QoS-Based Routing Algorithms for ATM Networks

Mort Naraghi-Pour
Dept. of ECE
Louisiana State Univ.
Baton Rouge, LA 70810

Manju Hegde
Dept. of EE
Washington Univ.
St. Louis, MO 63130

Jesus Sanchez-Barrera
Telecommunications
Dept. of Information Technology
CFE-LAPEM
Irapuato, Gto. C.P. 36541, Mexico

Corresponding author: Mort Naraghi-Pour
E-mail: mort@ee.lsu.edu Ph: (504) 388-5551 FAX: (504) 388-5200

Abstract

Cell loss probability (*clp*) is one of the most important measures of quality of service in an ATM network. In this paper we present two routing algorithms for ATM networks that meet the *clp* requirements of bursty ON-OFF sources that request connection while ensuring that the *clp* requirements of the existing calls are not compromised. The first algorithm checks the entire set of feasible routes while the second algorithm remembers the last successful route for each OD pair and is consequently simpler to implement. Our numerical results show that the *clp* requirements are achieved while maintaining low call blocking probabilities. Further, the routing algorithms are sensitive to the *clp* requirements and respond to small changes in them. If the *clp* requirements are made stricter, then the routing algorithms continue to maintain these new requirements at the expense of small increases in the call blocking probabilities. Different cell loss probability criteria can be accommodated for different types of calls which may differ in parameters such as burstiness, average burst length and average or peak rates. A comparison of the two algorithms in terms of the trade-off between *clp* and call blocking probability is also provided which confirms the effectiveness of the simpler second algorithm. We also present a feedforward neural network which is easily trained and which can be used for the real time implementation of the algorithms.

Index Terms: ATM networks, routing and congestion control, call admission, quality of service.

This work was supported in part by a grant from NSF under contract number NCR-9706545.

1 Introduction

Asynchronous transfer mode (ATM) is a virtual-circuit (VC) oriented packet-switching technique that allows for integration of a wide variety of traffic sources and provides selectable quality of service (QoS). At the time of call setup, the quality of service is negotiated between the user and the network. For the call to be accepted, the network must be capable of providing a negotiated quality of service, in effect, guaranteeing that as new calls are admitted, the quality of service of the existing calls will not be compromised. If this can not be guaranteed, the new call is rejected (or blocked). This results in a trade-off between call blocking probability and quality of service requirements. In ATM networks, as in telephone networks, call blocking probability is an important measure of network performance. In addition to the loss of revenues for the service providers, blocked calls result in customer dissatisfaction and annoyance.

At the cell level, QoS is described by cell loss, cell delay and cell jitter. In this paper, however, we restrict attention to cell loss probability (*clp*) as the only measure of QoS. It is clear that stricter *clp* requirements result in higher call blocking rates. However, in an ATM network employing statistical multiplexing and serving heterogeneous traffic sources, the level of sensitivity of call blocking probabilities to *clp* requirements is not clear. We consider an ATM network that supports heterogeneous traffic sources having different traffic statistics and/or QoS requirements. Ensuring that the cell loss probability requirements of calls are satisfied, we then investigate the trade-offs between call blocking probabilities and cell loss probabilities.

Assuming that the *clp* of a new call can be guaranteed, a routing algorithm is needed that can be used to select a path in the network over which the new call can be connected. For the purpose of studying the trade-offs described above, we develop two routing algorithms, reminiscent of the adaptive routing algorithms suggested for the telephone network [17] [13], that guarantee the negotiated *clp* for each call currently connected in the network as well as for the new calls requesting connection. If the *clp* requirements of a new call cannot be met or if meeting it would compromise the *clp* of any of the existing calls in the network, the new call is rejected. We show that this is achieved while maintaining a reasonable call blocking probability. The computations necessary to make the routing decisions can be performed off-line through any fast computational paradigm and in Section 4 we show how a feed-forward neural network can be gainfully utilized for such purpose.

Quality of service based routing is considered in [11] where the authors propose several routing algorithms for ATM networks. These are based on the well known least loaded routing (LLR) algorithm [5] used in circuit switched networks. The authors classify the ATM networks into four classes according to the following: (1) homogeneous calls (i.e., all calls have the same traffic statistics and QoS requirements) without statistical multiplexing. (2) homogeneous calls with statistical multiplexing. (3) heterogeneous calls (i.e., the calls have different traffic statistics and/or QoS requirements) without statistical multiplexing. (4) heterogeneous calls with statistical multiplexing. Of these, case (1) corresponds to single rate circuit switching and has been investigated extensively in the literature (see for example [16] [5] [17] [13] [23]). Cases (2) and (3) are investigated in [11]. Case (4) is not investigated. However, ATM networks are currently envisaged to carry heterogeneous traffic. Moreover, statistical multiplexing cannot be avoided as, otherwise, in the presence of bursty sources the network will be extremely inefficient in its utilization of link capacities.

In this paper we consider case (4) above where the network performs statistical multiplexing and handles heterogeneous traffic. Since we would like to guarantee the *clp* of every call admitted by the network, an accurate method is needed for computing the *clp* of heterogeneous traffic sources from an ATM multiplexer. In [24] a computationally tractable model is presented for homogeneous traffic. This work is extended to heterogeneous traffic in [25]. In these two papers the arrival process into the multiplexer is modeled as a Markov modulated deterministic process ($MMDP$) and the multiplexer is then modeled as an $MMDP/D/1/K$

queueing system. The authors make two renewal assumptions on the arrival process and the server which make the problem mathematically tractable. The cell loss probability for the multiplexer is then evaluated. However, as noted in [24], the renewal assumptions alter the nature of the arrival and departure processes. As a result the $MMDP/D/1/K$ model can not account for short term or cell level congestion. Consequently, as the results in [24] [25] indicate, the $MMDP/D/1/K$ model is not very accurate in estimating the cell loss probability. We have also obtained several numerical results using the $MMDP/D/1/K$ model all of which confirm this. Since our goal is to provide guaranteed QoS, we have developed an accurate method for the computation of *clp* which is presented in Section 3.

A great deal of other research work has been devoted to analyzing the performance of an ATM multiplexer [1] [2] [6] [9] [10] [12] [18] [20]. In particular, in [6] [9] and [20] the authors have studied queueing systems with periodic arrival processes when all the sources have the same period. In [20] the authors have obtained upper and lower bounds to the queue length distribution for a $\sum D_i/D/1$ queueing system. Further results on these queueing systems can be found in [15] and [18].

We use bursty ON-OFF models for sources and present two routing algorithms that meet the *clp* requirements. Our numerical results show that the *clp* requirements are achievable while maintaining low call blocking probability. Further, the routing algorithms are sensitive to the *clp* requirements and respond to small changes in them. If the *clp* requirements are made stricter, then the routing algorithms continue to maintain these new requirements at the expense of small increases in the call blocking probabilities. We also present a feedforward neural network which is easy to train and which can be used for the real time implementation of the algorithms.

The rest of the paper is organized as follows. In Section 2 we present our network model and the routing algorithms. In Section 3 we develop accurate models for evaluating the *clp* for an ATM multiplexer serving heterogeneous traffic. These models are used in the routing algorithms of Section 2. In Section 4 we present the computational paradigm of a feedforward neural network, demonstrate how it can be trained off-line and present numerical results confirming its applicability for our model. Numerical results are presented in Section 5. Finally, conclusions are presented in Section 6.

2 The Network Model and the Routing Algorithms

We consider an ATM network consisting of ATM switches interconnected by high-speed links. We assume that each pair of switches is connected by a single virtual path (VP), i.e., the VP network is fully connected. Using a multiplexer followed by a buffer, each VP serves a number of virtual circuits (VC's). (Throughout this paper the terms VC and call are used interchangeably.) We assume that all the VC's on a VP share a buffer space that is dedicated to the VP. (Since the VP consists of a number of VC's, this assumption does not significantly compromise the effects of statistical multiplexing on a link.)

The VP network is modeled as a directed graph $(\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of switches of the network and $\mathcal{L}$ is the set of (directed) VP's. The bandwidth of VP l is C_l cells/sec. and its buffer size is K_l cells. Also, we denote by $\mathcal{P}$ the set of all origin/destination (OD) pairs.

In order to make the task of managing routing decisions easier, VC's are classified into types based on their traffic characteristics and their QoS requirements. Since we consider *clp* based QoS, the call routing decisions are based on the number of VC's of each type that the network carries. Since ATM utilizes virtual circuit switching, the routing policy essentially consists of decision rules which determine (i) whether an arriving VC should be accepted or rejected and (ii) if accepted, what particular path the VC should be assigned to. Therefore, with each OD pair p we associate a route set $\mathcal{R}_p$ consisting of all the paths that a VC corresponding to this OD pair can take. Each route set consists of the *direct route* (the route consisting of the single VP connecting the two nodes) and a number of *alternate* *routes*. An alternate route r_p consisting of VP's l_1 and l_2 is denoted by $r_p = \{l_1, l_2\}$. In this paper we only consider alternate routes that are comprised of two VP's. Extension to the general case is straightforward.

