Formula for the Required Capacity of an ATM Multiplexer

Markus Fiedler, University of Karlskrona/Ronneby*

Abstract

This contribution deals with a formula for the capacity an ATM multiplexer must at least have to accomodate the loss probability demands of all connections. So it can be used for connection admission control as well as for network resource management purposes. The formula is based on the bufferless fluid flow multiplexer model. It allows for a more exact capacity evaluation than if equivalent bandwidths are used on a per-connection basis. On the other hand, it merely requires a computational effort which is comparable to evaluating the mean of a given distribution. Indeed, the bottleneck is the convolution of probability density functions, on which the formula operates. Two steps to reduce the computational effort are proposed: a framework for convolution operations, consisting of pre-computed probability density functions, and a suitable truncation of the state space.

Keywords: ATM multiplexing; performance evaluation; capacity assignment; CAC; NRM; bufferless fluid flow model; convolution algorithm

1 Introduction

The *Asychrone Transfer Mode* (ATM) offers the possibility to save bandwidth on *burst level* when connections with *Variable Bit Rate* (VBR) are to be multiplexed. Instead of using the *Peak Cell Rate* (PCR) of all connections, a smaller value may be used in most cases, leading to a certain overallocation which may cause loss and/or delay within the cell streams of connections. The *required capacity* is that capacity a multiplexer which bundles VBR connections must at least have so that the *Quality of Service* (QoS) demands of *all* those connections are met. It is also known as *equivalent bandwidth* [19] or *equivalent capacity* [6].

One application of required capacity is *Connection Admission Control*. The decision on a connection requests will be the outcome of the comparison of required and available capacity. Based on the required capacity for existing connections, a fast CAC can be performed using the peak cell rate of the connection to be accepted, which will be updated by a more precise evaluation being carried out by a background algorithm [12]. Measurement-based CAC, which operates on measured cell rate density functions, has been proposed e.g. in [9], [11], [15], [16]; instead of loss probability calculations, evaluations of required capacity could be used to decide on a connection request.

*Institute of Telecommunications and Mathematics, Campus Gräsvik, S-371 79 Karlskrona, Sweden, e-mail: mfi@itm.hk-r.se, phone: +46-455-78161, mobile: +46-708-537339, fax: +46-455-78057

Another application is *Network Resource Management*. For a *Virtual Path Connection* (VPC) carrying a given number of *Virtual Channel Connections* (VCC), the required capacity denotes the capacity which should be allocated to that VPC, see e.g. [1], [10].

A wide-spread approach to approximate required capacities consists in the use of *equivalent bandwidths per connection*, see e.g. [3], [13] or [14]. Equivalent bandwidths lead to capacity estimations almost instantly, but they can easily over- or underestimate the required capacity, which may lead to suboptimal network utilization or — much worse — QoS degradation below the negotiated values. In any case, an optimized balance between customer demands and network utilization is demanded. The formulæ and methods presented in this paper could help to come closer to this goal without spending too much computational effort. They could serve as an alternative to exivalent bandwidths per connection if a higher precision and safety of capacity evaluations is required.

The paper is structured as follows: In section 2, the modelling of the system as well as the labelling of variables and functions is introduced. Section 3 presents formulæ for direct and other evaluation of required capacities and a performance assessment of these methods. Section 4 deals with possibilities to speed-up the convolution of the probability density functions, and section 5 describes a further reduction of execution times by applying a suitable truncation to the state spaces. The paper is concluded by section 6.

2 Modelling the system

The model used for the multiplexer is the well-known bufferless fluid flow model [9], [13] *et al.* The cell streams of the connections are modeled as flows whose intensities are described by the cell rate, which in general varies with time. The process governing the cell rate may be a continuous or a discrete-state process; in the following, the latter is assumed.

The buffer is assumed to be large enough to avoid loss on cell level, but not on burst level. Thus, delays are restricted to short-term delays due to quasi-simultaneous cell arrivals. Loss occurs only in *overload states*; these are states in which the cell rate R of all connections at the entrance of the multiplexer exceeds its capacity C. Its intensity is described by the *loss rate* $R_L = \max\{R - C, 0\}$. With this, the *loss probability* P_L is given by the quotient of expectations

$$P_L = \frac{\mathbf{E}R_L}{\mathbf{E}R}\,. \tag{1}$$

In case of a discrete distribution of the cell rate R with n_B states of probability $\pi(b) = \Pr\{R = r(b)\}, b = \{0, \ldots, n_B - 1\}$, this becomes

$$P_L = \frac{1}{m} \sum_{r(b)>C} \pi(b)(r(b) - C)\,, \tag{2}$$

with the *mean cell rate* $m = \mathbf{E}R = \sum_{b=0}^{n_B-1} \pi(b)r(b)$. The states are assumed to be sorted by ascending cell rates, i.e. $r(b-1) \leq r(b)$, and so the *peak cell rate* is given by $h = \max_b\{r(b)\} = r(n_B - 1)$.

