Computer Communications 22 (1999) 1127–1135

Bandwidth-demand prediction in virtual path in ATM networks using genetic algorithms

N. Swaminathan[a], J. Srinivasan[b], S.V. Raghavan[a,*]

[a]*Department of Computer Science and Engineering, Indian Institute of Technology, Madras, India*
[b]*Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India*

Received 1 December 1998; received in revised form 8 April 1999; accepted 8 April 1999

Abstract

The concept of Virtual Paths (VP) is a powerful technique to improve the transmission efficiency in ATM networks. Transmission efficiency can be improved by dynamically changing the bandwidths of the VPs, based on the demand. Intelligent controllers, which *predict bandwidth-demand patterns* to enable better VP management, have the potential to revolutionize ATM network performance. We present a scheme based on the *Evolutionary Genetic Approach* to predict the bandwidth-demand patterns in VPs. The efficiency of this approach, quantified in terms of the *Degree of Learning* (DoL), is evaluated through simulation and the results are presented. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: ATM networks; VP bandwidth management; Genetic algorithms; Bandwidth-demand patterns

1. Introduction

The Virtual Path (VP) concept is considered an effective means to manage an ATM network [1,2]. The benefits of using VPs are many: reduced node processing per virtual circuit, reduced call setup time, simplified node structure, and proper control over routing and bandwidth management. VP bandwidth control improves transmission efficiency for a given call blocking probability (and vice versa) [1]. Improvement in efficiency is achieved through the multiplexing of the physical link bandwidth among the VPs that share the same physical link. The VP bandwidth is dynamically altered in accordance to the traffic demand. There are many schemes suggested in the literature for VP bandwidth control, some centralized [3–10] and others distributed [11–13]. For efficient management of the VP bandwidth, an accurate estimate of the bandwidth-demand of the traffic flowing through the VPs is required. In order to make an accurate estimate of the bandwidth-demand, a proper understanding of the behavior of the traffic is required.

The behavior of the traffic is usually expressed in terms of its past statistical properties. For short term (less than one hour) estimate of traffic behavior the statistical properties are assumed to vary slowly and hence daily, weekly and seasonal cycles are not considered, instead the peak demand during the previous observation period is considered as the current demand [13]. Characterizing the traffic by on-line measurement of the large deviation rate function (entropy) is studied in [14,15]. In this paper we characterize the traffic using bandwidth-demand patterns within a VP. The bandwidth-demand patterns have to be learned to predict the future behavior of the traffic.

Learning algorithms have been suggested for adaptive routing in ISDN [16] and multicast routing [17] based on the network load. We propose an evolutionary-genetic approach to learn the dynamic behavior of traffic on the basis of bandwidth-demand patterns in VPs. We also describe as to how to use the bandwidth-demand patterns, in order to make short term bandwidth-demand predictions in VPs that is subsequently used for VP bandwidth management. From our studies, it is observed that the genetic approach naturally captures the short-term trend and variation in bandwidth-demand for both Poisson and Self-Similar call arrivals.

The remaining part of the paper is divided as follows. Section 2 describes the proposed approach by developing the framework for using genetic algorithms in such class of problems. Simulation studies and the results are discussed in Section 3. Section 4 concludes the paper by exploring the scope for further research.

* Corresponding author. Tel.: + 91-44-2350563; fax: + 91-44-2350509.

E-mail addresses: swami@cs.iitm.ernet.in (N. Swaminathan), jsrini@iitk.ac.in (J. Srinivasan), svr@cs.iitm.ernet.in (S.V. Raghavan)

2. The evolutionary-genetic algorithm

A genetic algorithm is a heuristic approach that applies the natural genetic ideas of natural selection, mutation and survival of the fittest. A rigorous mathematical formalism was introduced by Holland and his collaborators, and is till date the basis for genetic algorithms. A comprehensive review of this field can be found in the book by Goldberg [18].