Let $M_{l,q}$ denote the number of VC's of type q being routed through VP l and let $\mathbf{M}_l = (M_{l,1}, M_{l,2}, \ldots, M_{l,Q})$ denote the state of VP l, where Q denotes the total number of VC types. For a fixed state and a fixed buffer size of VP l let $p_{l,q}$ denote the fraction of cells of type q that are lost from the buffer. Let $\mathbf{P}_l = (p_{l,1}, p_{l,2}, \ldots\ p_{l,Q})$. In order to describe our routing algorithms, we assume that, given $\mathbf{M}_l$ and K_l, $\mathbf{P}_l$ can be evaluated from some existing procedure. In Section 3 we present a method for calculating these cell loss probabilities. From now on we only assume that each switch has knowledge of the state of all the VP's in the network which share at least one link with any VP that passes through this switch and uses this information to compute the *clp*'s for each VC type.

If a route consists of a single VP l, say, then the cell loss probabilities are given by $\mathbf{P}_l$. For an alternate route $r_p = \{l_1, l_2\}$, we assume that the cell loss in the buffers of the two VP's are independent [3]. Then $clp_{r_p,q}$, the fraction of cells of type q that are lost on route r_p, is given by

$$clp_{r_p,q} = 1 - (1 - p_{l_1,q})(1 - p_{l_2,q}), \tag{1}$$

where $p_{l_1,q}$ and $p_{l_2,q}$ are the qth components of $\mathbf{P}_{l_1}$ and $\mathbf{P}_{l_2}$, respectively.

Assume that a VC of type q arrives requesting connection for OD pair p. For $q = 1, 2, \ldots\ Q$, let g_q denote the maximum *clp* acceptable for VC of type q. We now present the two routing algorithms.

Algorithm 1

1. First try the direct route. Assume the VC is accepted and is routed through the direct route using VP l. Compute the new *clp* for all VC's that utilize VP l. If the *clp* of all these VC's is less than the maximum acceptable *clp*, then accept the new VC; else

2. Initialize the set of feasible routes, $\mathcal{FR}$, to an empty set, $\mathcal{FR} \longleftarrow \emptyset$.

3. For each alternate route $r_p \in \mathcal{R}_p$, obtain the new *clp* for all VC's that utilize route r_p or either of the two VP's of r_p if the VC is connected through r_p. If all these *clp*'s are less than the maximum acceptable *clp*'s, let $\mathcal{FR} = \mathcal{FR} \bigcup \{r_p\}$.

4. If $\mathcal{FR} = \emptyset$, reject the VC; else

5. Let $\mathbf{CLP}_{r_p} = (clp_{r_p,1}, clp_{r_p,2}, \ldots\ clp_{r_p,Q})$. Define the cost function $f_{cost}(\cdot)$ as

$$f_{cost}(\mathbf{CLP}_{r_p}) = \sum_{q=1}^{Q} (\frac{clp_{r_p,q}}{g_q}) \,. \tag{2}$$

   Accept the VC for any route r_p^*, where

$$r_p^* = \arg \min_{r_p \in \mathcal{FR}} \{f_{cost}(\mathbf{CLP}_{r_p})\}. \tag{3}$$

Essentially, in the above algorithm if the cell loss probability requirements of all the existing VC's as well as the new VC can be met on some route, the VC is accepted. Preference is given to the direct route. If the VC cannot be accommodated on the direct route, then it is routed on the alternate route that results in the smallest network cost as measured in (2).

The Cost Function $f_{cost}(\cdot)$

In the previous algorithm if the direct route is not available and we have more than one alternate route that is feasible, i.e., the set $\mathcal{FR}$ has more than one candidate route, then we need to make a choice between these candidate routes. In general, it is not easy to determine which is the best candidate. One option is to choose the alternate route with the maximum available bandwidth. However, this may cause higher VC blocking probability for VC's with higher bandwidth requirements that arrive later. Another option is to consider a cost function that takes into account all the VC's that are affected if the arriving VC is accepted on a given route. Such cost functions, however, present the following problem. Consider a candidate route $r_p \in \mathcal{FR}$ and an alternate route that has a VP in common with r_p. Further, assume that the other VP of this route (not in common with r_p) is heavily congested. Then r_p may not be chosen to accept the incoming VC even if r_p is lightly loaded. This is due to the fact that we are considering all the VC's affected by route r_p, and the heavily congested routes have a greater effect on such cost functions. In fact, accepting the incoming VC on r_p would not have important repercussions on the heavily congested route. In comparison, the cost function given by (2) presents a good compromise. It only considers the effects of the chosen route on the two VP's of the alternate routes. However, the choice of a route in Algorithm 1 nevertheless requires the identification of a minimum cost path, which can be time-consuming. We therefore suggest Algorithm 2 which remembers the previous good path.

Algorithm 2

Initialize $r_p^{old} = 0$ for all p.

1. First try the direct route. Assume the VC is accepted and is routed through the direct route. Compute the new *clp* for all the VC's that utilize this VP. If the *clp* of all these VC's is less than the maximum acceptable *clp*, then accept the new VC; else

2. If $r_p^{old} = 0$, then choose one of the alternate routes at random, say $r_p \in \mathcal{R}_p$, set $r_p^{old} = r_p$ and go to step 3.
   If $r_p^{old} \neq 0$, set $r_p = r_p^{old}$.

3. Obtain the *clp* for all VC's that utilize route r_p or either of the two VP's of r_p. If all these *clp*'s are less than the maximum acceptable *clp*'s accept the new VC for route r_p; otherwise reject the VC and set $r_p^{old} = 0$.

In Algorithm 2, we do not check the entire set $\mathcal{R}_p$ for feasible routes; rather, we are content to remember for each OD pair the last route that successfully served as an alternate route and continue using it until it fails (i.e., the VC is rejected) at which point a new route is chosen at random and the entire process started anew. While Algorithm 2 may result in a larger VC blocking probability, it is considerably less complex to implement than Algorithm 1. Algorithm 2 is inspired by the Dynamic Alternate Routing scheme for circuit-switched networks implemented in the British Telecom network [8].

3 Evaluation of the Cell Loss Probabilities

To implement the routing algorithms we need a procedure for evaluating the *clp* of each type of VC given the number of various types of VC's connected over a VP of bandwidth C and buffer size K. A VC is modeled as an ON-OFF source. The durations of the ON and OFF periods for a VC of type q are independent exponentially distributed random variables with means $\frac{1}{\alpha_q}$ and $\frac{1}{\beta_q}$, respectively. When a VC of type q is in the OFF state, it is idle and when it is in the ON state it transmits at the fixed rate of BP_q bits/sec.. Thus a VC of type q is defined by three traffic parameters α_q, β_q and BP_q, as well as g_q, the required *clp*. Equivalently, the traffic parameters can be determined by BM_q, L_q and b_q, where BM_q denotes the VC's average bit rate, L_q denotes the average burst length (in cells), and b_q denotes the burstiness. These parameters can be calculated from the previous three as follows. $BM_q = BP_q \cdot \beta_q/(\alpha_q + \beta_q)$, $L_q = BP_q/(\alpha_q \cdot 424)$ (recall that an ATM cell consists of 53 bytes or 424 bits) and $b_q = BP_q/BM_q$.

Let $\mathbf{X}(t) = (X_1(t), X_2(t), ..., X_Q(t))$ where, for $q = 1, 2, ..., Q$, $X_q(t)$ indicates the number of VC's of type q that are in the ON state at time t. Clearly $\{\mathbf{X}(t), t \geq 0\}$ is an irreducible finite-state Markov Process. For $n = 0, 1, 2, ...$ define ξ_n as the nth transition epoch of $\mathbf{X}(t)$. Let $\xi_0 = 0$ and let $X(\xi_0) = (0, 0, ..., 0)$. Let $\mathbf{X}_n = \mathbf{X}(\xi_n^+)$. Then $\{\mathbf{X}_n\}$ is an irreducible Markov chain. Let $\mathbf{P}$ denote the transition probability matrix of $\{\mathbf{X}(t), t \geq 0\}$, and let $P_{\mathbf{ij}}$

denote the $(\mathbf{i}, \mathbf{j})$ element of $\mathbf{P}$, where $\mathbf{i} = (i_1, i_2, ..., i_Q)$ and $\mathbf{j} = (j_1, j_2, ..., j_Q)$. Then

$$P_{\mathbf{i}\mathbf{j}} = \begin{cases} \frac{\beta_{i_q}(M_q - i_q)}{\gamma_{\mathbf{i}}} & \text{if } \; j_q = i_q + 1; \; j_k = i_k \; \text{ for } \quad \forall k \neq q; \\ \frac{i_q \alpha_q}{\gamma_{\mathbf{i}}} & \text{if } \; j_q = i_q - 1; \; j_k = i_k \; \text{ for } \quad \forall k \neq q; \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

where $\gamma_{\mathbf{i}} = \sum_{q=1}^{Q} [(M_q - i_q)\beta_q + i_q \alpha_q]$.