2.1 Single connections

From the point of view of the bufferless fluid flow model, the behaviour of a connection is completely defined by its *state space*, denoted by the set $\{[r_g(b_g), \pi_g(b_g)] \,|\, b_g = 0, \ldots, n_{B,g} - 1\}$. The index g is the same for all connections with similar characteristics, i.e. similar state spaces; the probability density function of the cell rate is given by

$$f_{R_g}(r) = \sum_{b_g=0}^{n_{B,g}-1} \pi_g(b_g)\delta(r - r_g(b_g)) \,. \tag{3}$$

The simplest VBR connections are such ones with *On/Off characteristcs*, which are described by the state space $\{[0, 1-\alpha_g], [h_g, \alpha_g]\}$; the *activity factor* α_g is the probability that the connection delivers cells with peak cell rate h_g. *GCRA parameters* [14] may be used to identify these parameters in the way that the peak cell rate is taken over, i.e. $h_g = PCR_g$, and the mean rate is set to the sustainable cell rate, i.e. $m_g = \alpha_g h_g = SCR_g$ [17].

2.2 Group of connections

Connections of the same kind may be grouped; the size of such a *group*, i.e. the number of connections with similar characteristics is denoted by N_g.

As the processes governing the cell rates R_g of the connections are assumed to be independent, the probability density function $f_{R_{N,g}}(r)$ of the cell rate $R_{N,g}$ of all connections of group g may be obtained by convolution:

$$f_{R_{N,g}}(r) = \sum_{b_{N,g}=0}^{n_{B,N,g}-1} \pi_{N,g}(b_{N,g})\delta(r - r_{N,g}(b_{N,g})) \tag{4}$$

$$= \underbrace{f_{R_g}(r) * \ldots * f_{R_g}(r)}_{N_g} \,. \tag{5}$$

This is dealt with in detail in section 4.

A state space $\{[r_{N,g}(b_{N,g}), \pi_{N,g}(b_{N,g})] \,|\, b_{N,g} = 0, \ldots, n_{B,N,g} - 1\}$ arises, which in general would contain $n_{B,g}^{N_g}$ states and should be sorted in the way $r_{N,g}(b_{N,g} - 1) \leq r_{N,g}(b_{N,g})$. But as all connections within a group experience the same loss probability, states with the same cell rate $r_{N,g}$ need not be distinguished and may be gathered in one state. Thus, the size of the state space may be reduced significally. With On/Off connections, it merely consists of $n_{B,N,g} = N_g + 1$ states $\{[0, \pi_{N,g}(0)], [1h_g, \pi_{N,g}(1)], \ldots, [N_g h_g, \pi_{N,g}(N_g)]\}$. The cell rates are multiples of the peak cell rate of one connection h_g.

2.3 Integration of connections

The numbers of connections of more than one group are collected in the vector $\vec{N}$, whose dimension corresponds to the number of groups n_G.

If all connections should be integrated, i.e. share the same multiplexer capacity, the respective probability density function of the cell rate R is obtained by convoluting the probability density functions of the groups, i.e.

$$f_R(r) = \sum_{b=0}^{N_B-1} \pi(b)\delta(r - r(b)) \tag{6}$$

$$= f_{R_{N,1}}(r) * \ldots * f_{R_{N,n_G}}(r) \tag{7}$$

under the assumption of independency. The loss probabilities for different groups can be significantly different [18] and read

$$P_{L,g} = \frac{1}{m_{N,g}} \sum_{r(b)>C} \pi(b) l_{N,g}(b)(r(b) - C)\,, \tag{8}$$

with the *loss partition function* $l_{N,g}(b)$ which is given by

$$l_{N,g}(b) = \frac{r_{N,g}(b)}{r(b)} \tag{9}$$

under the assumption that the intensity of loss in group g is proportional to the cell rate delivered by group g in overload state b [9]. So the state space, whose size is given by $n_B = \prod_{g=1}^{n_G} n_{B,N,g}$, must contain information not only about the overall cell rate in state b, but also about the values of the loss distribution functions, so that it becomes the set $\{[r(b), l_{B,1}(b), \ldots, l_{B,n_G}(b), \pi(b)] \,|\, \forall b\}$.