The algorithm uses a set of offered solutions called a "population". Each solution called an "individual", can be any solution in the solution space represented by a string called "chromosome". The solution space is explored in order to find an optimal solution that satisfies the constraints posed by the problem. A solution is judged as optimal based on a criterion called the "fitness value". The current population is evolved creating a new population with higher fitness. The evolution can be done using the following operators:

1. *Crossover:* Two individuals are mated together in order to exchange genetic information. Crossover may take place at one, two or more points.
2. *Mutation:* Mutation is a random change in the chromosome. This enables a search in new avenues of the solution space.

The next generation of the population is evolved from the current population using the above operators. The fitness of each individual of the new population is evaluated. If an optimal individual is found then the process is terminated, else the next generation of individuals is evolved. The same process continues until a solution is found.

2.1. Applying the genetic algorithm to bandwidth-demand prediction

Coding: In nature, an allele[1] is encoded in an alphabet composed of four symbols. In the case of ATM networks, bandwidth-demands within a VP can be mapped to discrete ranges. These ranges in turn can be mapped to symbols of an alphabet. Thus the characteristics of the VP bandwidth-demand pattern can be encoded as a string composed of the symbols of the alphabet. The bandwidth-demand sampled after regular time intervals can be mapped to a string of symbols that characterizes the bandwidth-demand pattern for the sum of those intervals of time. We call such a string a *loadgene*. The bandwidth-demand pattern prediction problem then boils down to predicting the next loadgene given the previous few loadgenes. The methodology adopted is to generate a population of sample loadgenes, each obeying the distribution of symbols in the previous loadgene. The next loadgene is then predicted by evolving this population using a suitable fitness criterion.

Fitness criterion: The fitness of a living organism corresponds to its ability to survive the stress factors in its environment. In natural selection, the best or the fittest organism is selected to live and propagate further. We seek to model such a scenario in our bandwidth-demand pattern prediction problem. The aim of our problem is to *learn* from the relationship among previous loadgenes and predict the future loadgene. The range of difference in information content between k successive loadgenes is observed and set as a parameter for prediction. The difference in information content between the predicted loadgene and the current loadgene should lie within this observed range. The difference in information content of a possible pattern and the current loadgene when compared with the limits of the observed range (of differences over k previous loadgenes) provides us with a measure of fitness for the candidate loadgene. Candidates with high fitness values are preferentially chosen to propagate and reproduce further. Thus natural selection is incorporated.

Information content in loadgene: The important questions to be answered in our model are:

What is the metric for the evaluation of loadgenes? The answer is guided by the ultimate aim of the approach, i.e. managing VP bandwidth effectively. Effective management is achieved if we are able to predict drastic changes in bandwidth-demand beforehand. Given two loadgenes, if the number of drastic changes in demand in both the loadgenes are close to each other, then the two loadgenes are similar to each other. Thus, the number of such drastic changes in a loadgene is a metric for comparison.

What is the concept and cognizance of a drastic change? We formalize the concept of a "drastic change in demand" by defining a dynamically changing threshold value for the difference between two successive samplings of bandwidth-demands. If the difference is greater than the current value of the threshold, then the change in demand is deemed serious enough to warrant a contribution to the information content of the loadgene. The threshold value is dynamically changed by observing the minimum and maximum differences in successive symbols present in the previously arrived loadgene. The threshold is then set as the product of the sum of the minimum and maximum differences with a factor α.

2.2. The fitness criterion and DoL

The information content in a string can be captured in the form of a real valued number in the range 0–1.0. This number is called the *r-value* of the string. The first step in computing the r-value is to decide what is the information content within the string that we want to quantify. The information content are the patterns of interest which appear as substrings within a string. In an ATM context, the patterns of interest are the bandwidth-demand patterns. These patterns are coded as substrings of length 2, such that the difference in the values of the symbols contained in the substring is greater than the threshold value described

[1] An allele is a string representing a characteristic of an organism.

```
if ($r_d(S, S_1) < m$) then
    $F = 1.0 - (m - r_d(S, S_1))$
else if ($r_d(S, S_1) > M$) then
    $F = 1.0 - (r_d(S, S_1) - M)$
else
    $F = 1.0$
```

Fig. 1. Fitness criteria.

in Section 2.1. The steps involved in the computation of the r-value of a string are as follows:

1. Count all occurrences of all the substrings of length 2 in the given string S. Store the count in a matrix. For instance, occurrence of "*ab*" in the string S will result in entry [a,b] of the matrix getting incremented.
2. Treat the matrix as a one-dimensional vector and normalize it.
3. Compute the square root of the sum of squares of the entries of the matrix. This is taken as the r-value.