Let $T_n = \xi_{n+1} - \xi_n$. Given that $\mathbf{X}(\xi_n^+) = \mathbf{i} = (i_1, i_2, ..., i_Q)$, then T_n is an exponential random variable with parameter $\gamma_{\mathbf{i}}$. Furthermore, if $\mathbf{X}(\xi_n^+) = \mathbf{i} = (i_1, i_2, ..., i_Q)$, then during the interval $[\xi_n, \xi_{n+1})$, cells of type q arrive according to a periodic or deterministic process at a fixed rate $B_{i_q} = i_q B P_q$. However, since the sequence $\{\xi_n\}$ is random, the first cell of type q arrives at the multiplexer with some random delay of $\mathcal{T}_{n,q}$ after ξ_n. It is clear that $\mathcal{T}_{n,q} \in [0, \frac{1}{B_{i_q}}]$. We assume that the sequence $\{\mathcal{T}_{n,q}\}$ is an independent sequence of uniformly distributed random variables. Consider the nth transition epoch ξ_n of $\mathbf{X}(t)$ and let $\mathcal{T}_{n,0}$ denote the elapsed service time of the cell in transmission at the moment ξ_n. We also assume that $\{\mathcal{T}_{n,0}\}$ is a sequence of independent random variables each uniformly distributed on the interval $[0, \frac{1}{C}]$. Further, it is assumed that the sequences $\{\mathcal{T}_{n,i}\}$, $i = 0, 1, ..., Q$, are independent of each other and of the sequence $\{X_n\}$.

Let $Y(t)$ denote the number of cells in the buffer (including the one being served) at time t. Given that $\mathbf{X}(\xi_n^+) = \mathbf{i} = (i_1, i_2, ..., i_Q)$ and given the values of the phases $\mathcal{T}_{n,i}$ for $i = 0, 1, ..., Q$, during the interval $[\xi_n, \xi_{n+1})$, the process $Y(t)$ evolves deterministically. Let $Y_n = Y(\xi_n^+)$. Then our assumptions on the sequences $\{\mathcal{T}_{n,i}\}$, $i = 0, 1, ..., Q$, imply that the sequence $\{(\mathbf{X}_n, Y_n)\}$ is a Markov chain. We evaluate $\mathbf{Q}$, the transition probability matrix of this chain. Let $\underline{\mathcal{T}}_n = (\mathcal{T}_{n,0}, \mathcal{T}_{n,1}, ..., \mathcal{T}_{n,Q})$ and let $\underline{\tau} = (\tau_0, \tau_1, ..., \tau_Q)$. Then

$$Pr\left((\mathbf{X}_{n+1}, Y_{n+1}) = (\mathbf{j}, d) \,|\, (\mathbf{X}_n, Y_n) = (\mathbf{i}, k), \underline{\mathcal{T}}_n = \underline{\tau}\right) = P_{\mathbf{i}\mathbf{j}}\, a_{k,d}^{\mathbf{i}}\,(\underline{\tau}), \tag{5}$$

where $a^{\mathbf{i}}_{k,d}(\underline{\tau}) = P\left(Y_{n+1} = d \,|(\mathbf{X}_n, Y_n) = (\mathbf{i}, k), \underline{\mathcal{T}}_n = \underline{\tau}\right)$. Let $a^{\mathbf{i}}_{k,d} = E_{\underline{\mathcal{T}}_n}\left[a^{\mathbf{i}}_{k,d}(\underline{\mathcal{T}}_n)\right]$ and define the $(K+1) \times (K+1)$ matrix $\mathbf{A}_{\mathbf{i}} = [a^{\mathbf{i}}_{k,d}]$. The matrix $\mathbf{Q}$ can then be expressed as

$$\mathbf{Q} = \begin{bmatrix} P_{\mathbf{i}_a,\mathbf{i}_a}\mathbf{A}_{\mathbf{i}_a} & P_{\mathbf{i}_a,\mathbf{i}_b}\mathbf{A}_{\mathbf{i}_a} & \ldots & P_{\mathbf{i}_a,\mathbf{i}_z}\mathbf{A}_{\mathbf{i}_a} \\ P_{\mathbf{i}_b,\mathbf{i}_a}\mathbf{A}_{\mathbf{i}_b} & P_{\mathbf{i}_b,\mathbf{i}_b}\mathbf{A}_{\mathbf{i}_b} & \ldots & P_{\mathbf{i}_b,\mathbf{i}_z}\mathbf{A}_{\mathbf{i}_b} \\ \vdots & \vdots & \ldots & \vdots \\ P_{\mathbf{i}_z,\mathbf{i}_a}\mathbf{A}_{\mathbf{i}_z} & P_{\mathbf{i}_z,\mathbf{i}_b}\mathbf{A}_{\mathbf{i}_z} & \ldots & P_{\mathbf{i}_z,\mathbf{i}_z}\mathbf{A}_{\mathbf{i}_z} \end{bmatrix}, \tag{6}$$

where $\mathbf{i}_a, \mathbf{i}_b, ..., \mathbf{i}_z$, represents the lexicographical ordering of the state space of $\mathbf{X}(t)$.

Having obtained the matrix $\mathbf{Q}$, we can evaluate π, the stationary distribution of the chain $\{(\mathbf{X}_n, Y_n)\}$. We denote by $\pi_{\mathbf{i}k}$ the steady state probability that the chain is in state $(\mathbf{i}, k)$. Given that $(\mathbf{X}_n, Y_n) = (\mathbf{i}, k)$, denote by $N^{(q)}_{\mathbf{i},k}$ and $R^{(q)}_{\mathbf{i}\,k}$ the number of cells of type q that arrived and the number of cells of type q that were lost during the interval $[\xi_n, \xi_{n+1})$, respectively (the computation of these is described in Section 3.1). Then p_q, the cell loss probability for VC's of type q, is given by

$$p_q = \frac{\sum_{\mathbf{i}\,k} E\left[R^{(q)}_{\mathbf{i}\,k}\right]\pi_{\mathbf{i}\,k}}{\sum_{\mathbf{i}\,k} E\left[N^{(q)}_{\mathbf{i}\,k}\right]\pi_{\mathbf{i}\,k}}. \tag{7}$$

3.1 Computational procedures

To evaluate the matrix $\mathbf{Q}$, we need to compute the set of values $\{a^{\mathbf{i}}_{k,d}\}$. For a fixed value of $\underline{\tau}$, $a^{\mathbf{i}}_{k,d}(\underline{\tau})$ can be evaluated from the evolution of the process $Y(t)$. Give that $\mathbf{X}_n = \mathbf{i}$, $\underline{\mathcal{T}}_n = \underline{\tau}$, and $Y_n = k$, during the interval $[\xi_n, \xi_{n+1})$ the multiplexer can be modeled by the $\sum D_i/D/1/K$ queuing system [20] running for a period of T_n secs. The arrival process is the superposition of Q independent periodic sources (with different periods) with arrival rates B_{i_q} and initial phases τ_q for $q = 1, 2, ..., Q$; the service rate is equal to the link capacity C

cells/sec. with an initial phase τ_0, and the buffer size is K.

Fix the values of $\mathbf{i}$, k and $\underline{\tau}$. Let $t_0 = 0$ and let $t_1, t_2, t_3, ...$ be the buffer state transition times in the $\sum D_i/D/1/K$ model. Let $I_r = [t_{r-1}, t_r)$. Note that the value of $Y(t)$ is fixed for all $t \in I_r$. For ease of notation, we use "$Y(I_r) = l$" instead of "$Y(t) = l$, for all $t \in I_r$". Define $\mathcal{I}^{(d)}$ as the set of time intervals in the $\sum D_i/D/1/K$ model for which the buffer contains d cells, i.e., $Y(t) = d$ for $t \in \mathcal{I}^{(d)}$. Therefore $\mathcal{I}^{(d)} = \bigcup\limits_{r:Y(I_r)=d} I_r$. Then

$$a^{\mathbf{i}}_{k,d}(\underline{\tau}) == P\left(T_n \in \mathcal{I}^{(d)}\right) = \sum_{r:Y(I_r)=d} \left[\exp\left(-\gamma_{\mathbf{i}} t_{r-1}\right) - \exp\left(-\gamma_{\mathbf{i}} t_r\right)\right]. \tag{8}$$

Since the model is deterministic, given the values of $\mathbf{i}$, k and $\underline{\tau}$, we can straightforwardly calculate the values of $t_1, t_2, t_3, \ldots$. In fact, a simulation procedure for $\sum D_i/D/1/K$ can be used to compute $a^{\mathbf{i}}_{k,d}(\underline{\tau})$ very efficiently. It should be pointed out that when $\sum_{q=1}^{Q} B_{i_q} < C$, then for $d = 0$ and $d = 1$ equation (8) involves an infinite number of terms on the right hand side. (The number of cells in the buffer decreases to zero and thereafter it alternates between zero and one.) This is also true in the case when $\sum_{q=1}^{Q} B_{i_q} > C$, and for $d = K - 1$ and $d = K$. However, in both cases the series converges rapidly and, thus, can be accurately approximated by a finite sum. We evaluate $a^{\mathbf{i}}_{k,d}$ from $a^{\mathbf{i}}_{k,d}(\underline{\tau})$ by Monte Carlo simulation. The transition matrix $\mathbf{Q}$ can then be evaluated from (6). Having obtained $\mathbf{Q}$, we can use the balance equations to obtain the stationary distribution π (see [24] and [25] for more details).