If it is sufficient to know the *overall loss probability* given by (2), the size of the state space $\{[r(b), \pi(b)] \,|\, \forall b\}$ reduces itself to the number of different cell rates.

3 Evaluation of the required capacity

The required capacity is that value of C so that *all* QoS values for *all* connections are fulfilled. Unfortunately, its evaluation is mostly not straightforward; it may be compared with finding an appropriate cause (= capacity) so that a desired result (= QoS) is obtained.

3.1 The formula for direct evaluation

For the bufferless fluid flow model, a closed formula for the required capacity may be deducted from (2) with a *loss probability objective* δ_e:

$$C_{req} = \frac{\sum_{r(b)>C_{req}} \pi(b) r(b) - \delta_e m}{\sum_{r(b)>C_{req}} \pi(b)}\,. \tag{10}$$

For a QoS objective $\delta_{e,g}$ of group g, the formula reads

$$C_{req,g} = \frac{\sum_{r(b)>C_{req}} \pi(b) l_{N,g}(b) r(b) - \delta_{e,g} m_{N,g}}{\sum_{r(b)>C_{req}} \pi(b) l_{N,g}(b)}\,. \tag{11}$$

To fulfil *all* QoS demands of all groups, the required capacity must be maximized:

$$C_{req} = \max_g \{C_{req,g}\} \tag{12}$$

The formulæ (10f) are implicit in $C_{req(,g)}$, but this is restricted to the lower bound of the sums, so that a suitable evaluation helps to overcome this problem. Let $\check{b}$ be the smallest index in the sums in (10f). Beginning with the maximal index value $\check{b} = n_B - 1$, one state after another is taken into the sums, i.e. $\check{b}$ is decreased, until the quotient fulfils the condition

$$r(\check{b} - 1) \leq C_{req(,g)} < r(\check{b})\,, \tag{13}$$

which means that the required capacity has been reached exactly. This method is referred to as *direct evaluation of required capacity*.

3.2 Recursive method

The first alternative bases on a recursive formulation of the loss probability; such a *recursive method* has been proposed by Mitrou *et al* [12]. With $r(b+1) > r(b)$ and the complementary probability distribution function $G(r) = \Pr\{R > r\}$, formulæ (2) and (8) may be written as:

$$P_L(C = r(b)) = P_L(C = r(b+1)) + \frac{r(b+1) - r(b)}{m} G(r(b)) \,, \tag{14}$$

$$P_{L,g}(C = r(b)) = P_{L,g}(C = r(b+1)) + \frac{r(b+1) - r(b)}{m_{N,g}} \sum_{b':r(b')>r(b)} \pi(b') l_{N,g}(b') \,, \tag{15}$$

$$P_{L(,g)}(C = h) = 0 \,. \tag{16}$$

The evaluation starts with $C = h$, and as soon as the QoS value exceeds the objective value, an upper bound for the loss probability is found:

$$P_{L(,g)}(C = r(\hat{b})) > \delta_{e(,g)} \Rightarrow C_{req(,g)} \leq r(\hat{b}+1) \,. \tag{17}$$

3.3 Combined method

Once the upper bound index $\check{b} = \hat{b} + 1$ has been determined by the recursive method, the exact value can be refined by using the formulæ of the direct method (10) or (11) with $b = \check{b}, \ldots, n_B - 1$ in both sums. This *combined method* merely uses another way of overcoming the implicitness of (10) or (11) by searching the right "position" of the required capacity within the state space via QoS evaluations. Note that the difference between $C_{req(,g)}$ and its upper bound $r(\check{b})$ may be substantial if the cell rates $r(\hat{b})$ and $r(\check{b})$ are widely spread.

3.4 Interval-based search method

Another alternative is an *interval-based search* which also evaluates the QoS. Beginning with a lower bound $\hat{C}_{req} = m$ and an upper bound of $\check{C}_{req} = h$, the search interval is splitted, and dependent on the comparison of the QoS for C in the middle of $\hat{C}_{req}$ and $\check{C}_{req}$ with the given objective δ_e resp. $\delta_{e,g}$, either the upper or the lower subinterval is chosen for the next iteration step, that means that either $\hat{C}_{req}$ or $\check{C}_{req}$ is set to C. See [2], [5] and [19] for a more detailed description of the algorithm. Here, the iteration stops when the search interval has reached a size of 0.1 % of a given base cell rate r_0. The upper bound is returned which represents a safe approximation of the required capacity to be evaluated. In distinction from the other methods, the state space needn't be sorted.