The absolute difference in the r-values of two loadgenes: S_1 and S_2 is denoted by $r_d(S_1,S_2)$. If $S_k,S_{k-1},\ldots,S_1$ represent the k previous loadgenes that have been sampled, and m and M represent the minimum and maximum respectively, of $r_d(S_k,S_k-1)$, $r_d(S_k-1,S_{k-2}),\ldots,r_d(S_2,S_1)$ then the fitness value, F, of candidate loadgene S can be formally defined as in Fig. 1.

Clearly, a candidate with higher fitness is one which lies close to the permissible range. The entire algorithm hinges on the fact that the next loadgene will share a relationship with the current loadgene, which is very similar to the relationship that existed among the previous k loadgenes. We introduce the term *Degree of Learning* (DoL) which represents the similarity between the r-values of the actual loadgene and the predicted loadgene. The DoL of the algorithm is defined as follows:

$$\mathrm{DoL} = 1.0 - r_d(S_{\mathrm{predicted}}, S_{\mathrm{actual}}). \quad (1)$$

Thus, if the r_d value of the predicted loadgene is small, then the DoL is near 1.0. A high value of DoL signifies that the predicted loadgene has information content very similar to that of the actual loadgene that arrives. In the current context, a high DoL indicates that the number of high fluctuations in the bandwidth-demand is similar for the actual and the predicted loadgenes. If the genetic algorithm has a high DoL then the bandwidth-demand pattern estimates for the next interval will be accurate, thus making bandwidth management schemes more efficient. It must be noted here that it is impractical to assume that one can predict deterministically the exact instance when there will be a high fluctuation in demand. The best one can do is to predict the statistical behavior in a given time interval. We illustrate the proposed scheme using simulations.

3. Simulation experiments

In this section we verify the effectiveness of the proposed bandwidth-demand predictor through simulation. First we describe the simulation environment and then explain the various experiments conducted and the inferences derived.

3.1. Description of the simulator

A call level ATM simulator was developed in C language for the purpose of simulating the call level traffic in a typical ATM network. Two physical network topologies were considered for the simulation. One network contains four nodes as shown in Fig. 2 (left) and the other contains 11 nodes as shown in Fig. 2 (right). In this simulation study we

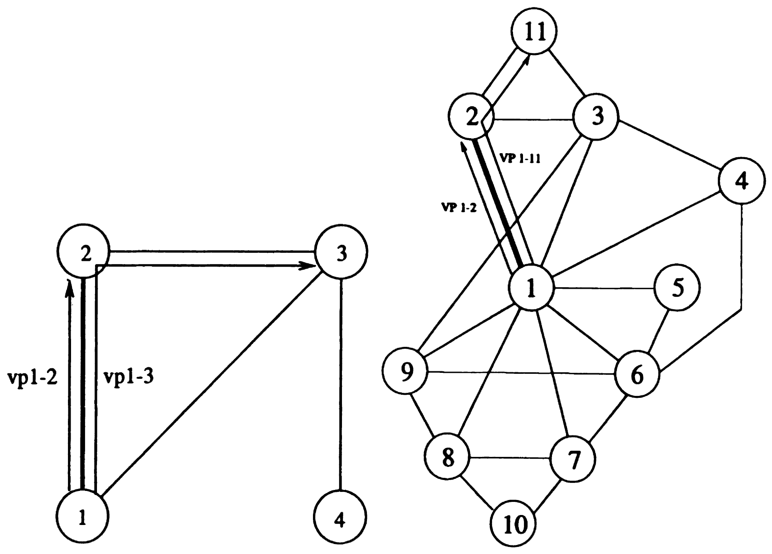

Fig. 2. The networks used in the simulation.