Finally, to compute p_q we need to evaluate $E[R^{(q)}_{\mathbf{i},k}]$ and $E[N^{(q)}_{\mathbf{i},k}]$. For fixed values of $\mathbf{i}$, k and $\underline{\tau}$ let $s^q_0 = 0$, and let s^q_j denote the time when the jth cell of type q is lost from the buffer during the period $[\xi_n, \xi_{n+1})$. Then

$$\begin{aligned} E[R^{(q)}_{\mathbf{i},k} | \underline{T}_n = \underline{\tau}] &= \sum_{j=1}^{\infty} jP(R^{(q)}_{\mathbf{i},k} = j | \underline{T}_n = \underline{\tau}) = \sum_{j=1}^{\infty} jP(T_n \in [s^q_j, s^q_{j+1})) \\ &= \sum_{j=1}^{\infty} j[\exp(-\gamma_{\mathbf{i}} s^q_j) - \exp(-\gamma_{\mathbf{i}} s^q_{j+1})]. \end{aligned} \tag{9}$$

As in the case of the computation of $a^{\mathbf{i}}_{k,d}(\underline{\tau})$, we choose a number of samples for $\underline{\tau}$ and for each sample evaluate $E[R^{(q)}_{\mathbf{i},k}|\underline{\mathcal{T}}_n = \underline{\tau}]$ from the simulation of the $\sum D_i/D/1/K$ model. $E[R^{(q)}_{\mathbf{i},k}]$ is then calculated by averaging the results. Finally for $E[N^{(q)}_{\mathbf{i},k}]$ we have

$$E[N^{(q)}_{\mathbf{i},k}] = B_{i_q}E[T_n - \mathcal{T}_{n,q}] = B_{i_q}(\frac{1}{\gamma_{\mathbf{i}}} - \frac{1}{2B_{i_q}}).$$

We would like to point out that these computations *can be performed off-line* and are not part of the real-time routing decisions that the network management needs to make at the time of VC setup. Subsequently, the results can be used to train a feedforward neural network as described in Section 4 which allows the *clp* to be computed in real time.

Figure 1 shows a comparison of the cell loss probability obtained from the model described in this section vs. simulation. There are two types of VC's and the cell loss probability is plotted as a function of the number of VC's of type 2. There are $M_1 = 40$ VC's of type one. The VP bandwidth is $C = 50$ Mbps and the buffer capacity is $K = 17$ cells. The parameters of the VC's are $BM_1 = BM_2 = .5$, $L_1 = L_2 = 100$, $b_1 = 2.5$ and $b_2 = 25$. It can be seen that the two results are very close. Several other examples were considered all of which confirmed the close match. For comparison we also show the results from the $MMDP/D/1/K$ model, [25], vs. simulation in Figure 2. We would like to point out that when this model was used in the routing algorithms, the *clp* requirements of the VC's could not be guaranteed.

4 A Neural Network for *clp* Computations

In cases where the number of VC types is high and the number of VC's in each type is also high, an attractive approach to compute the *clp*'s in real time is to use an artificial neural network (ANN). Such networks have been successfully utilized in other applications in ATM networks (see for example [14] [4] and [19]). Once the analytical computations are performed the results can be used to train a neural network which can be used to provide the *clp*'s in real time. The input to such a network consists of the number of VC's of each type and the output is the *clp* of each VC type. If it is desired to design an ANN for all the VP's in the network, then the parameters of the VP's such as bandwidth, buffer size, etc., can also be provided as inputs to the ANN. As an example we trained an ANN for the case presented in Figure 1. The training set contained 255 training vectors (M_1, M_2, p_1, p_2), where M_1 and M_2 are the number of VC's of type 1 and 2, respectively, and p_1 and p_2 are the *clp*'s corresponding to VC's of type 1 and 2, respectively. The network is a three layer feedforward network with four input units, 40 hidden units and two output units. The activation function in the input units is given by the linear function $g(h) = h$ whereas the activation function for all the other units (neurons in either the hidden or output layers) is given by the sigmoidal function

$$g(h) = \frac{1 - e^{(-h)}}{1 + e^{(-h)}} \,. \tag{10}$$

Since the values of *clp* in the region of interest are very small (in general no greater than 10^{-3}), a transformation was applied to the value p_q of the training pair, $q = 1, 2$, in order to improve the learning in such region. The actual output value to be learned, denoted by p'_q, is defined as

$$p'_q = \begin{cases} \frac{\log(p_q)}{12} + 1 & \text{if } p_q > 10^{-24}\,, \\ -1 & \text{if } p_q \leq 10^{-24}\,, \end{cases} \tag{11}$$

for $q = 1, 2$. Observe that the output of the activation function for the output unit defined by Equation (10) is in the range $[-1, 1]$ which is also the range of values for p'_q.

The backpropagation algorithm, [21], is used in the training process. Figure 3 shows a comparison of the *clp*'s from analytical results with those from the ANN. It can be seen that the ANN is able to learn the mapping very closely (the maximum observed error is less than 5%). Once the network is trained, given the number of VC's of each type, it can produce the *clp*'s in a single forward pass. This computation is very fast and can be performed in real-time.

In this application the universal approximation property of the feedforward neural networks is very useful [7]. Since the function representing the *clp* in terms of the number of VC's of each type is a smooth function, a small number of training vectors is sufficient to achieve successful training. In general, the properties inherent to neural networks such as learning capability, adaptability and generalization capability are elements that make them attractive candidates for the implementation of such a mapping.

5 Simulation Results

We have conducted a number of simulations to examine the performance of the routing Algorithms 1 and 2 in terms of the trade-offs between cell loss probability and call blocking probability. The VP network used in the simulations is shown in Figure 4 where the VP's are numbered as shown. (We have conducted more simulations on a six node and an eight node network [22]. The results and conclusions, which are not presented here, are similar to those in the following.) Each VP has a bandwidth of 50 Mbps. It is assumed that the arrival process of VC's of type q for OD pair p is Poisson with rate λ_p^q and that these arrival processes are independent for different VC types and different OD pairs. The VC durations are assumed to be exponential random variables with a mean of 10 secs and are independent of earlier arrival times and durations. For brevity of presentation, in all the examples we considered bidirectional VC's and only two VC types. We obtain the *clp* and the VC blocking probability (*vcbp*) for the two types of VC's for all the OD pairs.

The *clp*'s of a given VC type on the different routes in the route set of an OD pair can be different due to the interactions between different VC's going over the same VP's. We present the *clp*'s for each VC type on each route in order to exhibit this difference. On the other hand the VC blocking probability is simply a function of an OD pair and the VC type and so it is presented for each OD pair/VC type. To facilitate the graphical presentation, we have used two numbering systems presented in Table 1. In order to present the *clp* for each VC type on each route in the route set of an OD pair, we assign a number to each route/type pair (denoted r-type in the table and figures). For example for OD pair $(0, 2)$ in Table 1, there are three routes, namely routes $\{2\}$, $\{1, 4\}$ and $\{3, 11\}$, and thus there are six r-type numbers, namely 1,2,3,16,17, and 18. To present the VC blocking probability for each VC type for every OD pair we have assigned a number (denoted OD-type in the table and figures) to each OD pair/VC type. For example in Table 1, OD-type=4 corresponds to VC's of type 1 on OD pair $(0, 3)$ and OD-type=9 corresponds to VC's of type 2 on OD pair $(0, 3)$. In the figures we have plotted the *clp* vs. the r-type and the *vcbp* vs. the OD-type.

Figures 5 and 6 present the *clp* and the *vcbp*, respectively, obtained from simulations using Algorithm 1. The parameters of the VC types are given in Table 2. The arrival rates for the OD pairs are given by $\lambda_1^1 = 2.1$, $\lambda_p^1 = 2.7$, for $p = 2, ..., 5$ and $\lambda_p^2 = .3$, for $p = 1, 2, ..., 5$. The two VC types have equal mean rates BM and average burst length L. Type 2 VC's, however, are 10 times more bursty than type 1 VC's (the type 2 peak rate BP_2 is 10 times bigger than the type 1 peak rate BP_1). Notice that $BP_2 = \frac{1}{4}C = 12.5$ Mbps. Each VP has a buffer space of 17 cells. The *clp* requirements for VC's of type 1 and 2 are $g_1 = 10^{-5}$ and $g_2 = 10^{-4}$, respectively. As Figure 5 shows the algorithm meets the QoS requirements of both VC types. Clearly in such a setting the type 2 VC's experience a larger *clp* and *vcbp* and this is demonstrated in the figures. The dotted line in these figures show the average *clp* and the average *vcbp* for each VC type. (Note that changes to the arrival rates will result in changes in the cell loss probabilities and for certain combination of arrival rates they can be closer to the required QoS.) The same simulation was performed using Algorithm 2 (Algorithm 2) and the results are shown in Figures 7 and 8. It can be seen that except for one r-type, the QoS requirement is satisfied. As pointed out earlier, Algorithm 2 provides a less strict control over the resulting clp's.