3.5 Comparison

Table 1 contains execution times for sake of a qualitative comparison of the different approaches just described. The respective figures are the averages of ten times 1000 calculations and have been recorded on a Sun Ultra10 Workstation using the `gprof` tool. They should merely illustrate magnitudes and serve as trend indicators because they depend on the capacity value to be evaluated (which is determined by the distribution as well as by the QoS demand), on the computer hardware, the implementation and the

N_g	$\frac{C_{req}}{h_g}$	mean execution times					
		evaluation of mean	evaluation method				
			direct	recursive	combined	iterative	
1000	153.86	0.15 ms	0.31 ms	0.17 ms	0.32 ms	3.22 ms	(20)
4000	501.70	0.54 ms	1.02 ms	0.48 ms	0.97 ms	12.83 ms	(22)
7000	832.04	0.89 ms	1.66 ms	0.83 ms	1.62 ms	22.74 ms	(23)
10000	1156.13	1.30 ms	2.20 ms	1.22 ms	2.28 ms	32.95 ms	(24)

Table 1: Mean execution times for differen methods of required capacity evaluation.

granularity of `gprof` measurements. The distributions are those of N_g On/Off connections with activity factor $\alpha_g = 0.1$, the requested loss probability was 10^{-9}. For sake of comparison, the execution times needed to calculate the mean values of the respective distributions are also given.

The computational effort both the direct method and for the combined method is about double as high as for the recursive method. The reason for this behaviour lies in the fact that the effort to evaluate the quotient in (10) is higher than checking the QoS based on the recursive formula (14). Anyway, these execution times are of the same magnitude as the calculation time for the mean, whereas those of the iterative solution are one magnitude larger. Even for 10000 On/Off connections, it took less than three milliseconds to get the exact capacity value.

So the use of the combined method is recommended because of its flexibility to deliver both a safe upper bound for the required capacity in shorter time and a refined value based on the direct formula in a second step.

The next sections will address the question how to get the distributions of cell rates in an efficient way.

4 Convolution

The effort to compute the probability density functions mentionned above strongly depends on the sizes of the groups N_g and their number n_G, see (5) and (7). Because it is more likely that quite a small number of groups with some connections exists — perhaps even only one — and the convolution operation for the integration is essentially the same as within the groups (except for a different characterization of the connections), we shall focus on the group.

4.1 Convolution framework for a group

Performing $N_g - 1$ convolutions to get the probability density function for a group of N_g connections out of that for one connection as shown in (5) could be a mess, cf. the example in subsection 4.4. On the other hand, if the probability density function for $N_g - 1$ is already known, which is a typical situation during CAC, the effort is not great at all. The framework for convolutions of probability density functions within a group which is presented here allows both for a reduction of the number of convolution operations to get the probability density function for N_g connections as well as for an increase or a decrease of N_g by 1.

The basic idea is to store specific precalculated probability density functions, namely those for $N_g = 2^{k_g}$, $k_g = 0, 1, \ldots, l_g - 1$, so that a range for the number of connections

$$N_g \in \{1, \ldots, 2^{l_g} - 1\} \tag{18}$$

may be covered with l_g stored functions. On the other hand, a maximal number $N_{g,max}$ implies a need of

$$l_g = \lfloor \mathrm{lb} N_{g,max} \rfloor + 1 \tag{19}$$

precalculated functions. The memory consumption is that of $2^{l_g} - 1$ floats (at least with double precision).

Bit k_g of the binary representation $(N_g)_2$ of N_g delivers the information whether function $f_{R_{2^{k_g},g}}(r)$ is needed to get the desired function $f_{R_{N,g}}(r)$ or not, which easens a realization in hardware. The algorithm reads as follows:

Initialize $f_{R_{N,g}}(r) := \delta(r)$;
For $k_g := 0$ **to** $l_g - 1$:
  If $(N_g)_2^{[k_g]} = 1$:
    Let $f_{R_{N,g}}(r) := f_{R_{N,g}}(r) * f_{R_{2^{k_g},g}}(r)$.

For instance, the probability density function for 100 connections will be composed of

$$f_{R_{100,g}}(r) = f_{R_{4,g}}(r) * f_{R_{32,g}}(r) * f_{R_{64,g}}(r)\,. \tag{20}$$

An example for the performance is given in subsection 4.4.

