Date: January 20, 1997 COST257TD(97)15
Issue: 1

Remarks on the effectiveness of dynamic VP bandwidth management

Jorma Virtamo, Helsinki University of Technology

Samuli Aalto, VTT Information Technology

1 Introduction

In report [1] we considered the technical problem of calculating the time dependent blocking probabilities for the dynamic VP bandwidth management scheme developed by Mocci et al. [2]-[5]. In this scheme, the bandwidth allocation for various VP's are updated at regular time intervals. At the beginning of an interval the system occupancy is observed and new VP capacity allocation is done in such a way that the expected time average of the blocking probability in the ensuing interval will be less than a predefined limit. In this report we assess the advantages and disadvantages of the scheme trying also to give a quantitative idea about the savings that can be achieved.

2 Comparisons

In the VP management scheme referred to one tries to find an optimal setting between two extreme cases:

- Static allocation where a constant bandwidth is reserved for each VP such that the average blocking probability is below the prescribed level ϵ, i.e. the capacity is determined from the Erlang's formula.
- Dynamic call-by-call allocation, where VP's are not used at all, or, if used, their capacities are adjusted at every call set-up and tear-down to accommodate precisely the number of calls offered (as far as capacity is available).

The static allocation is simple from the management point of view. The VP's are set up once and for all, and at the call set-up phase one only has to check the available capacity in a VP in order to decide whether a call can be accepted or not. Its disadvantage is that some capacity is wasted as statistical multiplexing is allowed only within the VP. Also it is unresponsive to the variations in the offered traffic level; uncertainties in this respect will necessitate the reservation of an even larger capacity for each VP.

The call-by-call allocation makes full use of statistical multiplexing and requires the least total capacity in average for a given ϵ. However, this is achieved only at the cost of complex call processing: for each new offered call one has to check the available capacity on each link along the route of the call. In addition, VP management load becomes dependent on the call arrival rate: the more traffic, the more frequent reallocation is needed.

The VP management scheme of Mocci et al. balances between these trends: the VP bandwidth allocations are updated only at regular intervals, chosen to be long in comparison with the mean interarrival time, in order to relieve the call processing load. Some capacity saving will be obtained in comparison with the static allocation but not as much as in the case of call-by-call allocation. Another advantage comes from the fact that the VP management load becomes independent of the traffic load.

Let us consider a VP with an offered traffic intensity of a (for simplicity let us assume that all the calls belong to the same class and have equal effective bandwidths). The saving that can be achieved is upper bounded by the difference in capacity allocations in the worst case (static allocation) and the best case call-by-call allocation with an infinite reservoir. The former is given by the Erlang capacity, i.e., least N such that $\mathrm{Erl}(N, a) < \epsilon$, and for the latter the mean allocated capacity is a. The Erlang capacities for $\epsilon = 0.01$ are given for a few traffic intensities in the following table:

a	N
10	18
100	117
1000	1029

In relative terms, the potential saving is of course bigger for smaller systems. For a system with $a = 1000$ the attainable saving of less than 3 % can hardly justify the complexities of a dynamic bandwidth management. (Another issue is that some degree of flexibility in the bandwidth allocation is needed in order to cope with the uncertainty related to the estimation of a. However, the associated time scale is much longer than the time scale of some multiple of interarrival times we are concerned here).

As mentioned, the above values indicate an upper bound to the saving. In practice, the capacity saving is smaller because

- The resource pool is finite and the statistical multiplexing between different VP's is not perfect.
- The periodic VP allocation scheme is less efficient than call-by-call handling.

2.1 Statistical multiplexing between different VP's

As to the statistical multiplexing between the VP's note that capacity released by one VP may not be immediately utilizable by another VP. Conversely, if temporarily deallocated capacity is taken by another VP, the capacity may not be available for the original VP when needed, leading to a higher call blocking unless specific allowance is made for it. For instance, if the offered traffic in a VP is $a = 10$ and we wish to carry 10 such VP's, then a capacity of $n = 117$ is needed, at the minimum, and the saving per VP is at most $18 - 11.7$ and not $18 - 10$.

