A HEURISTIC ALGORITHM FOR CAPACITY SIZING OF FIXED ROUTING MULTISERVICE ERLANG LOSS NETWORKS

Zdenko Vrdoljak and Mladen Kos
University of Zagreb,
Faculty of Electrical Engineering and Computing,
Unska 3, HR-10000 Zagreb, Croatia
E-mail: {zdenko, kos}@tel.fer.hr

KEYWORDS
Capacity sizing, multiservice loss network, reduced load approximation.

ABSTRACT

This paper deals with optimum link capacity sizing in Erlang multiservice loss networks with fixed routing. The objective function is link capacity installation cost which is to be minimized subject to call blocking probability limits. We describe a new heuristic algorithm we use to find a close match solution to this optimization problem. The algorithm is based on knapsack reduced load approximation.

INTRODUCTION

Multiservice Erlang loss networks are the model most frequently used to describe connection level traffic in connection oriented networks. An introduction to loss networks theory can be found in (Kelly, 1991).

The loss network considered in this paper consists of J links, each of capacity C_j, $j = 1, \ldots, J$. One can identify N different classes of calls, each belonging to a pair of network nodes. The routing in the network is fixed and each call of class i is associated with a single route. A route is a set of links connecting source to destination node(s) and is represented by an array of link indexes forming the route, $R_i = \{j_1, j_2, \ldots, j_{|R_i|}\}$. $|R_i|$ is the route length. The stochastic process describing the call arrival process for call class i is a Poisson process with intensity λ_i, $i = 1, \ldots, N$. The holding time of a call of class i has a general distribution with mean $1/\mu_i$. $a_i = \lambda_i/\mu_i$ is the offered load for class i. A call of class i requires bandwidth B_{ij} on link j. The set of all classes with routes passing through link j is denoted $\Gamma_j = \{\ i\ :\ j \in R_i\ \}$.

The resource sharing policy in the network is complete sharing (CS). A call of class k is accepted if there are enough resources on all links on route:

$$\sum_{i=1}^{N} B_{ij} n_i \leq C_j - b_k, \quad \forall j \in R_k,$$

where n_i is number of class i calls established on route R_i. It is possible to consider other sharing policies, especially those of threshold type. In this case a single link with its specific sharing policy is replaced by an equivalent subnetwork of completely shared links. Capacity sizing in such network includes calculation of optimal sharing policy parameters for all links and is out of scope of this paper.

The state of the network is described by a vector $\mathbf{n} = [n_1, n_2, \ldots, n_N]$, where n_i is the number of class i calls (established on route R_i). State space Ω of the fixed routing network can be defined as:

$$\Omega = \left\{ \ \mathbf{n}\ :\ \sum_{i=1}^{N} B_{ij} n_i \leq C_j, \quad \forall j = 1, \ldots, J \ \right\}.$$

By setting a multidimensional Markov chain model for such a network, one can write an array of local balance equations which concatenated yield a product form solution for the stationary point distribution of vector $\mathbf{n}$:

$$\boldsymbol{\pi}(\mathbf{n}) = G^{-1} \times \prod_{i=1}^{N} \frac{a_i^{n_i}}{n_i!}, \tag{1}$$

where G is the normalization constant:

$$G = \sum_{\mathbf{n} \in \Omega} \prod_{i=1}^{N} \frac{a_i^{n_i}}{n_i!}.$$

Deduction of this result for a less general case can be found in (Kaufman, 1981).

Let Ω_i^+ be the set of blocking states for flow i. The blocking probability for class i is the sum of the stationary point probabilities of states in set Ω_i^+:

$$P_{B_i} = \sum_{\mathbf{n} \in \Omega_i^+} \boldsymbol{\pi}(\mathbf{n}).$$

In this paper we try to answer the question: what is the minimum capacities configuration $\mathbf{C} = [C_1, \ldots, C_J]$ so that blocking probabilities $P_{B_i} \leq \bar{P}_{B_i}$, where $\bar{P}_{B_i}$, $i = 1, \ldots, N$, are predefined upper limits. In order to find a solution to this problem, we first review reduced load approximation. We admit that knapsack approximation is the most suitable approximation and define the capacity sizing optimization problem based on this approximation. We introduce two simple algorithms for fast capacity sizing of single resources based on the Kaufman-Roberts recursion formula and uniform asymptotic approximation (UAA)

Proceedings 12th International Conference ASMTA 2005
Khalid Al-Begain, Gunter Bolch, Miklos Telek © ECMS, 2005
ISBN 1-84233-112-4 (Set) / ISBN 1-84233-113-2 (CD)

and present our simple heuristic algorithm based on two key properties. Finally, we apply our algorithm to a concrete problem and test its accuracy.

REDUCED LOAD APPROXIMATIONS

The largest problem with the application of the product form solution to blocking probabilities (1) is the inability to evaluate constant G in an acceptable time. The problem complexity can be partially reduced by applying multidimensional recursions and decomposition of Dziong and Roberts and the improved recursion presented in (Conway et al., 1994). However, the complexity is still too high and approximation is needed.

Reduced load approximations are the most popular approximations for blocking probabilities in loss networks. The first approximation was introduced by Whitt in (Whitt, 1985) for single service networks. The theory was later extended by work of Kelly, and Chung and Ross. The reader is advised to consult deduction in (Kelly, 1991) for a full theoretical background of the theory.

The main idea of reduced load theory is that the end-to-end call blocking for class i for networks with large capacities and offered loads can be well approximated by

$$P_{B_i} \approx 1 - \prod_{j \in R_i} \theta_{ij}, \tag{2}$$

where θ_{ij} is the probability that class i call will be accepted on link j. In other words, (2) says that call blocking on different links is approximately independent. Reduced load schemes differ in the way θ_{ij} is calculated.

Kelly's Approximation

Kelly suggests that θ_{ij} be approximated by $(1 - L_j)^{B_{ij}}$, that is

$$P_{B_i} \approx 1 - \prod_{j \in R_i} (1 - L_j)^{B_{ij}}. \tag{3}$$

L_j is determined by the Erlang loss formula

$$L_j = E_B\left[C_j; A_j\right], \tag{4}$$

where A_j is the reduced load offered to link j:

$$A_j = \frac{1}{1 - L_j} \sum_{i \in \Gamma_j} B_{ij} a_i \prod_{j \in R_j} (1 - L_j)^{B_{ij}}. \tag{5}$$

Note that (4) and (5) represent mapping of convex set $[0,1]^J$ onto itself. According to Brouwer's theorem there is at least one fixed point solution of the mapping. Kelly in (Kelly, 1991) proves that there is a unique fixed point solution to (4) and (5) and it can be found by repeated substitutions (Gauss-Seidel method) starting from any value of $0 \leq L_j \leq 1$. The procedure usually converges after a few iterations. After obtaining all L_j, $j = 1, \ldots, J$, blocking probabilities are determined by (3).