In [7], the idea of storing intermediate results on a binary basis is used to easen the computational burden of calculating individual performance measures for different traffic streams on connection level.

4.2 Prerequisities

The algorithms presented in the following subsection are based on the assumption that all cell rates being involved can be expressed as multiples of a base cell rate Δr [12]. Otherwise, the convolution cannot simply operate on integer indices, and in most cases the outcoming state space has to be sorted; both implies a greater computational effort. The assumption is in any case valid for a group of On/Off connections. According to requirement, cell rates must be quantized to multiples of Δr; a quantized cell rate should not be smaller than the original value to avoid underestimations of the required capacity.

As the use of the loss partition function in the case of integration of connections leads to a much higher effort despite of the convolution operation (7), see subsection 2.3, it should be avoided. A bundle of possibilities to treat an integration without using the loss partition function is listed below:

1. The overall loss probability could serve as QoS objective δ_e; if there are different QoS objectives, δ_e should be set to the smallest, i.e. most stringent value.

2. If the latter possibility is used as an approximation for the required capacity based on individual loss probabilities, it is not guaranteed that the most stringent individual QoS demand will be met. To overcome this problem, the required capacity based on the overall loss probability C'_{req} could be increased by the maximal peak cell rate

of all connections, which is a heuristic upper bound for the required capacity based on the most stringent QoS demand:

$$\max_g\{C_{req,g}\} \leq C'_{req} + \max_g\{h_g\}\,. \tag{21}$$

3. The problem of individual QoS demands might also be solved by splitting the capacity into subcapacities $C_{req}(N_g)$ which are to be evaluated per group and summed up in the end; the inequality

$$C_{req}(\vec{N}) \leq \sum_{g=1}^{n_G} C_{req}(N_g) \tag{22}$$

holds. Another advantage consists in the fact that the convolution operation (7) does not need to be carried out, which easens the computational burden and helps to avoid huge state spaces due to a lot of groups and/or a lot of connections within them. Not at least, different base cell rates Δr_g could be defined for different groups.

Besides these apparent advantages, it is to observe that this *groupwise multiplexing* renounces to gain achieved by integration, which could be significant dependent on the numbers and characterizations of connections being involved [4], [5].

The state spaces will be redefined in multiples of the base cell rate Δr. For instance, a state space $\{[0,\pi_g(0)],[\frac{2}{3}h_g,\pi_g(1)],[h_g,\pi_g(2)]\}$ will be translated in $\{[0,\pi_g(0)],[1,0],[2,\pi_g(1)],[3,\pi_g(2)]\}$ with $\Delta r = \frac{1}{3}h_g$; the respective probability density function is given by

$$f'_{R_g}(r) = \sum_{b'_g=0}^{n'_{B,g}-1} \pi'_g(b'_g)\, b'_g\, \Delta r \tag{23}$$

with the number of states $n'_{B,g}$ and an eventually modified peak cell rate $h'_g = (n'_{B,g}-1)\Delta r$. Note that this renormation has to be taken care of when required capacities are to be evaluated. When the stroke symbol appears in the following, it should remind the reader of the assumptions presented in this subsection.

4.3 Convolution and Deconvolution

Based on these assumptions, the *convolution algorithm* to convolute the probability density functions $f'_1(r) * f'_2(r) = f'(r)$ reads as follows:

For $i := 0$ **to** $n'_{B,1} + n'_{B,2} - 2$
  Initialize $\pi'(i) := 0$;
For $i_1 := 0$ **to** $n'_{B,1} - 1$
  For $i_2 := 0$ **to** $n'_{B,2} - 1$
    $\pi'(i_1 + i_2) := \pi'(i_1 + i_2) + \pi'_1(i_1)\pi'_2(i_2)$.

From the numerical point of view, the convolution operation is very stable [8], so that there's no problem of reaching the probability density function for N_g connections by successive convolutions with that for a single connection $f_{R,g}(r)$ based on the recursive formula

$$f_{R_{N_g,g}}(r) = f_{R_{N_g-1,g}}(r) * f_{R_g}(r)\,. \tag{24}$$

This is of interest when N_g is raised by one after a connection acceptance.