2.2 Efficiency of the periodic bandwidth allocation

The main point we wish to study is the efficiency of the proposed VP management scheme vs. those of static allocation and call-by-call allocation. In order to separate this from the effect of mutual statistical multiplexing of different VP's we assume now that the resource pool is infinite and gauge the efficiency of a method by the mean value of the reserved bandwidth.

In a careful implementation of the method by Mocci et al., the capacity allocated for a VP at the update instant is such as to make the expected blocking probability in the next interval precisely equal to ϵ. The required capacity depends on many parameters: desired overall blocking probability ϵ, offered traffic intensity a, current occupancy n and the update interval Δ. The method presented in [1] allows one to determine the required capacity. However, the calculations are rather involved and real-time computation is out of question. Precalculated tables or approximations are needed, but even then the number of parameters makes the task rather complex.

Moreover, one of the parameters, a, is not known exactly. A separate estimation algorithm is needed. Full reliance on an estimated value, however, may make the whole method sensitive to the estimation errors. For the sake of robustness, it would be preferrable to have a method which does not at all depend on a. Here we propose one simple allocation function:

$$N(n) = n + k(\epsilon, \Delta, n)n^{1/2}. \qquad (1)$$

The rationale behind this function is that the for a given a the occupancy distribution in an infinite system is Poisson(a) which has mean a and standard deviation $\sqrt{a}$ and most of the time $n \approx a$. (In the same spirit, Mocci et al. use n as an estimate for a.) The safety factor k depends obviously on ϵ and Δ. We let it additionally depend on n in order to achieve a nearly constant blocking probability of ϵ over a wide range of values of a. (An interesting question is whether there is a unique function $N_{\epsilon,\Delta}(n)$ which for a given Δ makes the blocking probability strictly constant (ϵ) for all values of a).

Using the method of [1] we can now calculate the average blocking probability $b(n, a)$ for offered traffic intensity a in an interval Δ given that the system occupancy was n in the beginning of the interval and that the bandwidth reservation is made according to (1).

Assuming that blocking is anyway small we can take the distribution of the initial state n to be Poisson(a), i.e.,

$$p_n(a) = \frac{a^n}{n!} e^{-a}, \tag{2}$$

and get the overall blocking probability

$$B(a) = \sum_{i=0}^{\infty} p_n(a) b(n, a). \tag{3}$$

Similarly we get the mean reserved capacity

$$\mathrm{E}\left[N\right] = \sum_{i=0}^{\infty} p_n(a) N(n). \tag{4}$$

In figure 1 we study a system with offered traffic $a = 10$. The figure shows how the mean reserved capacity depends on the updating interval (mean holding time of a call is used as the unit of time) when coefficient k in (1) was chosen so that (3) yields a blocking probability of 0.01. The bottom of the figure represents a, i.e., the mean capacity in the call-by-call allocation and the top corresponds to the static allocation by Erlang's formula. We see that the studied scheme is clearly less efficient than the call-by-call allocation. However some saving can be obtained with respect to the static allocation. Unfortunately, this saving diminishes as the updating interval becomes longer. Note that when $a\Delta = 1$ there is no gain in the VP management work. Similar results for a system with $a = 100$ are shown in figure 2. In this case, an update interval of 0.1 might be a reasonable choice, giving about 10% saving in the mean capacity in comparison with static allocation and reducing the call processing load by a factor of 10 in comparison with the call-by-call allocation.