Knapsack Approximation

Knapsack approximation introduced in (Chung and Ross, 1993) is the most intuitive and the most accurate reduced load approximation. The blocking probability is calculated according to

$$P_{B_i} = 1 - \prod_{j \in R_i} (1 - L_{ij}), \tag{6}$$

where L_{ij} is the blocking probability of a class i call on link j. L_{ij} is determined using Kaufman-Roberts recursion (Kaufman, 1981). The traffic of classes in set Γ_j is offered to a single resource of capacity C_j. However, offered traffic is reduced by the amount blocked on all other links in the network, except the observed link j. Let $q(n)$ satisfy the following recursion:

$$q(n) = \frac{1}{n} \sum_{i \in \Gamma_j} B_{ij} A_{ij} q(n - B_i j), \; n = 0, \ldots, C_j$$

$$\sum_{n=0}^{C_j} q(n) = 1, \quad q(n) = 0, \quad \text{for} \quad n < 0.$$

A_i is the reduced load of class i:

$$A_{ij} = a_i \cdot \prod_{l \in R_i - \{j\}} (1 - L_{il}). \tag{7}$$

Blocking probability L_{ij} is determined by

$$L_{ij} = \sum_{n = C_j - B_{ij} + 1}^{C_j} q(n). \tag{8}$$

Blocking probabilities L_{ij} are found by repeated substitutions of (7) and (8). End-to-end blocking probabilities are determined by (6). Even though knapsack approximation is asymptotically correct just as Kelly's approximation, knapsack approximation does not necessarily have a unique fixed point (see (Chung and Ross, 1993)).

Pascal Approximation

The Pascal approximation is a natural relaxation of the knapsack approximation. The knapsack approximation describes single completely shared resources with multidimensional Markov chains to which Kaufman-Roberts recursion can be applied. The Pascal approximation uses a stationary birth-death process m to describe the occupied capacity of link j with capacity C_j. The birth rate is $\xi_j^2/\sigma_j^2 + m(1 - \xi_j/\sigma_j^2)$

and the death rate is m. Approximation of multidimensional Markov chains by the birth-death process is obtained by equalizing mean and variance for the case when $C_j = \infty$. As shown by Chung and Ross in (Chung and Ross, 1993), Pascal approximation is asymptotically correct, but may have more than one fixed point.

Numerical testing of approximations

The approximations are compared for the example network from (Chung and Ross, 1993) shown in Figure 1. The network has four edge and one backbone

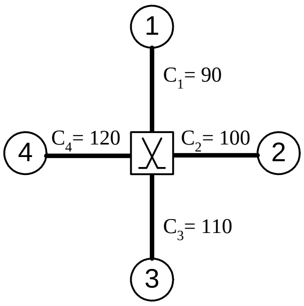

Figure 1: Example Network for Comparison of Reduced Load Approximations

node. The link capacities are $C_1 = 90$, $C_2 = 100$, $C_3 = 110$ and $C_4 = 120$. The network provides two services, S_1 with bandwidth requirement $B_{ij} = 1$, and S_2 with bandwidth requirement $B_{ij} = 5$ in both directions on all links. There are 12 classes and 6 routes: $(1, 2)$, $(1, 3)$, $(1, 4)$, $(2, 3)$, $(2, 4)$ and $(3, 4)$, one S_1 and one S_2 class per one route. The offered traffics for S_1 classes are 8 and for S_2 classes 1.5 on all routes. The offered traffic is increased from factor 1 to 2.2 in steps of 0.15.

The blocking probability for each class is first exactly calculated using recursion (Conway et al., 1994) and then approximated by (1) The Kelly approximation, (2) the knapsack approximation, (3) the knapsack approximation using the Labourdette-Hart approximation for single resource blocking probability (Labourdette and Hart, 1992) instead of Kaufman-Roberts recursion, (4) the knapsack approximation using the UAA approximation (Mitra and Morrison, 1994) for single resource blocking probability instead of Kaufman-Roberts recursion and (5) the Pascal approximation. Reduced load approximations (3) and (4) only take advantage of numerical approximation to blocking probabilities.

Due to a large amount of numerical data, we present results for path $(3, 4)$ for both services because deviations are the most evident. Graph labels in figures 2 and 3 correspond to labels introduced before. It is evident that knapsack approxima-

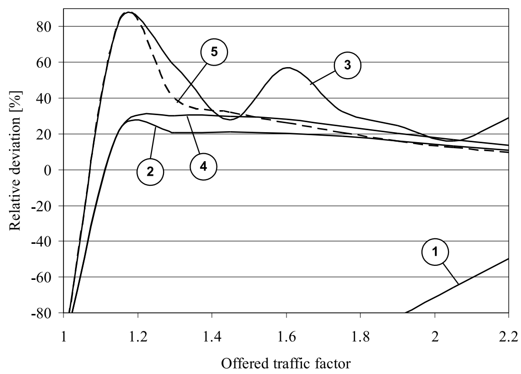

Figure 2: Relative Deviation for Service S_1 on $(3, 4)$

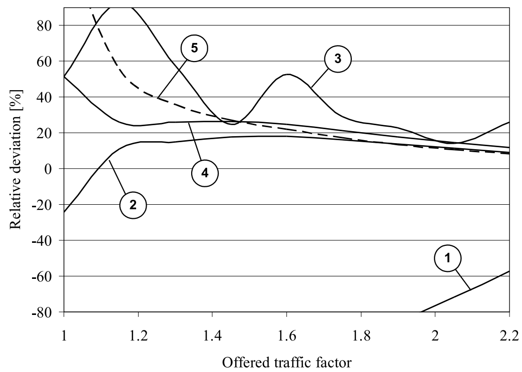

Figure 3: Relative Deviation for Service S_2 on $(3, 4)$

tion schemes (2) and (4) obtain the lowest deviation from exact values of blocking probabilities, with pure knapsack approximation being slightly better.