A comparison of Figures 5 and 7 shows that the *clp*'s for different route/types resulting from the two algorithms are close and the average *clp*'s for the two VC types are very close. Comparing Figures 6 and 8 shows that the *vcbp* resulting from Algorithm 2 is generally higher than that from Algorithm 1. This is of course due to the fact that for each OD pair Algorithm 1 searches the entire set of alternate routes while Algorithm 2 only recalls the last route that successfully served as an alternate route. Consequently, Algorithm 1 may admit calls that are rejected by Algorithm 2. However, as these figures show the increase in *vcbp* is not large and, in fact, the average *vcbp* increases only slightly. We should point out that this is a small cost to pay as Algorithm 2 is much simpler to implement

In order to verify the control of the algorithms on QoS, we increased g_1 slightly from $g_1 = 10^{-5}$ to $g_1 = 3 \times 10^{-5}$ and left $g_2 = 10^{-4}$ unchanged. The results for Algorithm 1 are presented in Figures 9 and 10. A comparison of Figures 5 and 9 shows that the average *clp* has increased from 3.1×10^{-6} to 4.5×10^{-6} for type 1 VC's and from 2.7×10^{-5} to 3.4×10^{-5} for type 2 VC's. This shows the responsiveness of the algorithm to the QoS requirements. Even slight adjustment of the QoS requirement results in an adjustment of the resulting *clp*'s. A comparison of Figures 6 and 10 shows that the average *vcbp* remains unchanged for type 1 VC's and decreases slightly for type 2 VC's in Figure 10 indicating that even a slightly less restrictive QoS requirement results in a lower *vcbp*. Similar results are obtained using Algorithm 2 and are shown in Figures 11 and 12. Comparing the performance of the two algorithms in terms of *clp*'s and *vcbp*'s we can draw similar conclusions as in the previous case.

In the next example we change the VC type parameters to those in Table 3. The only change made to the type parameters is that BM_1 is increased to 5 Mbps. This is equivalent to making the two peak rates equal, i.e., $BP_1 = BP_2 = \frac{1}{4}C$. The arrival rates are given by $\lambda_p^1 = .1$ and $\lambda_p^2 = .25$ for $p = 1, 2, ..., 5$. The results for Algorithm 1 are shown in Figures 13 and 14 and those for Algorithm 2 are shown in Figures 15 and 16. It can be seen that the QoS requirements are again satisfied. However in this case VC's of type 1 suffer a higher blocking probability than before as they have a larger average rate. With the two VC types having the same peak rate, the type with higher burstiness (type 2) can take greater advantage of the statistical multiplexing nature of the network. We observe that in this case the control for the *clp* is stricter than before for both VC types, with most of the *clp* values for each r-type being significantly below their maximum values g_1 and g_2. The reason for this is that due to the high peak rates of all the VC's, each accepted VC greatly affects the *clp* in the route assigned to it. In order to guarantee the required *clp*'s, the algorithms are conservative in call admission resulting in lower *clp*'s than the maximum allowed values g_1 and g_2. Note also that, as before, the two algorithms are comparable in their performance.

To observe the effect of buffer size and to further verify the control of the algorithms we perform a set of simulations where the buffer size for every VP is only 6 cells (instead of 17 cells in the previous examples). The parameters for the VC types are given in Table 2 and the arrival rates are given by $\lambda_1^1 = \lambda_1^2 = .2$ and $\lambda_p^i = .6$ for $p = 2, ..., 5$ and $i = 1, 2$. The results for Algorithm 1 are shown in Figures 17 and 18 and for Algorithm 2 in Figures 19 and 20. Due to the small size of the buffer, admission of new VC on a particular route can significantly affect the *clp* of all the VC's that are currently utilizing this route. This effect is more pronounced for VC's of type 2 which have a higher bandwidth demand. Consequently, the algorithms exercise a stricter control on call admission. This results in the resulting *clp*'s being further from the QoS requirements. Note that the QoS requirement was met for every r-type in the network using either of the two algorithms. However, in order to guarantee the *clp* requirements of the two VC types, the algorithms are forced to satisfy a more stringent *clp* requirement for VC's of type 2. This is a consequence of the small buffer size which does not allow the network to perform statistical multiplexing effectively. As Figures 5 and 7 show when the buffer sizes are larger, the *clp*'s for the two VC types are closer to their respective maximum values and thus are substantially different.

6 Conclusions

We have presented two routing algorithms that meet the *clp* requirements of bursty ON-OFF sources that request connection while ensuring that the *clp* requirements of the existing calls are not compromised. The second routing algorithm is particularly easy to implement and in terms of the trade-off between *clp* and *vcbp* is comparable to the first algorithm. Our numerical results show that the *clp* requirements are achievable while maintaining low call blocking probability. Further, the routing algorithms are sensitive to the *clp* requirements and respond to small changes in them. If the *clp* requirements are made stricter, then the routing algorithms continue to maintain these new requirements at the expense of small increases in the call blocking probabilities. We have also presented a feedforward neural network which is easy to train and which can be used for the real time implementation of the algorithms. Different cell loss probability criteria can be accommodated for different types of calls which may differ in parameters such as burstiness, average burst length and average or peak rates. Numerical results have been provided which demonstrate the effectiveness of the routing algorithms.

References

[1] Anick, D., Mitra, D. and Sondhi, M.M., "Stochastic theory of a data-handling system with multiple sources," *BSTJ*, vol. 61, pp. 1871-1894, 1982.

[2] Baiocchi, A., Melazzi N.B., Listanti M., Roveri A. and Winkler R., "Loss Performance Analysis of an ATM Multiplexer Loaded with High-speed On-off Sources," *IEEE J. Select. Areas Commun.*, vol. 9, no. 3, pp. 388-393, Sept. 1992.

[3] Bertsekas, D. and Gallager, R., *Data Networks,* Second Edition, Prentice-Hall Inc., 1992.

[4] Chong, S., Li, S.Q. and Ghosh, J., "Predictive Dynamic Bandwidth Allocation for Efficient Transport of Real-Time VBR Video over ATM," *IEEE J. Select. Areas Commun.*, vol. 13, no. 1, pp. 12-23, Jan. 1995.

[5] Chung, S.P., Kaspher, A. and Ross, K.W., "Computing Approximate Blocking Probabilities for Large Loss Networks with State-Dependent Routing," *IEEE/ACM Transactions on Networking*, vol. 1, no. 1, pp. 105-115, February 1993.

[6] Eckberg, A.E., "The Single Server Queue with Periodic Arrival Process and Deterministic Service Time," *IEEE Trans. Commun.*, vol. 27, pp. 556-562, March 1979.

[7] Hornik, K., Stinchcombe, M. and White H., "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, 359-366, 1989.

[8] Gibbens, R.J., and Kelly, F.P., "Dynamic routing in fully connected networks," *IMA Journal of Mathematical Control and Information,* vol. 7, pp. 77-111, 1990.

[9] Gravey, A., "Temps d'attente et nombre de clients dans une file nD/D/1," *Annales de l'Institue H. Poincare-Probabilities et Statistiques,* vol. 20, no. 1, pp. 53-73, 1984.

[10] Guerin, R., Ahmadi, H. and Naghshineh, M., "Equivalent Capacity and its Application to Bandwidth Allocation in High-Speed Networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 968-981, Sept. 1991.

[11] Gupta, S., Ross, K.W. and El Zarki, M., "On Routing in ATM Networks," *Routing in Communication Networks,* Martha Steenstrup, ed., Prentice Hall, 1995, pp. 49-74.

[12] Heffes H. and Lucantoni D.M., "A Markov Modulated Characterization of Packetized Voice and Data Traffic and Related Statistical Multiplexer Performance," *IEEE J. Select. Areas Commun.*, vol. 4, no. 6, pp. 856-868, Sept. 1986.

[13] Hegde, M.V., Min, P.S. and Rayes, A., "State Dependent Routing: Traffic Dynamics and Performance Benefits" *Journal of Network and Systems Management* vol. 2., no. 2, pp. 125-149, 1994.

[14] A. Hiramatsu, "ATM Communications Network Control by Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 122-130, March 1990.

[15] Humblet, P.A., Bhargava, A. and Hluchyj, M.G., "Ballot Theorems Applied to the Transient Analysis of nD/D/1 Queues," *IEEE/ACM Trans. on Networking*, vol. 1, no. 1, pp. 81-95, Feb 1993.

[16] Mitra, D., Gibbens, R.J. and Haung, B.D., "State-Dependent Routing on Symmetric Loss Networks with Trunk Reservation-I," *IEEE Trans. Commun.*, vol. 41, no. 2, pp. 400-411, Feb 1993.