In the previous examples we still let k depend on a. For instance with $\Delta = 0.1$ we had to choose $k = 0.971$ for $a = 10$ and $k = 0.748$ for $a = 100$ in order to obtain $\epsilon = 0.01$. Replacing once again a by n and assuming the dependence of k on n be of the power law form we arrive at an approximation $k(0.01, 0.1, n) \approx 1.26n^{-0.11}$. Substitution into (1) gives the allocation formula ($\epsilon = 0.01$, $\Delta = 0.1$)

$$N(n) = n + 1.26\, n^{0.39}. \tag{5}$$

We have checked by simulations that this allocation function (to be precise, the least integer larger or equal to $N(n)$ was used) indeed leads to an almost constant blocking probability over a wide range of traffic intensities. This is demonstrated in the following table.

a	$B(a)$	E[N]
3	1.4%	5.2
10	0.8%	13.4
30	0.8%	34.5
100	1.0%	107.5
300	1.1%	308.1

References

[1] J. Virtamo and S. Aalto, Blocking probabilities in a transient system, COST257TD(96).

[2] C. Bruni, P. D'Andrea, U. Mocci and C. Scoglio, Optimal capacity assignment of virtual paths in ATM networks, *Globecom'94*, San Francisco, (1994) pp. 207-11.

[3] C. Bruni, U. Mocci, P. Pannunzi and C. Scoglio, Efficient capacity assignment for ATM virtual paths, in Hot Topics on traffic and Performance from RACE to ACTS, Milano, June 14th-15th (1995), paper 14.

[4] U. Mocci, P. Perfetti and C. Scoglio, VP capacity management in ATM networks for short and long term traffic variations, COST 242 TD(95)59.

[5] U. Mocci, , P. Pannunzi and C. Scoglio, Adaptive capacity management of virtual path networks, to be presented at *Globecom'96*, London (1996).

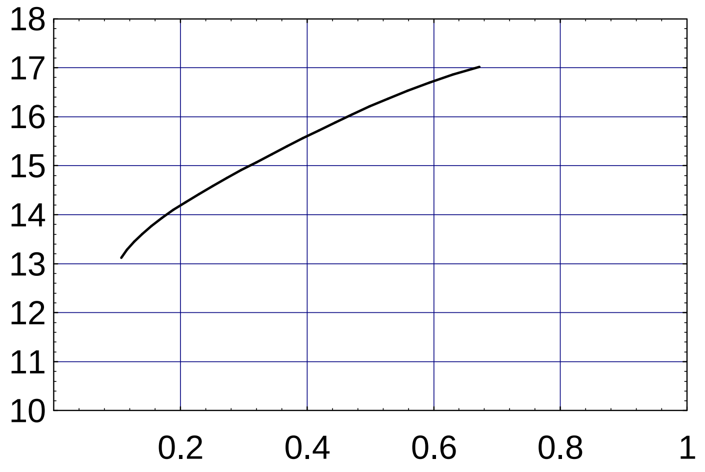

Figure 1: Average resource allocation (4) as a function of the update interval for a system with $a = 10$.

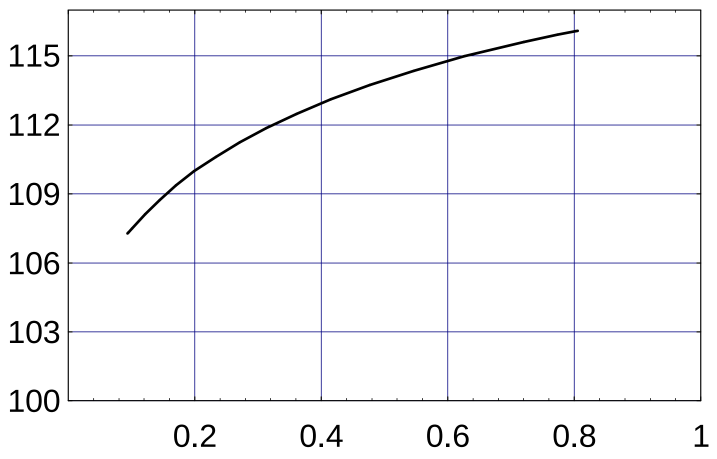

Figure 2: Average resource allocation (4) as a function of the update interval for a system with $a = 100$.