THE OPTIMIZATION PROBLEM

Let $P_{B_i}(\mathbf{C})$ be the blocking probability for class i calls obtained using the knapsack reduced load approximation for a network with capacity vector $\mathbf{C} = [C_1, \ldots, C_N]$. Let ξ_j be the installation cost per capacity unit of link j. Capacity sizing in this paper can be described with the following optimization problem:

$$\begin{aligned} \text{Minimize :} \quad & f(\mathbf{C}) = \sum_{j=1}^{J} \xi_j C_j, \\ \text{Subject to :} \quad & P_{B_i}(\mathbf{C}) \leq \bar{P}_{B_i}, \quad \forall\, i = 1, \ldots, N, \end{aligned} \tag{9}$$

where $\bar{P}_{B_i}$, $i = 1, \ldots, N$, are the maximum allowed blocking probabilities and $\mathbf{C} \in (\mathbb{N} \cup 0)^J$.

Define Ψ to be the set of all capacity configurations that produce $P_{B_i}(\mathbf{C}) \leq \bar{P}_{B_i}$:

$$\Psi = \left\{ \mathbf{C} \;\; : \;\; P_{B_i}(\mathbf{C}) \leq \bar{P}_{B_i} \right\}.$$

The main relaxation we introduce in this paper is that Ψ is a convex set, i.e. there is a convex set $\tilde{\Psi}$ in $\mathbb{R}^J$ enveloping Ψ with all edge points in Ψ corresponding to the subset of edge points in $\tilde{\Psi}$. This relaxation might be true, however authors failed to deduce a convincing proof.

The relaxation is mainly motivated by numerical examples. For instance, consider the network shown in figure 4. Each flow shown in the figure 4 is com-

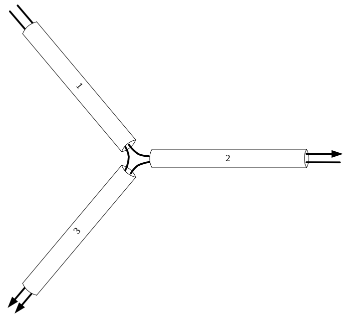

Figure 4: Example Network for Testing Ψ Convexity

prised of three classes each reserving $B_1 = 1$, $B_2 = 8$, $B_3 = 5$ on all links on route. Traffic intensities of classes are $a_1 = 10$, $a_2 = 100$ and $a_3 = 50$. By applying the knapsack approximation we created a 3D representation of the edge of set Ψ. It is shown in figure 5.

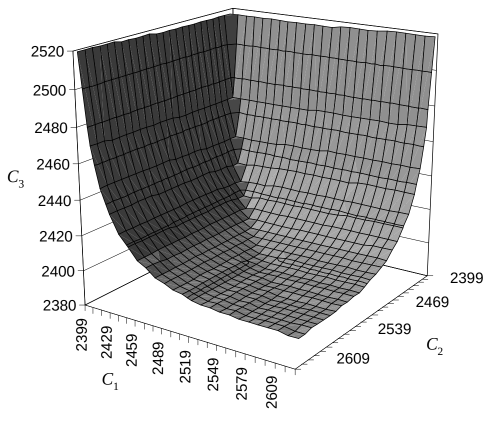

Figure 5: Edge of Convex Set Ψ

SOLUTION TO THE OPTIMIZATION PROBLEM FOR A SINGLE RESOURCE

Optimization problem (9) can be easily solved for a single link (resource). We observe a single completely shared Erlang resource of capacity C with k services (classes) requiring $b_1, \ldots, b_k$ units of resource per call. Traffic intensities are $a_1, \ldots, a_k$. We are interested in calculating the minimum capacity C so that all blocking probabilities P_{B_i} are less than or equal to the predefined upper limits $\bar{P}_{B_i}$, $i = 1, \ldots, k$. The equivalent optimization problem is

$$\begin{array}{rl} \text{Minimize}: & C, \\ \text{Subject to}: & P_{B_i}(C) \leq \bar{P}_{B_i}, \quad \forall\, i = 1, \ldots, k. \end{array} \tag{10}$$

Consider Kaufman-Roberts recursion (Kaufman, 1981), (Morrison, 1981) for blocking probabilities P_{B_i}:

$$q(n) = \frac{1}{n}\sum_{i=1}^{k} a_i b_i q(n - b_i), \quad \sum_{n=0}^{C} q(n) = 1,$$

where $q(n) = 0$ for $n < 0$.

$$P_{B_i} = \sum_{n=C-b_i+1}^{C} q(n).$$

Classes may have different values for $\bar{P}_{B_i}$. Each class has its own minimum C^i for which $P_{B_i} \leq \bar{P}_{B_i}$. The optimum capacity is $C = \max_i\{C^i\}$. In order to find C, one can start from a value of $C = 0$, increase it in each step by 1 and for each C calculate values $q(0)$ to $q(C)$. When the sum of the last b_i elements of q for each class becomes less than or equal to $\bar{P}_{B_i}$, one stops the procedure and the current C is the solution to (10).

The complexity of the algorithm can be reduced to $O(kC)$ by taking into account recursive relation. Namely, $q(C)$ can be calculated in each step with complexity $O(k)$. One maintains a vector $\hat{q}$ and normalization variable Σ with $\hat{q}(0) = 1$ and $\hat{q}(n) = 0$ for $n < 0$. $\hat{q}$ is obtained using Kaufman-Roberts recursion in each step and Σ by summing the old value of Σ with $\hat{q}(C)$. $q(C) = \hat{q}(C)/\Sigma$. The blocking probabilities P_{B_i}, $i = 1, \ldots, k$ in each step are calculated by adding $(\hat{q}(C) - \hat{q}(C - b_i))$ to a variable h_i that equals the sum of $\hat{q}(C-1)$ and $\hat{q}(C - b_i - 1)$ from the previous step. The blocking probabilities are calculated as $P_{B_i} = h_i/\Sigma$. The whole algorithm has complexity $O(kC)$, where C is the solution to (10). It is shown in figure 6.

Even though the recursion inversion algorithm obtains a solution in very short time, the time may not always be short enough for fast applications of the algorithm within other algorithms. This is why we developed a simple algorithm which obtains approximate solution to (10) based on the numerical inversion of the uniform asymptotic approximation (UAA) for Engset's model introduced in (Mitra and Morrison, 1994).

Let z^* be the solution to equation

$$\sum_{i=1}^{k} a_i b_i z^{b_i} = C.$$

$C = 0$, $h_i = 0$, $i = 1, \ldots, k$,
$\hat{q}(x) = 0$ for $x < 0$, $\hat{q}(0) = 1$, $\Sigma = 0$;

1. $\hat{q}(C) = \dfrac{1}{C} \sum_{i=1}^{k} a_i b_i \hat{q}(C - b_i)$;
2. $\Sigma = \Sigma + \hat{q}(C)$;
3. $h_i = h_i + [q(C) - q(C - b_i)]$, $i = 1, \ldots, k$;
4. $P_{B_i} = h_i / \Sigma$;
5. if $P_{B_i} \leq \bar{P}_{B_i}$, $i = 1, \ldots, k$, STOP, C is solution to (10);
6. $C = C + 1$, go to step 1.

