEE Communications Magazine, IEEE Journal of Selected Areas in Communications, IEEE Transactions on Networking, and the International Journal of Communication Systems.