If a connection is released, the probability density function for the N_g remaining connections may be obtained by deconvolution

$$f_{R_{N_g,g}}(r) = f_{R_{N_g+1,g}}(r) \overline{*} f_{R_g}(r)\,. \tag{25}$$

The *deconvolution algorithm* to deconvolute $f'(r) \overline{*} f'_1(r) = f'_2(r)$ reads

For $i_2 := 0$ **to** $n'_{B,2} - 1$
  Let $\pi'_2(i_2) := \pi'(i_2)$;
  For $j_1 := 1$ **to** $\min\{i_2, n'_{B,1} - 1\}$
    Let $\pi'_2(i_2) := \pi'_2(i_2) - \pi'_1(j_1)\pi'_2(i_2 - j_1)$;
  Let $\pi'_2(i_2) := \pi'_2(i_2)/\pi'_1(0)$.

It is far less stable than the convolution algorithm [8] due to the facts that *all* values $\pi'_2(0), \ldots, \pi'_2(i_2 - 1)$ which have just been computed before are needed to compute the topical value $\pi'_2(i_2)$ in the sense of a difference, and that a division by a possibly small value $\pi'_1(0)$ occurs.

The deconvolution can also start at the other end of the density function, which is to be preferred if $\pi'_1(n'_{B,1} - 1) > \pi'_1(0)$ holds. The respective algorithm becomes

For $i_2 := n'_{B,2} - 1$ **downto** 0
  Let $i := i_2 + n'_{B,1} - 1$;
  Let $\pi'_2(i_2) := \pi'(i)$;
  For $j_1 := \max\{0, i - n'_{B,2} + 1\}$ **to** $n'_{B,1}$
    Let $\pi'_2(i_2) := \pi'_2(i_2) - \pi'_1(j_1)\pi'_2(i - j_1)$;
  Let $\pi'_2(i_2) := \pi'_2(i_2)/\pi'_1(n'_{B,1} - 1)$.

Except for cases with small numbers of connections and quite high state probabilities, it is recommended to use deconvolution only once or twice before a *refresh* takes place. Such a refresh means a convolution of the desired probability density function for N_g connections using the framework described in subsection 4.1.

4.4 Examples

A first example compares execution times of two ways of getting the probability density function for $N_{g,max}$ On/Off connections

1. by using the convolution framework proposed in subsection 4.1;
2. by applying successive convolutions with the probability density function for one connection in the sense of (24), until $N_{g,max}$ is reached.

The mean execution times, which have been measured in principle as described in subsection 3.5, are reported in table 2. Due to the longer execution times, the number of evaluations per measurement has been reduced from 1000 to 1 (resp. 10 for $N_{g,max} = 1000$ and the framework to reduce the influence of `gprof` granularity). Also, the mean execution times to build up the base of probability density functions for $N_{g,max}$, which is only needed at set-up time, has been noted. The gain represents the factor by which the mean execution time has been lowered, once the build-up time has been spent. The more connections $N_{g,max}$ are to be considered, the greater becomes the advantage of using the

$N_{g,max}$	mean execution time			gain
	framework		successive convolutions	
	build-up	use		
1000	14 ms	54 ms	255 ms	4.7
4000	0.34 s	1.51 s	5.33 s	3.5
7000	1.48 s	3.75 s	16.37 s	4.4
10000	5.44 s	4.13 s	32.85 s	8.0

Table 2: Mean execution times for the determination of the probability density function for $N_{g,max}$.

k	N_g	mean execution time
5	63	0.28 ms
6	127	1.03 ms
7	255	3.81 ms
8	511	15.02 ms

Table 3: Worst-case mean execution times for k convolutions of the first $k+1$ probability density functions of the convolution framework.

convolution framework — in the case of $N_{g,max} = 10000$ connections, it almost reaches one magnitude.

For all that, the time to convolute the density functions is about two to three magnitudes (!) higher than the time needed to calculate the required capacity, cf. table 1. Table 3 shows some related mean execution times. The numbers of connections have been chosen to be $N_g = 2^{k+1} - 1$ with $k = 5, \ldots, 8$ to cover the worst case of k convolutions to be computed to get the respective probability density function. For $N_g = 127$ or 6 successive convolutions, the magnitude of one millisecond is reached. Note that the mean execution time for $N_g = 511$ exceeds that for $N_g = 1000$ (table 2).

Although the successive convolution leads to much longer execution times than that using the respective framework, it can be of advantage in cases when a lot of sucessive incrementations of N_g happen. If, for instance, On/Off probability density functions up to $N_{g,max} = 1000$ are to be evaluated using a convolution based on the framework for each N_g, a mean execution time of 13.87 s is obtained, which is about fifty times as much as that using successive convolution, cf. table 2. Mean execution times for single incrementals are reported in subsection 5.2, table 5.