Figure 6: Kaufman-Roberts inversion algorithm.

z^* is a so called saddle point. Define

$$f(z) = \sum_{i=1}^{k} \left\{ a_i \left(z^{b_i} - 1 \right) / C \right\} - \ln(z),$$
$$v(z) = \sum_{i=1}^{k} a_i b_i^2 z^{b_i} / C.$$

According to the UAA, the blocking probability for class i is approximately

$$P_{B_i} \approx \frac{e^{C \cdot f(z^*)} \left[1 - (z^*)^{b_i} \right]}{\sqrt{2\pi C v(z^*)} (1 - z^*) M}, \qquad (11)$$

where

$$M = \frac{1}{2} \mathrm{Erfc} \left[\mathrm{sgn}(1 - z^*) \sqrt{-C f(z^*)} \right] + \\ + \frac{e^{C f(z^*)}}{\sqrt{2\pi C}} \cdot \left\{ \frac{1}{\sqrt{v(z^*)}(1 - z^*)} - \frac{\mathrm{sgn}(1 - z^*)}{\sqrt{-2 f(z^*)}} \right\}.$$

The uniform asymptotic approximation has excellent performance. In order to demonstrate it, we constructed an example with four classes with bandwidth requirements and offered traffics as follows: $b_1 = 23$, $b_2 = 20$, $b_3 = 7$, $b_4 = 9$, $a_1 = 12$, $a_2 = 16$, $a_3 = 9$ and $a_4 = 20$. The largest deviation of UAA from exact values, obtained by Kaufman-Roberts recursion, is the one of class 3 because it has the smallest bandwidth requirement and offered traffic. Figure 7 shows the exact values of blocking probability on a logarithmic scale for class 3, and figure 8 shows the relative deviation of UAA from exact values for class 3. Relative deviation was lower than 1 percent in most of the numerical examples we conducted with blocking probabilities less than 5%. This makes UAA quite suitable for creating algorithms for fast capacity sizing.

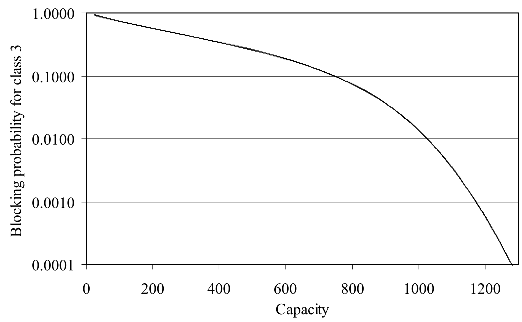

Figure 7: Exact Blocking Probability for Class 3 calls

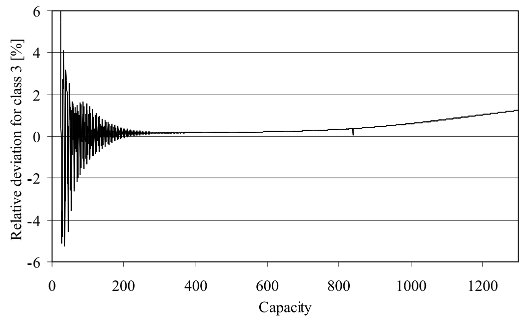

Figure 8: Relative Deviation from Exact Blocking Probability for Class 3 calls.

Even though UAA approximation is continuous and some classical inversion methods such as the iteration of the Newton tangent method may seem a reasonable choice, that is not possible due to two reasons: it is very complicated to find the derivative $\partial P_{B_i}(C)/\partial C$ since z^* can only be found numerically and $P_{B_i}(C)$ is not monotonic. This is why we developed a procedure that uses a combination of linear and binary search on all $P_{B_i}(C)$.

The optimum capacity is searched for all class simultaneously. The search is conducted in two phases. During the first phase, a linear search finds the window (L_i, R_i), $R_i = L_i + E$, inside which it is guaranteed there is a minimum capacity C^i for which $P_{B_i} \leq \bar{P}_{B_i}$. E is the mean occupancy of an optimally sized resource: $E = \sum_{i=1}^{k} a_i b_i \left(1 - \bar{P}_{B_i} \right)$. After all the windows are determined, the ones with largest R_i are chosen and binary search is performed inside these windows to find C^i. Binary search narrows the window size $(R_i - L_i)$ until it becomes less than a predefined tolerance τ. The largest value of all R_i is chosen as the result. The algorithm is shown in figure 9. Note the basic difference of the described algorithms. Results of the Kaufman-Roberts inver-

$UAA_i(C)$ equals P_{B_i} in (11) for class i and capacity C.

1. $E = \sum_{i=1}^{k} a_i b_i \left(1 - P_{B_i}\right);$
2. for each class $i = 1, \ldots, k$:
   a) $L_i = R_i, \quad R_i = L_i + E;$
   b) while $UAA_i(L_i) > \bar{P}_{B_i}$ and $UAA_i(R_i) < \bar{P}_{B_i}$
   $L_i = L_i + E, \quad R_i = R_i + E.$
3. $R_{max} = \max_i\{R_i\}, \quad \Phi = \{\ i\ :\ R_i = R_{max}\ \};$
4. for all $i \in \Phi$: while $(R_i - L_i) > \tau$ do
   a) $V = (L_i + R_i)/2;$
   b) if $UAA_i(V) > \bar{P}_{B_i}$, $L_i = V$, else $R_i = V$.
5. Result: $C = \max_{i \in \Phi} \{R_i\}$.

Figure 9: UAA inversion algorithm.

sion algorithm are always in set $\mathbb{N} \cup 0$. Results of the UAA inversion algorithm are in $\mathbb{R}$.