Table 1
Call characteristics used in the simulation

Call type	Bandwidth-demand per call in 64 kbps units	Mean call duration	Call blocking prob. acceptable
A	1	3 min	0.03
B	2	30 min	0.03
C	4	15 s	0.03

consider both *Poisson* and *Self-Similar* call arrivals. For generating Poisson traffic, the call inter-arrival time and call holding times were drawn from an exponential distribution. For generating self-similar traffic, the call inter-arrival times and call holding times were drawn from a Pareto distribution. The value of the shape parameter β is chosen as 0.9, which is the typical value observed for various applications such as web, FTP data, NNTP, SMTP, and telnet as reported in [19,20]. For further studies on burstiness of self-similar traffic one is referred to [21,22]. The calls are assumed to be of three types based on their bandwidth-demand and duration of calls, the details of which are given in Table 1. The proportion of calls of type A, B and C generated are 90%, 9%, and 1%, respectively. The traffic from any node to any other node in the network is uniformly distributed. The traffic in the VPs named vp1-2, vp1-3 of the 4-node network and vp1-2, vp1-11 of the 11-node network were monitored. The physical links which these VPs share are shown in thick line in Fig. 2. The bandwidth-demand in these VPs was sampled at intervals of one second and one day's worth of data was simulated and used for the analysis. The nodes are logically connected through a full mesh VP network. VP network was considered fully meshed since this is the worst case by which the physical links can be shared by the VPs. All the VPs are assumed to be unidirectional.

```
loadgene EGA(loadgene Z, double m, double M)
    {
      /* Generate a population of loadgenes which have same */
      /* distribution of symbols as Z */
          P = Generate_Population(Z);

      /* Calculate the fitness of the members of the population */
          Calculate_Fitness(P , Z, m, M);

      /* Check for solution in population */
          repeat
              For each loadgene, L, in the population do
                  if ( Fitness(L) == 1.0) return L;

      /* Since no solution has been found, evolve the next */
      /* generation */
              Next_Generation(P);

      /* Calculate fitness of the members of the next generation */
              Calculate_Fitness(P , Z, m, M);
          until a maximum of 100 generations
    }
```

Where :

N	:	Length of loadgene
X	:	String representing the bandwidth-demand sampling from time t to $t + N$
Y	:	String representing bandwidth-demand sampling from time $t + N$ to time $t + 2N$
Z	:	String representing bandwidth-demand sampling from time $t + 2N$ to time $t + 3N$
P	:	Predicted bandwidth-demand sampling from time $t + 3N$ to $t + 4N$
m	:	$\min(r_d(X,Y), r_d(Y,Z))$
M	:	$\max(r_d(X,Y), r_d(Y,Z))$
P	:	EGA(Z,m,M)

Fig. 3. The EGA algorithm.

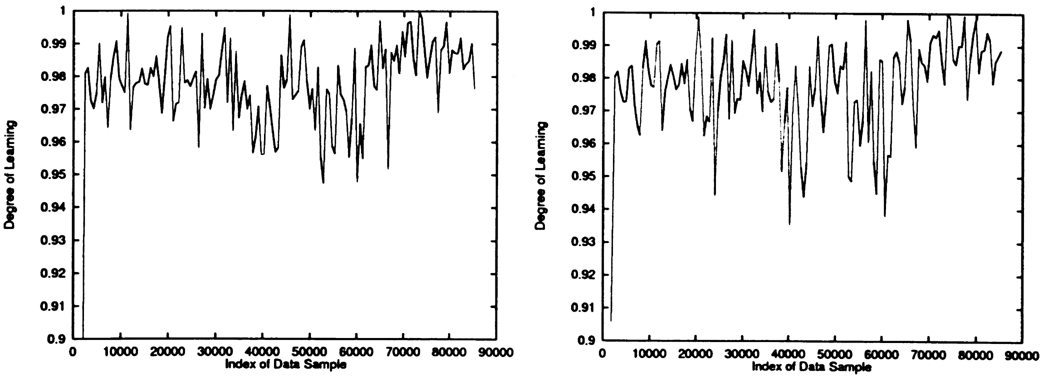

Fig. 4. Degree of learning of EGA with loadgene length = 600 and $\alpha = 0.5$ (left) first method, (right) second method.