[17] Mitra, D., Gibbens, R.J. and Huang, B.D., "Analysis and Optimal Design of Aggregated Least-Busy-Alternative Routing on Symmetric Loss Networks with Trunk Reservations," *International Teletraffic Congress, 13*, pp. 477-481, Copenhagen 1991.

[18] Norros, I., Roberts, J.W., Simonian, A. and Virtamo, J.T., "The Superposition of Variable Bit Rate Sources in an ATM Multiplexer," *IEEE J. Selct. Areas in Commun.*, vol. 9, no. 3, pp. 378-387, April 1991.

[19] Park, Y.K., Cherkassky, V. and Lee, G., "Omega Network-Based ATM Switch with Neural Network-Controlled Bypass Queueing and Multiplexing," vol. 12, no. 9, pp. 1471-1480, Dec. 1994.

[20] Roberts, J.W. and Virtamo, J.T., "The Superposition of Periodic Cell Arrival Processes in an ATM Multiplexer," *IEEE Trans. Commun.*, vol. 39, no. 2, pp. 298-303, Feb. 1991.

[21] Rumelhart, D.E., G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," *Parallel Distributed Processing*, vol. 1, chap. 8, 1986.

[22] Sanchez-Barrera, J., " Call admission and routing in ATM networks," M.S. thesis, Dept. of ECE. LSU, Baton Rouge, LA, 1994.

[23] Vargas, C., Hegde, M., Naraghi-Pour, M. and Min, P., "Shadow Prices for LLR and ALBA," *IEEE/ACM Transactions on Networking*, vol. 4, no. 5, pp. 796-807, Oct. 1996.

[24] Yang T. and Tsang D.H.K., "A Novel Approach to Estimating Cell Loss Probability in an ATM Multiplexer Loaded with Homogeneous On-Off Sources," *IEEE Trans. Commun.*, vol. 43, no. 1, pp. 117-126, Jan. 1995.

[25] Yang T. and Li H., "Individual Cell Loss Probabilities and Background Effects in ATM Networks," *Proc. IEEE GLOBECOM '93*, pp. 1373-1379, 1993.

Table 1: 4-Node Network Paths.

4-Node Network Paths					
OD	**OD-type**		Paths	r-type	
	Call type 1	Call type 2		Call type 1	Call type 2
(0,2)	1	6	{2}	1	16
			{1,4}	2	17
			{3,11}	3	18
(1,2)	2	7	{4}	4	19
			{5,11}	5	20
			{6,2}	6	21
(3,2)	3	8	{11}	7	22
			{10,4}	8	23
			{9,2}	9	24
(0,3)	4	9	{3}	10	25
			{1,5}	11	26
			{2,7}	12	27
(0,1)	5	10	{1}	13	28
			{2,8}	14	29
			{3,10}	15	30

Table 2: First set of types parameters.

Call Type	BM	L	b	% in ON	BP
Type 1	0.5	100	2.5	40	1.25
Type 2	0.5	100	25	4	12.5

Table 3: Second set of types parameters.

Call Type	BM	L	b	% in ON	BP
Type 1	5	100	2.5	40	12.5
Type 2	0.5	100	25	4	12.5

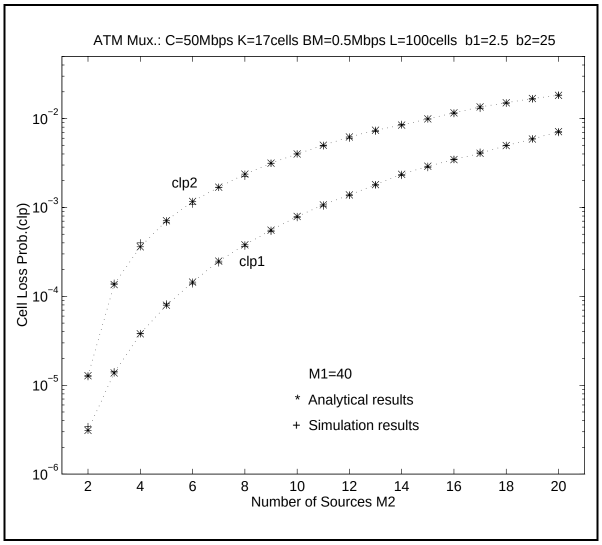

Figure 1: Simulation vs. analytical results from the model in Section 3.

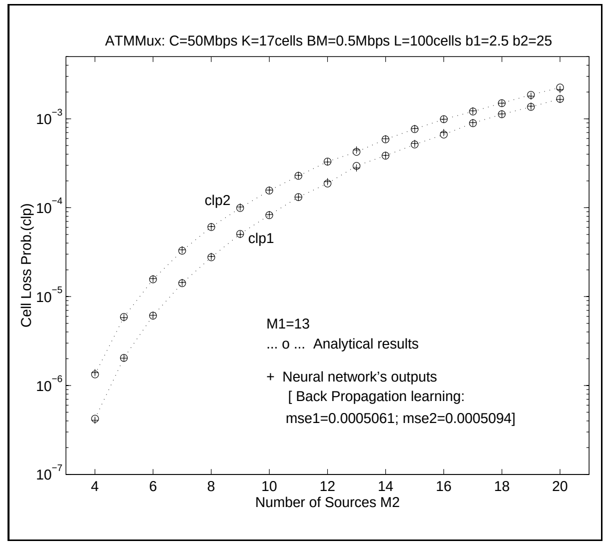

Figure 3: A comparison of the analytical results and neural network trained values.

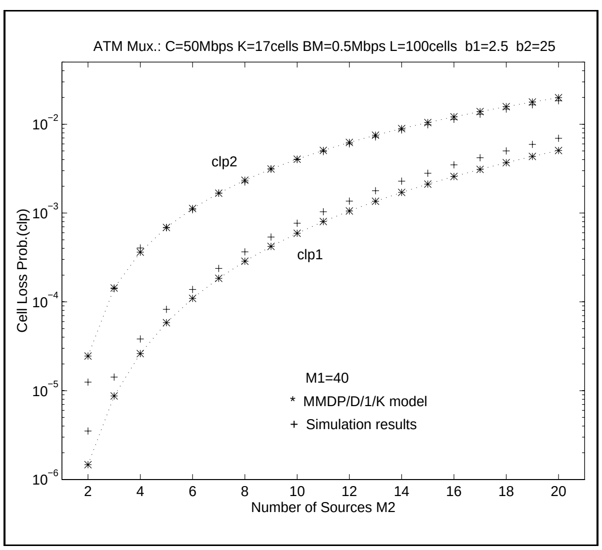

Figure 2: Simulation vs. analytical results from $MMDP/D/1/K$ model.

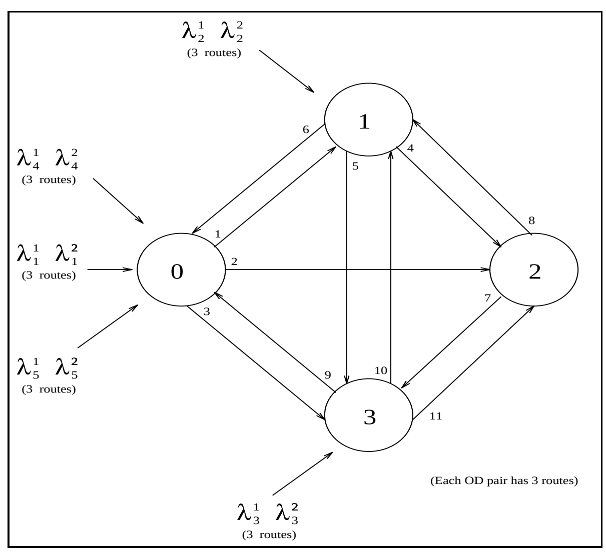

Figure 4: The four node network.

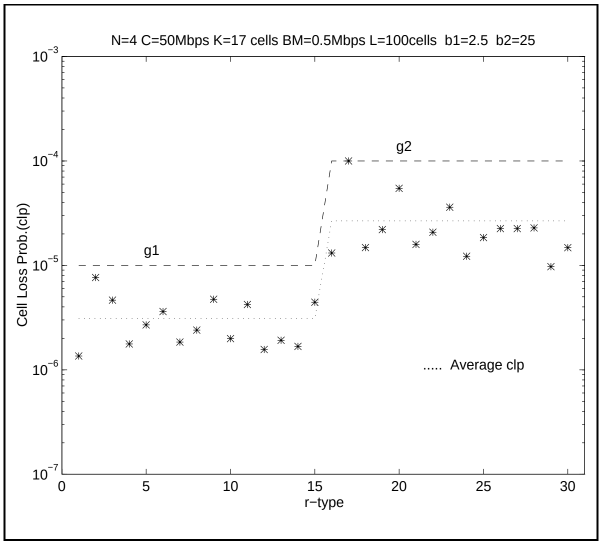

Figure 5: Cell Loss Probability vs. r-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$).