The algorithm obtains satisfactory results. The average deviation from exact values shown in figure 6 is less than 2%. However, the algorithm may obtain large deviations in cases when ratios among $\bar{P}_{B_i}$-s are large. Consider the example of a single resource with two classes. The offered traffic and bandwidth requirement for the first class are $a_1 = 100$, $b_1 = 100$. The traffic and bandwidth requirement for the second class are varied: $b_2/b_1 = 0.1, \ldots, 660$ and $a_2 = 10, \ldots, 1060$. $\bar{P}_{B_1}$ and $\bar{P}_{B_2}$ are also varied over values 10^-2 to 10^-7. For each pair $(\bar{P}_{B_1}, \bar{P}_{B_2})$, the maximum deviation for all values of b_2 and a_2 is determined. These deviations are shown in figure 10. Even though this might discourage the use of the UAA inversion algorithm, one must notice that if ratios of different P_{B_i}-s are less than 100, the maximum deviations are less than 2%. In our experience, the deviation is in most of the cases no greater than 1%.

A HEURISTIC ALGORITHM FOR NETWORK CAPACITY SIZING

The inversion algorithms described in the previous section are the main component of the heuristic al-

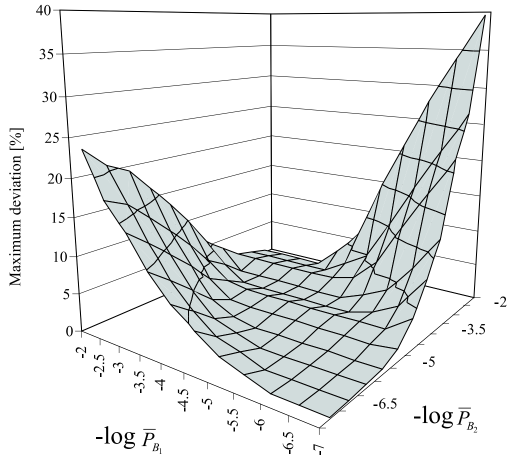

Figure 10: Maximum Deviations of the UAA Inversion Algorithm for Different Values of $\bar{P}_{B_1}$ and $\bar{P}_{B_2}$

gorithm for network capacity sizing. We denote the results of the Kaufman-Roberts and UAA inversion algorithms with $C_j(\mathbf{a}, \mathbf{b}, \bar{\mathbf{P}}_B)$. $\mathbf{a}$, $\mathbf{b}$ and $\bar{\mathbf{P}}_B$ are sets of offered traffics, bandwidth requirements and the maximum allowed blocking probabilities. Furthermore, we will consider set Ψ to be continuous and convex. When using knapsack approximation, all continuous values of vector $\mathbf{C}$ will be rounded up and used as such within the approximation. The heuristic algorithm for network capacity planning is based on two simple properties we prove below.

Theorem 1 (Lower limits of Ψ). *Consider the optimization problem* (9). *Let* $C_j(\mathbf{a}, \mathbf{b}, \bar{\mathbf{P}}_B)$ *be solution to optimization problem* (10). *Denote* Γ_j *to be the set of indexes of classes establishing calls through link* j. *Define sets:*

$$\mathbf{a}_j = \left[\ a_i \left(1 - \bar{P}_{B_i}\right)\ :\ i \in \Gamma_j\ \right],$$
$$\mathbf{b}_j = \left[\ B_{ij}\ :\ i \in \Gamma_j\ \right],$$
$$\bar{\mathbf{P}}_{B_j} = \left[\ P_{B_i}\ :\ i \in \Gamma_j\ \right].$$

If the blocking probability for classes is determined by using knapsack approximation, then for each point $\mathbf{C} = [C_1, \ldots, C_J]$ *the following expression holds:*

$$\tilde{C}_j = C(\mathbf{a}_j, \mathbf{b}_j, \bar{\mathbf{P}}_{B_j}) - 1 \leq C_j, \quad j = 1, \ldots, J.$$

Proof. If some point $\mathbf{C} \in \Psi$, then in a fixed point of the knapsack approximation for each class the following holds:

$$\bar{P}_{B_i} \geq 1 - \prod_{j \in R_i} (1 - L_{ij}).$$

Since the maximum blocking probability for class i is P_{B_i}, it follows that the minimum possible offered traffic of a class i to any link j equals $a_i(1 - \bar{P}_{B_i})$. Let

$$\tilde{C}_j = C(\mathbf{a}_j, \mathbf{b}_j, \bar{\mathbf{P}}_{B_j}) - 1,$$

$j = 1, \ldots, J$. In point $\tilde{\mathbf{C}}$ there is at least one class with $P_{B_i} > \bar{P}_{B_i}$ because there is at least one $L_{ij} > \bar{P}_{B_i}$. In other words, at least one C_j is undersized. Any point $\mathbf{C} = [C_1, \ldots, \tilde{C}_l, \ldots, C_J]$ with $C_i > \tilde{C}_i$, $i \neq l$ is definitely out of Ψ as well. □

Theorem 2 (Interior point of Ψ). *Consider the optimization problem* (9). *Let* $C_j(\mathbf{a}, \mathbf{b}, \bar{\mathbf{P}}_B)$ *be a solution to optimization problem* (10). *Denote* Γ_j *to be the set of indexes of classes establishing calls through link* j. *Define sets:*

$$\begin{aligned}\mathbf{a}_j &= [\ a_i \ :\ i \in \Gamma_j\ ]\,,\\ \mathbf{b}_j &= [\ B_{ij} \ :\ i \in \Gamma_j\ ]\,,\\ \bar{\mathbf{P}}_{B_j} &= \left[\ 1 - \sqrt[|R_i|]{1 - \bar{P}_{B_i}} \ :\ i \in \Gamma_j\ \right].\end{aligned}$$

If the blocking probability for classes is determined by using the knapsack approximation, then point $\mathbf{C}^0$ *with components determined by*

$$\tilde{C}_j^0 = C(\mathbf{a}_j, \mathbf{b}_j, \bar{\mathbf{P}}_{B_j}), \quad j = 1, \ldots, J,$$

is within set Ψ, *i.e.* $\mathbf{C}^0 \in \Psi$.

Proof. If $\mathbf{C}^0 \in \Psi$, then for all $i = 1, \ldots, N$

$$\bar{P}_{B_i} \geq 1 - \prod_{j \in R_i} (1 - L_{ij})\,.$$

Since every $L_{ij} \leq 1 - \sqrt[|R_i|]{1 - \bar{P}_{B_i}}$, because all links are oversized, it follows that:

$$\prod_{j \in R_i} (1 - L_{ij}) \geq \prod_{l \in R_i} \sqrt[|R_i|]{1 - \bar{P}_{B_i}} = 1 - P_{B_i}.$$

Therefore

$$P_{B_i} \geq 1 - \prod_{j \in R_i} (1 - L_{ij})\,,$$

for all $i = 1, \ldots, N$. $\mathbf{C}^0 \in \Psi$. □

The first theorem defines point $\tilde{\mathbf{C}}$ which represents the vertex of the subset of $\mathbb{R}^J$ which completely comprises Ψ. Ψ is surrounded by J planes $\mathbf{C} = [C_1, \ldots, \tilde{C}_j, \ldots, C_N]$, $j = 1, \ldots, J$, crossing at point $\tilde{\mathbf{C}}$. Optimal values of $C_j \geq \tilde{C}_j$ for all j. Coordinate planes in figure 5 are chosen according to $\tilde{\mathbf{C}}$.