5 Truncated density functions

To truncate a probability density function means to take away states whose probabilities are so small that a performance measure won't be affected by them anymore. Truncation has also been proposed by Iversen and Stepanov [7] for performance evaluations on connection level.

5.1 Truncation criterion

The truncation criterion will be deducted for individual loss probabilities (8); a similar argumentation may be applied for the overall loss probability. The loss probability (8)

N_g	$\frac{C_{req}}{h_g}$	mean evaluation time			gain	
		convolution (framework)	capacity evaluation		convolution (framework)	combined evaluation
			combined	recursive		
1000	153.86	5.54 ms	0.015 ms	0.010 ms	9.7	21
4000	501.70	34.3 ms	0.031 ms	0.014 ms	44	31
7000	832.04	67.9 ms	0.042 ms	0.025 ms	55	38
10000	1156.13	94.5 ms	0.045 ms	0.023 ms	43	51

Table 4: Mean execution times and gains obtained by truncation of the state space.

can be split in parts

$$P_{L,g} = \sum_{r(b)>C} P_{L,g}(b)\,. \tag{26}$$

Summand $P_{L,g}(b)$ will not contribute to the outcome $P_{L,g}$ if it is smaller than the product of a constant ε, given by the numerical representation of floats in the computer, and the loss probability $P_{L,g}$:

$$P_{L,g}(b) < \varepsilon P_{L,g}\,. \tag{27}$$

As the loss rate of a group of connections won't exceed its peak cell rate, i.e. $l_{N,g}(b)(r(b)-C) \le h_{N,g}$, the criterion for neglecting a state b

$$\pi(b) < \varepsilon P_{L,g} \frac{m_g}{h_g} \tag{28}$$

follows, which is based on its state probability. If the overall loss probability is to be considered, the criterion reads

$$\pi(b) < \varepsilon P_L \frac{m}{h}\,. \tag{29}$$

For evaluations of required capacity, P_L resp. $P_{L,g}$ are known in advance. The value $\varepsilon = 10^{-17}$ is recommended for a float representation with double precision.

5.2 Examples

The first example refers to subsection 3.5, table 1 and subsection 4.4, table 2. Again, mean execution times for convolution based on the framework of subsection 4.1 and for the recursive and combined method (subsections 3.3 and 3.4) have been measured. N_g On/Off connections with activity factor $\alpha_g = 0.1$ have been taken into account, the loss probability demand was 10^{-9}. According to (28f), *every* probability density function was truncated to values $\pi(b) \ge 10^{-17} \cdot 10^{-9} \cdot 0.1 = 10^{-28}$.

Compared to table 1, the capacity values haven't been changed by the truncation — but the mean execution times have descreased by one to two magnitudes (cf. gain values in table 4) both for the convolution (cf. also table 2) as well as for capacity evaluation. Note that the gain achieved by truncation depends on the characteristics of the connections, the used float representation and the loss probability demand.

As before, the capacity evaluation lasts only a very small share of the time which is needed for the convolution. So if the number of connections is incremented by one, which leads to a much faster convolution of the respective density functions, or if this density

N_g	mean execution time	
	truncation:	
	no	yes
$1024 \rightarrow 1025$	0.59 ms	0.12 ms
$2048 \rightarrow 2049$	1.48 ms	0.19 ms
$4096 \rightarrow 4097$	2.64 ms	0.37 ms
$8192 \rightarrow 8193$	5.37 ms	0.74 ms

Table 5: Mean execution times for the increase of the number of connections without and with truncation.

function has been measured (compare table 5 with tables 2–4), the computational effort for the determination of the required capacity is not high at all, especially when the state space has been truncated.

At least, the second example — table 5 — shows the advantage of truncation for the increase of the number N_g of On/Off connections with $\alpha_g = 0.1$ by one (24), based on probability density functions which are stored in the framework. Without truncation, a mean execution time of one millisecond is kept for more than 1000 connections, whereas with truncation, this number is raised to more than 8000.

6 Conclusion

In this paper, formulæ for the direct evaluation of required capacities based on the bufferless fluid flow model have been shown and compared. A combined method which is able to deliver a safe upper bound after about half of the execution time needed to calculate the exact value is recommended. Indeed, the bottleneck is the convolution operation to get the probability density function on which the formulæ operate. A framework which speeds up the convolution of probability density functions for connections with same characteristics is shown; a further speed-up may be reached by a suitable truncation of the state spaces.