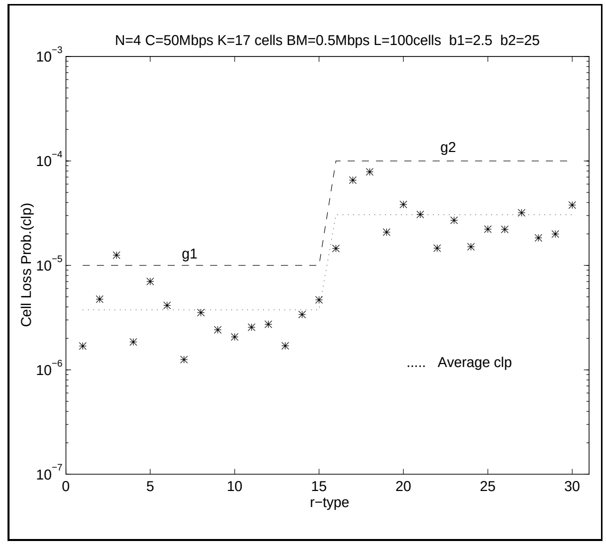

Figure 7: Cell Loss Probability vs. r-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$).

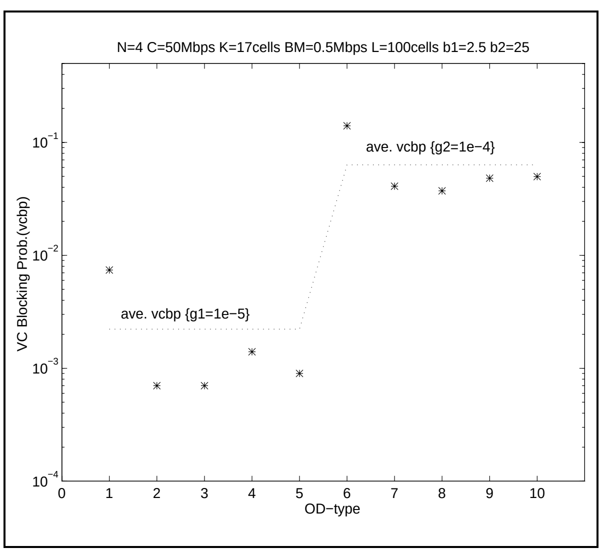

Figure 6: VC Blocking Probability vs. OD-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$).

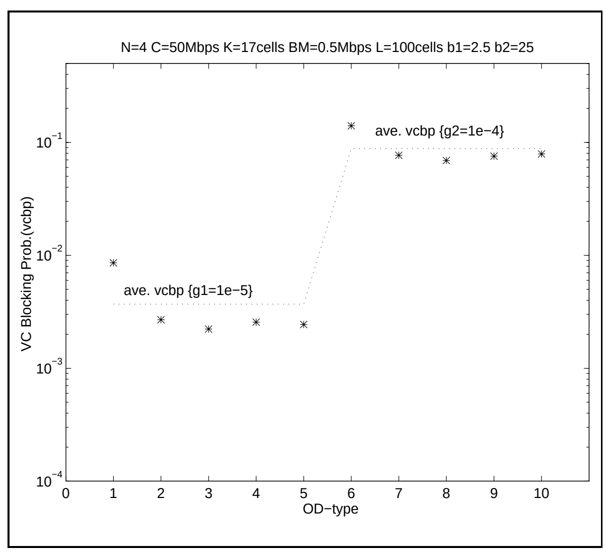

Figure 8: VC Blocking Probability vs. OD-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$).

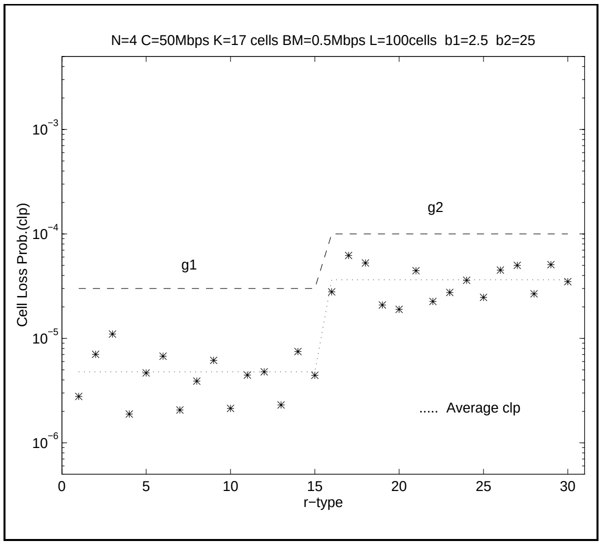

Figure 9: Cell Loss Probability vs. r-type (Algorithm 1, $g_1 = 3 \times 10^{-5}; g_2 = 10^{-4}$).

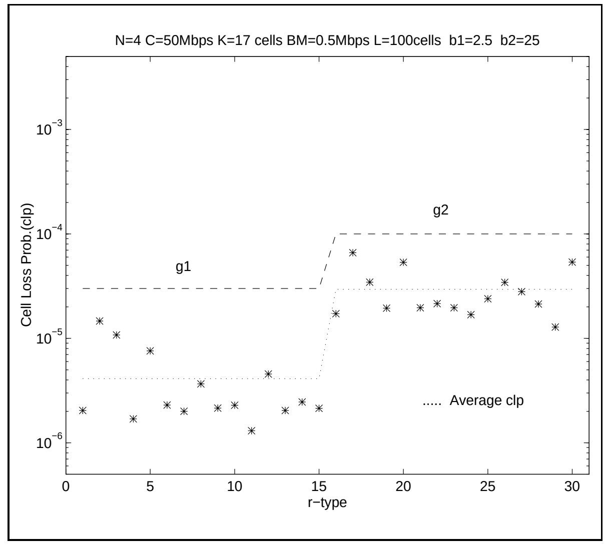

Figure 11: Cell Loss Probability vs. r-type (Algorithm 2, $g_1 = 3 \times 10^{-5}; g_2 = 10^{-4}$).

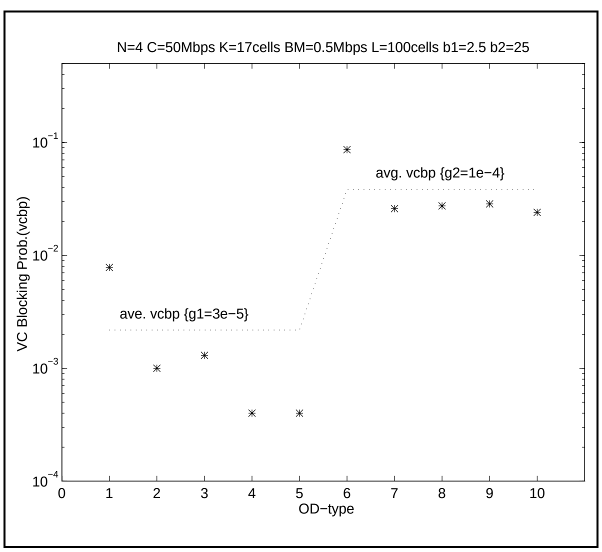

Figure 10: VC Blocking Probability vs. OD-type (Algorithm 1, $g_1 = 3 \times 10^{-5}; g_2 = 10^{-4}$).

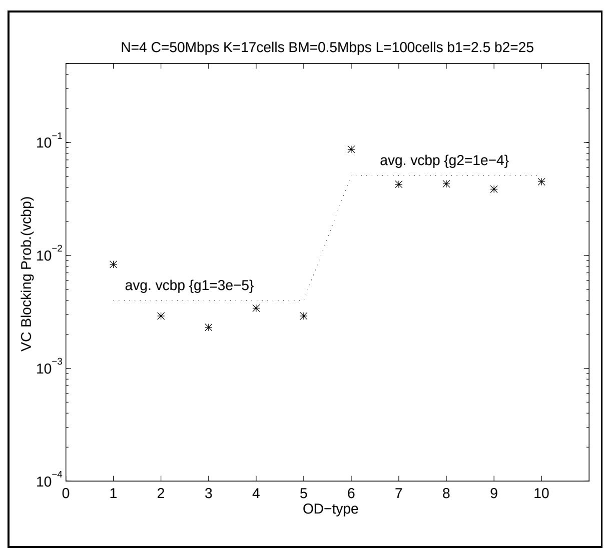

Figure 12: VC blocking probability vs. OD-type (Algorithm 2, $g_1 = 3 \times 10^{-5}; g_2 = 10^{-4}$).

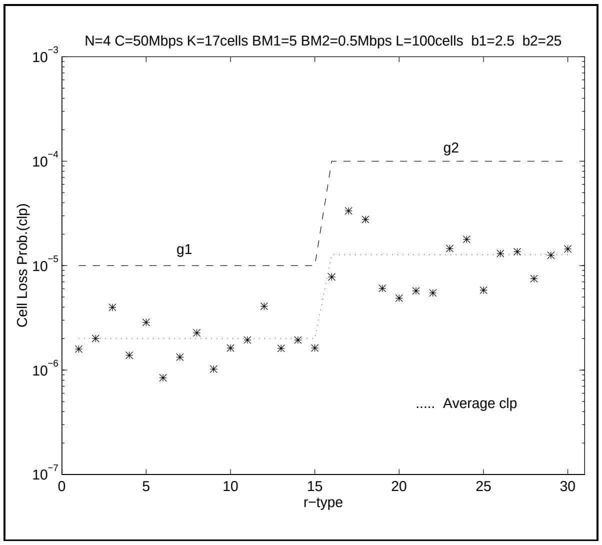

Figure 13: Cell Loss Probability vs. r-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$), second set of type parameters.