The second theorem defines point $\mathbf{C}^0$ that is certainly inside of Ψ, but relatively close to the optimum. $\mathbf{C}^0$ with gradient vector $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_J]$ defines a hyperplane which together with J planes defined by point $\tilde{\mathbf{C}}$ forms a subset of $\mathbb{R}^J$ which certainly contains a solution to optimization problem (9). These two theorems are the basis for building the heuristic algorithm for network capacity sizing.

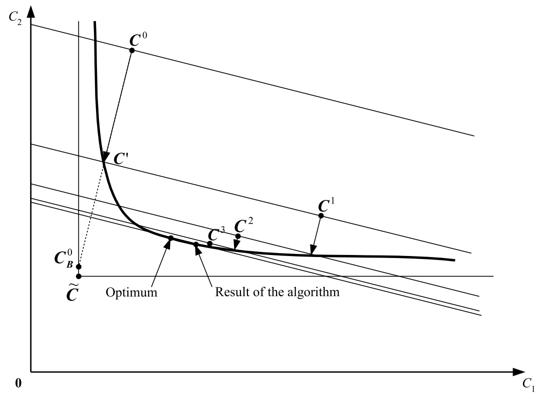

Figure 11: Graphical Representation of Network Capacity Sizing Heuristic Algorithm

The idea of the algorithm is depicted in figure 11 for the two-dimensional case. We define the cost function at point $\mathbf{C}$ as:

$$\Theta(\mathbf{C}) = \sum_{j=1}^{J} \xi_j \cdot C_j.$$

Furthermore, we define $\Theta^n = \Theta(\mathbf{C}^n)$. The idea of the heuristic algorithm is to construct a point $\mathbf{C}^{n+1}$ based on point $\mathbf{C}^n$ (both in Ψ) so that $\Theta^{n+1} < \Theta^n$ in each step until certain criteria is reached. The algorithm starts from point $\mathbf{C}^0$ calculated according to theorem 1. Each cost $\Theta^n = \Theta(\mathbf{C}^n)$ defines a set

$$H^n = \left\{\ \mathbf{C}\ :\ \Theta^n = \sum_{j=1}^{J} \xi_j \cdot C_j,\ \text{and } \mathbf{C} \in \Psi\ \right\}.$$

Point $\mathbf{C}^n$, as well as any other point $\mathbf{C} \in H^n$, has an adequate point on the edge of Ψ that lies on the line defined by $\mathbf{C}^n$ and the negative gradient of objective function: $-\boldsymbol{\xi} = -[\xi_1, \ldots, \xi_J]$. Point $\mathbf{C}'$ is closer to the optimum only if $\mathbf{C}^n$ is not an edge point of Ψ, because it is not possible to advance further in the direction of the negative gradient from $\mathbf{C}^n$. Therefore, each $\mathbf{C}^n$ should be an non-edge point of Ψ (and of H^n).

Starting from $\mathbf{C}^n$, one must first find corresponding $\mathbf{C}'$. $\mathbf{C}'$ lies somewhere on the line determined by the parametric equation

$$\mathbf{C}(t) = \mathbf{C}^n - t \cdot \boldsymbol{\xi}.$$

This line crosses the planes determined by point $\tilde{\mathbf{C}}$ at point $\mathbf{C}_B^n$. We define

$$t^n = \min_j \left\{ \frac{C_j^n - \tilde{C}_j}{\xi_j} \right\}.$$

Point $\mathbf{C}_B^n$ equals

$$\mathbf{C}_B^n = \left[\ C_j^n + t^n \xi_j \ :\ j = 1, \ldots, J\ \right].$$

Point $\mathbf{C}'$ is placed somewhere on line $(\mathbf{C}_B^n, \mathbf{C}^n)$. Since for each point $\mathbf{C}$ on the line between $\mathbf{C}_B^n$ and $\mathbf{C}^n$ one can determine if it belongs to Ψ or not by using the knapsack approximation. $\mathbf{C}'$ can be determined using binary search on set $\{ \ \mathbf{C} = \mathbf{C}^n - t \cdot \boldsymbol{\xi} \ : \ 0 \leq t \leq t^n \ \}$. Binary search is finished when the search window is smaller than the defined tolerance.

When $\mathbf{C}'$ is determined, $\Theta^{n+1} = \Theta(\mathbf{C}')$. One must find $\mathbf{C}^{n+1} \in H^{n+1}$. Within this work several methods for determining $\mathbf{C}^{n+1}$ based on Θ^{n+1} were considered. The problem is the fact that the edge of H^{n+1} is very difficult to find in reasonable time. The following method yields satisfactory results with minimum numerical complexity.

Define new variables ΔC_j:

$$\Delta C_j = C_j - \tilde{C}_j. \tag{12}$$

Hyperplane H^n can now be defined in terms of $\Delta\mathbf{C}$:

$$\hat{\Theta}^n = \Theta^n - \sum_{j=1}^{J} \xi_j C_j = \sum_{j=1}^{J} \xi_j \Delta C_j.$$

The next point to be chosen $\mathbf{C}^{n+1}$ corresponds to point $\Delta\mathbf{C}^{n+1} = \tilde{\mathbf{C}} - \mathbf{C}^{n+1}$. $\Delta\mathbf{C}^{n+1}$ in our algorithm is chosen as the central point of the part of hyperplane H^{n+1} that is in the first ortant of the ΔC_j coordinate system:

$$\Delta\mathbf{C}^{n+1} = \left[\ \Delta C_j^n = \frac{\hat{\Theta}^{n+1}}{J\xi_j} \ : \ j = 1, \ldots, J \ \right].$$

By use of transformation (12) one obtains the following expression for point $\mathbf{C}^{n+1}$:

$$\mathbf{C}^{n+1} = \left[\ C_j^n = \frac{\Theta^n - \sum_{i=1}^{J} \xi_i \tilde{C}_i}{J\xi_j} + \tilde{C}_j : j = 1, \ldots, J \ \right].$$

Points in figure 11 are chosen according to this method. When the new point is determined, all of its components are rounded up because the knapsack algorithm does not work with real numbers. One must notice that $\mathbf{C}^{n+1}$ may easily fall out of Ψ. If this happens, the algorithm stops and the solution of the algorithm is $\mathbf{C}^n$. The other way to determine the end of the algorithm is if the old and new points are equal after rounding up. The whole algorithm is shown in figure 12.