Execution time measurements show that it is possible to calculate exact values for required capacities for bufferless multiplexing in the magnitude of milliseconds, which means that exact evaluations (in the sense of bufferless multiplexing) are not necessarily condemned to background calculations; they may serve as alternatives to capacity approximations based on equivalent bandwidths if a higher accuracy is requested for CAC and NRM.

References

[1] Bolla, R.; Davoli, F.; Marchese, M.: *Bandwidth allocation and call admission control in high-speed networks.* In: IEEE Communications Magazine, Vol. 35, No. 5, May 1997, pp 130–137.

[2] Chan, J. H. S.; Tsang, D. H. K.: *Bandwidth allocation of multiple QoS classes in ATM environment.* In: Proceedings of INFOCOM'94, Vol. 1, pp 360–367.

[3] Elwalid, A. I.; Mitra, D.: *Effective bandwitdh of general Markovian traffic sources*

and admission control of high speed networks. In: IEEE/ACM Transactions on Networking, Vol. 1, No. 3, June 1993, pp 329–343.

[4] Fiedler, M.: *Erforderliche Kapazität beim Multiplexen von ATM-Verbindungen.* Ph.D. thesis, Saarbrücken 1998. Utz, München, ISBN 3-89675-385-1.

[5] Fiedler, M.: *Direct evaluation of required capacity for ATM multiplex.* COST temporary document 257TD(98)27, May 1998, 13 pp.

[6] Gallassi, G.; Rigolio, G.: Fratta, L.: *ATM: Bandwidth asssignment and bandwidth enforcement policies.* In: Proceedings of GLOBECOM'89, Vol. 3, pp 1788–1793.

[7] Iversen, V. B.; Stepanov, S. N.: *The usage of convolution algorithm with truncation for estimation of individual blocking probabilities in circuit-switched telecommunication networks.* In: Proceedings of the ITC-15, 1997, Vol. 2b, pp 1327–1336.

[8] Iversen, V. B.: *Teletraffic Engineering.* Preprint, 1998.

[9] Kröner, H.; Renger, T.; Knobling, R.: *Performance Modelling of an Adaptive CAC Strategy for ATM Networks.* In: Proceedings of ITC-14, 1994, Vol. 1b, pp 1077–1088.

[10] Larsson, S.-O.; Arvidsson, Å.: *A comparison between different approaches for VPC bandwidth management.* COST temporary document 257TD(97)29, May 1997, 10 pp.

[11] Lee, T. H.; Lai, W. M.; Duann, S.-T.: *Real time call admission control for ATM networks with heterogeneous bursty traffic.* In: Proceedings of ICC'94, Vol. 1, pp 80–85.

[12] Mitrou, N. M. u.a.: *Statistical multiplexing, bandwidth allocation strategies and connection admission control in ATM networks.* In: European Transactions on Telecommunications, Vol. 5, No. 2, March–April 1994, pp 161–175.

[13] Roberts, J. W. (ed.): *EUR 14152 — COST 224 — Performance evaluation and design of multiservice networks. final report.* Commission of the European Communities. Luxembourg: Office for Official Publications of the European Communities, October 1992. ISBN 92-826-3728-X.

[14] Roberts, J. W., Mocci, U., Virtamo, J. (ed.): *Broadband network teletraffic: performance evaluation and design of broadband multiservice networks; final report of action COST 242.* Berlin, Heidelberg: Springer, 1996, ISBN 3-540-61815-5.

[15] Saito, H.; Shiomoto, K.: *Dynamic call admission control in ATM networks.* In: IEEE Journal on Selected Areas in Communications, Vol. 9, No. 7, Sept. 1991, pp 982–989.

[16] Saito, H.: *Dynamic resource allocation in ATM networks.* In: IEEE Communications Magazine, Vol. 35, No. 5, May 1997, pp 146–153.

[17] Siebenhaar, R.: *Verkehrslenkung und Kapazitätsanpassung in ATM-Netzen mit virtuellen Pfaden.* Ph.D. thesis, München: Utz, 1996, ISBN 3-931327-49-3.

[18] Yang, T.; Li, H.: *Individual cell loss probabilities and background effects in ATM networks.* In: Proceedings of ICC'93, Vol. 3, pp 1373–1379.

[19] Zhang, Z.; Acampora, A. S.: *Equivalent bandwidth for heterogeneous sources in ATM networks.* In: Proceedings of ICC'94, Vol. 2, pp 1025–1031.