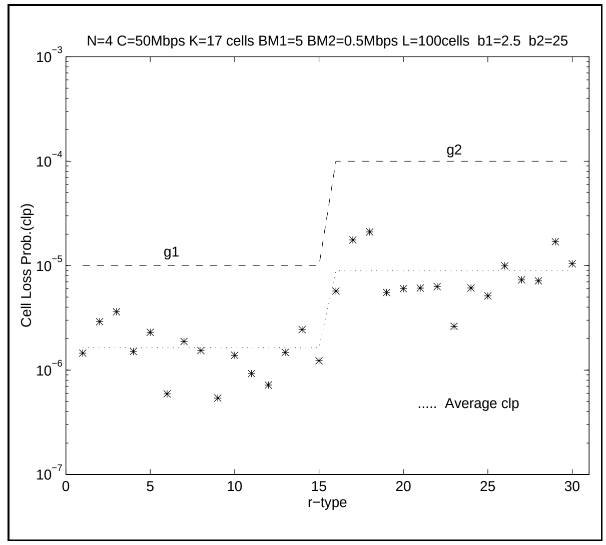

Figure 15: Cell Loss Probability vs. r-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$), second set of type parameters.

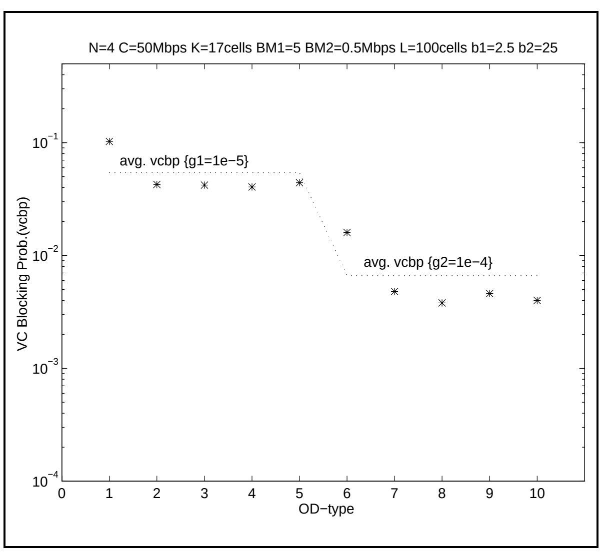

Figure 14: VC blocking probability vs. OD-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$), second set of type parameters.

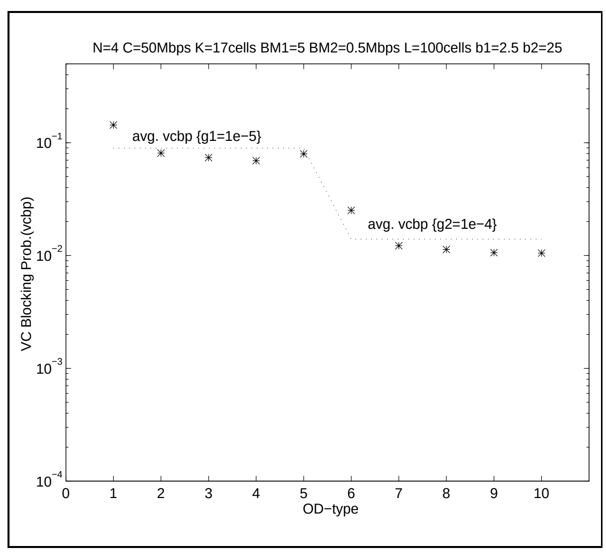

Figure 16: VC blocking probability vs. OD-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$), second set of type parameters.

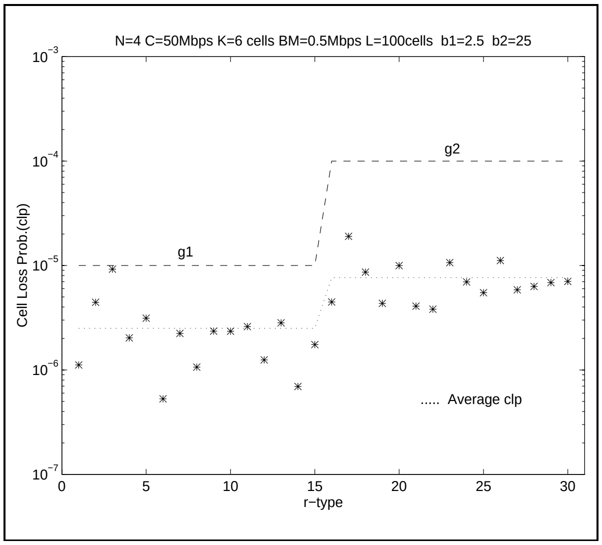

Figure 17: Cell Loss Probability vs. r-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$), buffer size $K = 6$.

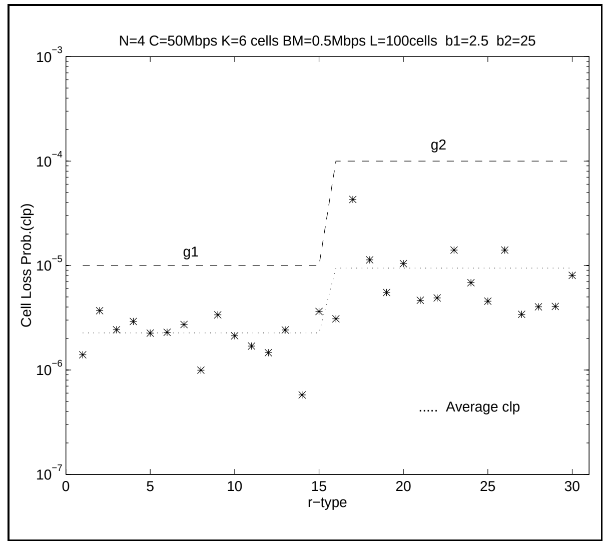

Figure 19: Cell Loss Probability vs. r-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$), buffer size $K = 6$.

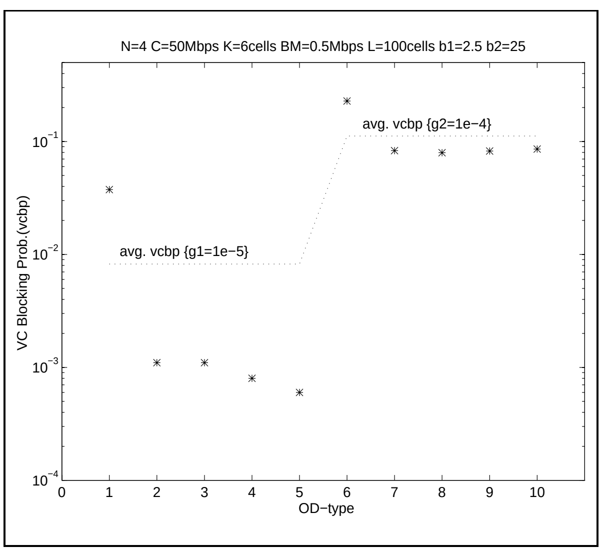

Figure 18: VC blocking probability vs. OD-type (Algorithm 1, $g_1 = 10^{-5}; g_2 = 10^{-4}$), buffer size $K = 6$.

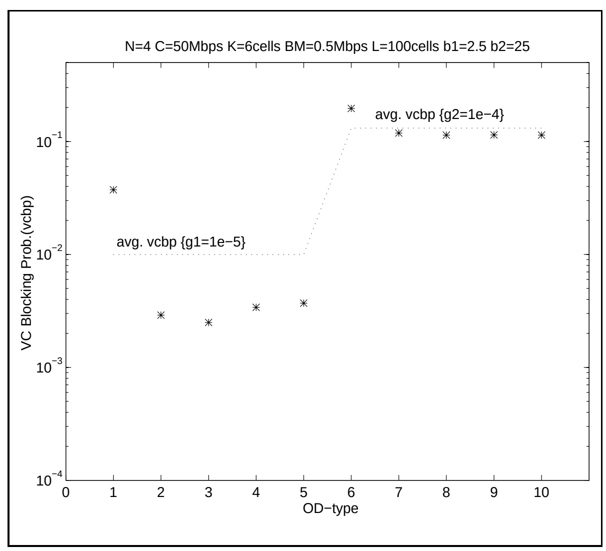

Figure 20: VC blocking probability vs. OD-type (Algorithm 2, $g_1 = 10^{-5}; g_2 = 10^{-4}$), buffer size $K = 6$.