The algorithm uses the inversion algorithms discussed in the previous section only during the first two steps. This is quite useful when UAA inversion is considered, due to its deviation. The algorithm will not reach the optimum, at least not often, but will get quite close to it. Due to this, it is necessary to calculate the deviation bound. The best estimate is the one based on cost. From theorem 1 it is clear that cost at point $\tilde{\mathbf{C}}$ is unreachable since it represents the smallest possible cost. The deviation from the optimum is therefore less than the difference of costs at the result point of the algorithm and $\tilde{\mathbf{C}}$. The deviation estimate is therefore:

$$\Delta_{dev} \leq \frac{\Theta(\mathbf{C}') - \Theta(\tilde{\mathbf{C}})}{\Theta(\tilde{\mathbf{C}})} \cdot 100\,[\%]\,. \tag{13}$$

1. Calculate $\tilde{\mathbf{C}}$ according to theorem 1;
2. Calculate $\mathbf{C}^0$ according to theorem 2; n=0;
3. $t^n = \min_j \left\{ \dfrac{C_j^n - \tilde{C}_j}{\xi_j} \right\}$;
4. $\overline{t} = 0$, $\underline{t} = t^n$, $\overline{\mathbf{C}} = \mathbf{C}^n$, $\underline{\mathbf{C}} = \overline{\mathbf{C}} - \underline{t} \cdot \boldsymbol{\xi}$;
5. $t' = (\overline{t} + \underline{t})/2$, $\mathbf{C}' = \mathbf{C}^n - t' \cdot \boldsymbol{\xi}$;
6. If $\mathbf{C}' \in \Psi$, $\underline{\mathbf{C}} = \mathbf{C}'$, $\underline{t} = t'$, else $\overline{\mathbf{C}} = \mathbf{C}'$, $\overline{t} = t'$;
7. If $\left|\overline{\mathbf{C}} - \mathbf{C}\right| < d$, $\mathbf{C}' = \overline{\mathbf{C}}$, go to 9.;
8. Go to 5.;
9. $\Theta' = \sum_{j=1}^{J} \xi_j C_j'$, $n = n + 1$;
10. $\mathbf{C}^n = \left[C_j^n = \dfrac{\Theta' - \sum_{i=1}^{J} \xi_i \tilde{C}_i}{J\xi_j} + \tilde{C}_j \ : \ j = 1, \ldots, J \ \right]$;
11. if $\lceil \mathbf{C}^n \rceil \notin \Psi$ or $\lceil \mathbf{C}' \rceil = \lceil \mathbf{C}^n \rceil$, go to 12, else go to 3;
12. Result: $\mathbf{C}'$.

Figure 12: Network Capacity Sizing Heuristic Algorithm

CAPACITY SIZING EXAMPLES

In order to demonstrate the heuristic algorithms performance, we tested the algorithms on a network with 8 nodes and 10 edges shown in figure 13. The network provides for three services establishing bidirectional symmetric calls. Each service reserves the same bandwidth on all links on route. Routes and offered traffics are listed in table 1. The offered traffic for each pair represents the sum of the offered traffics in both directions. Routes and offered traffics are equal for all three services. Bandwidth requirements are: $b_1 = 1$, $b_2 = 7$ and $b_3 = 19$. The maximum

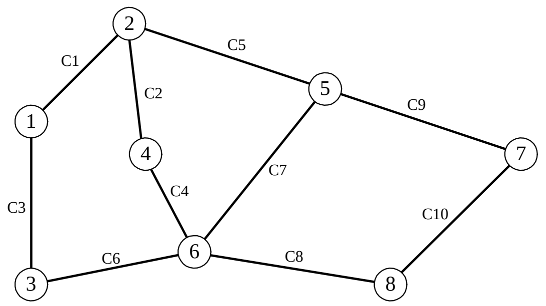

Figure 13: Test Network

blocking for all $N = 42$ definable classes is $\bar{P}_{B_i} = 1\%$, $i = 1, \ldots, 42$. The cost vector equals $\boldsymbol{\xi} = [1, 1, \ldots, 1]$.

Table 1: Routing and Offered Traffic in the Test Network

pair	route	traffic		pair	route	traffic
1,2	1,	40		3,5	6,7	30
1,3	3,	40		3,6	6,	40
1,4	1,2	30		3,7	6,7,9	20
1,5	1,5	30		3,8	6,8	30
1,6	3,6	30		4,5	2,5	30
1,7	1,5,9	20		4,6	4,	40
1,8	3,6,8	20		4,7	4,8,10	20
2,3	1,3	30		4,8	4,8	30
2,4	2,	40		5,6	7,	40
2,5	5,	40		5,7	9,	40
2,6	2,4	30		5,8	9,10	30
2,7	5,9	30		6,7	8,10	30
2,8	5,7,8	20		6,8	8,	40
3,4	6,4	30		7,8	10,	40

In cases when the knapsack approximation is used in its original version with the Kaufman-Roberts recursion formula, the following values of $\tilde{\mathbf{C}}$ and $\mathbf{C}^0$ are obtained:

$$\begin{aligned}\tilde{\mathbf{C}} &= \{2393, 2090, 1939, 2393, 2696,\\ &\quad 3148, 1786, 2997, 2242, 1939\},\\ \mathbf{C}^0 &= \{2495, 2158, 2026, 2495, 2806,\\ &\quad 3272, 1869, 3117, 2339, 2026\}.\end{aligned}$$

Note that the values are quite close. This is why the algorithm converges after one step and yields the following result:

$$\begin{aligned}\mathbf{C}^1 &= \{2491, 2154, 2022, 2491, 2802,\\ &\quad 3268, 1865, 3113, 2335, 2022\}.\end{aligned}$$

The costs at the points are: $\Theta(\tilde{\mathbf{C}}) = 23623$, $\Theta(\mathbf{C}^0) = 24603$ and $\Theta(\mathbf{C}^1) = 24563$. Deviation from optimum is within $\Delta_{dev} = 3.98\%$.

When the algorithm is used with the UAA approximation, the result is reached after two iterations

$$\begin{aligned}\mathbf{C}^2 = \{&2490.75,\ 2153.75,\ 2021.75,\ 2490.75,\ 2801.75,\\ &3267.75,\ 1864.75,\ 3112.75,\ 2334.75,\ 2021.75\}.\end{aligned}$$

with cost $\Theta(\mathbf{C}^2) = 24560.5$ and deviation bound $\Delta_{dev} = 3.97\%$. The result has real numbers because the UAA approximation can accept non-integer values of capacity.

The blocking probability for all pairs is less than 1%. Deviations from maximum allowed blocking probability vary. The smallest value of blocking probability is for pair $(1, 2)$ for service 1: $\bar{P}_{B_{(1,2)}} = 0.043\%$. The largest blocking probability is for pair $(4, 8)$ for service 2, $\bar{P}_{B_{(4,8)}} = 0.95\%$.

In order to find the real deviations from the optimum capacity, we conducted the following experiment. The capacity vector $\mathbf{C}^*$ obtained for maximum blocking probabilities of 1% is pronounced to be the optimum solution and obtained blocking probabilities are pronounced to be the upper limits on the blocking probability of the new optimization problem. The cost vector is the same. In this case, the optimum solution to the new problem should be $\mathbf{C}^*$ with cost $\Theta(\mathbf{C}^*)$. We tested how close our heuristic algorithm approached $\mathbf{C}^*$. The same experiment was conducted for several offered traffics. Offered traffic in table 1 was multiplied with coefficients $\alpha = 0.1$, 0.5, 1, 2, 5, 10. The results of experiments are summarized in table 2. The second column represents

Table 2: Routing and Offered Traffic in Test Network

α	Opt. Θ	Θ	dev.	Δ_{dev}	Δ^*_{dev}
0.1	3220	3241	0.652 %	1.06 %	6.25 %
0.5	12942	12964	0.170 %	0.48 %	4.83 %
1	24563	24594	0.126 %	0.43 %	3.98 %
2	47328	47374	0.097 %	0.43 %	3.19 %
5	114836	114950	0.099 %	0.41 %	2.61 %
10	226450	226646	0.087 %	0.40 %	2.28 %

cost Θ obtained by running algorithm with all maximum blocking probabilities equaling 1% (case 1). The third column represents achieved costs Θ after running the algorithm with blocking probabilities in case 1 taken as maximum blocking probabilities for case 2. The fourth column represents the real deviation and the fifth column represents bounds for the deviation according to (13). The sixth column represents deviation bounds for case 1. One can see that real deviations from optimum are much smaller than the bound, and the deviation decreases as the traffic increases. This effect can be explained due to the very sharp curvature of the edge of Ψ close to $\tilde{\mathbf{C}}$. Due to this effect it is expected that the results of the algorithm for different cost vectors $\boldsymbol{\xi}$ will be quite concentrated. Table 3 shows the resulting vectors $\mathbf{C}$ for different cost vectors. The concentration of results is evident.

Table 3: Algorithm Results for Different Cost Vectors $\boldsymbol{\xi}$

$\boldsymbol{\xi}$	$\mathbf{C}$		$\boldsymbol{\xi}$	$\mathbf{C}$		$\boldsymbol{\xi}$	$\mathbf{C}$
1	2491		1	2495		6	2494
1	2154		5	2155		23	2154
1	2022		9	2021		16	2024
1	2491		3	2494		9	2494
1	2802		2	2805		7	2805
1	3268		5	3269		40	3265
1	1865		8	1865		18	1866
1	3113		8	3113		38	3110
1	2335		7	2335		24	2335
1	2022		7	2022		11	2024

CONCLUSION

In this paper we presented a new heuristic algorithm for capacity planning in Erlang multiservice loss networks. The algorithm is based on the knapsack reduced load approximation, since this method proved to obtain the best results in numerical comparison to other reduced load approximations. Two versions of the algorithm were developed, one which uses Kaufman-Roberts recursion, and the other one which uses uniform asymptotic approximation (UAA). The values for capacity obtained by both versions of the algorithm do not differ significantly. Both versions of the algorithm converge quickly although the UAA version is faster. Numerical experiments show that the difference between the optimum capacities and capacities obtained by the algorithm is less than 3%. In order to verify that the error introduced by the algorithm is small in the general case, we plan to run the algorithm on a large number of randomly chosen network scenarios. The new results will be reported in the papers to follow.

REFERENCES

Chung, S.-P. and Ross, K. (1993). "Reduced Load Approximations for Multirate Loss Networks". *IEEE Transactions on Communications*, 41:1222–1231.

Conway, A. E., Pinski, E., and Tridandpani, S. (1994). "Efficient Decomposition Methods for the Analysis of Multi-Facility Blocking Models". *Journal of the ACM*, 41:648–675.

Kaufman, J. S. (1981). "Blocking in a Shared Resource Environment". *IEEE Transactions on Communications*, 29:1474–1481.

Kelly, F. P. (1991). "Loss networks". *The Annals of Applied Probability*, 1:319–378.

Labourdette, J.-F. and Hart, G. (1992). "Blocking Probabilities in Multitraffic Loss Systems: Insensitivity, Asymptotic Behavior, and Approximations". *IEEE Transactions on Communications*, 40:1355–1366.

Mitra, D. and Morrison, J. A. (1994). "Erlang Capacity and Uniform Approximations for Shared Unbuffered Resources". *IEEE Transactions on Networking*, 6:558–570.

Morrison, J. A. (1981). "A Service System with Heterogeneous User Requirements - Application to Multi-Services Telecommunications Systems". *Performance of Data Communication Systems and their Applications*, 6:423–431.

Whitt, W. (1985). "Blocking When Service is Required from Several Facilities Simultaneously". *AT&T Technical Journal*, 64:1807–1856.

AUTHOR BIOGRAPHIES

ZDENKO VRDOLJAK received his B.Sc. and M.Sc. in Electrical Engineering from the University of Zagreb in 2000 and 2003 respectively. He joined Department of Telecommunications, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, in 2000, where he works as research assistant. He is member of the "Laboratory for Advanced Telecommunication Studies (LATS)" and student member of IEEE.

MLADEN KOS received a B.Sc., M.Sc. and Ph.D. in Electrical Engineering from the University of Zagreb, Croatia in 1972, 1976 and 1983, respectively. He is with the Faculty of Electrical Engineering and Computing of the University of Zagreb where he works as a full professor at the Department of Telecommunications. His research interests include the network design and optimization, stochastic models, teletraffic theory and performance analysis, high-speed network dynamics. He is a founder of the "Laboratory for Advanced Telecommunication Studies (LATS)". He is a member of IEEE and INFORMS.