The bandwidth-demand is quantized in terms of multiples of 64 kbps. Every symbol of alphabet represents a multiple of 64 kbps. We consider T1 (1.5 Mbps) type of physical links for the 4-node network, and hence the cardinality of the alphabet set for the 4-node network is 24. For the 11-node network we consider four numbers of T1 links per physical link between any two nodes, so the net bandwidth available between the two nodes is 6 Mbps. Hence the cardinality of the alphabet set for the 11-node network is 92. We assume higher bandwidth for the 11-node network since in a fully mesh connected VP network there will be 11 VPs sharing a physical link and if there was only one T1 link per physical link, then each VP will hardly get three quantized bandwidth units per VP.

The VP bandwidth management is part of every ATM switch and hence it is a distributed scheme. This scheme exploits the advantages of a distributed system such as fault tolerance and fault isolation (local problems do not affect the global network). Further, this implementation could co-exist with heterogenous switches and is application independent.

3.2. Experiments

Performance of EGA: The Evolutionary-Genetic Approach (EGA) was used to predict the bandwidth-demands based on the data generated by the simulator. The algorithm used for the EGA is given in Fig. 3. The data from the simulator was treated as a set of loadgenes, arriving over a period of time. The EGA predicted the next loadgene on the basis of the previous three loadgenes. The predicted loadgene was compared with the actual loadgene from the simulator and the DoL was plotted. The results are shown in Fig. 4 (left) for a loadgene length of 600. The actual bandwidth-demands and the predicted demands are plotted over a small time interval in Fig. 5. We set the value of $k = 3$ which means that the limits within which $r_d(S_{\text{predicted}}, S_{\text{current}})$ should lie, is determined on the basis of the current loadgene and the previous two loadgenes. This in effect means that half-an-hour's worth of data influences the prediction for the next 10 min. As is seen from the plot even the worst DoL is above 0.8, i.e. $r_d(S_{\text{predicted}},S_{\text{actual}})$ is less than 0.2.

A scheme for actual implementation: The EGA takes a finite amount of time to evolve the next loadgene. This presents a problem because the idea behind the whole exercise is to predict the next loadgene before it arrives. We illustrate the problem and a possible solution to it with an example. Consider the successive loadgenes X, Y and Z where Z is the loadgene, which represents the bandwidth samplings in the last n minutes. The EGA has to predict

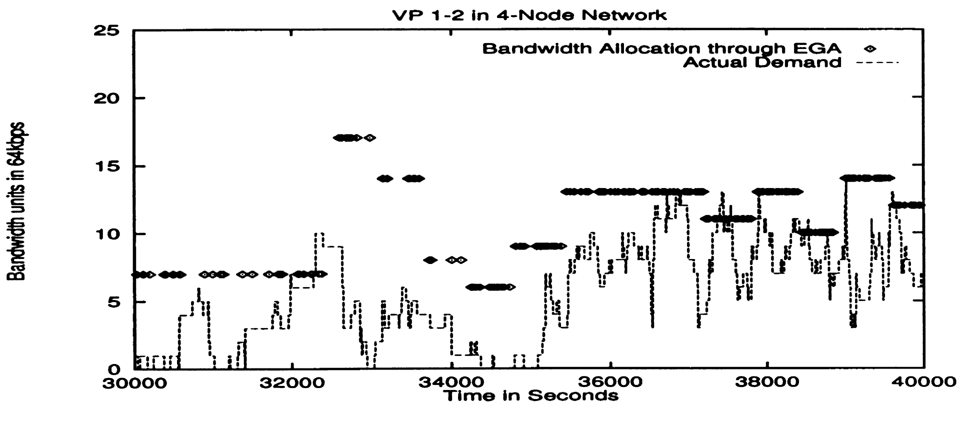

Fig. 5. Actual bandwidth-demand and predicted bandwidth-demand.

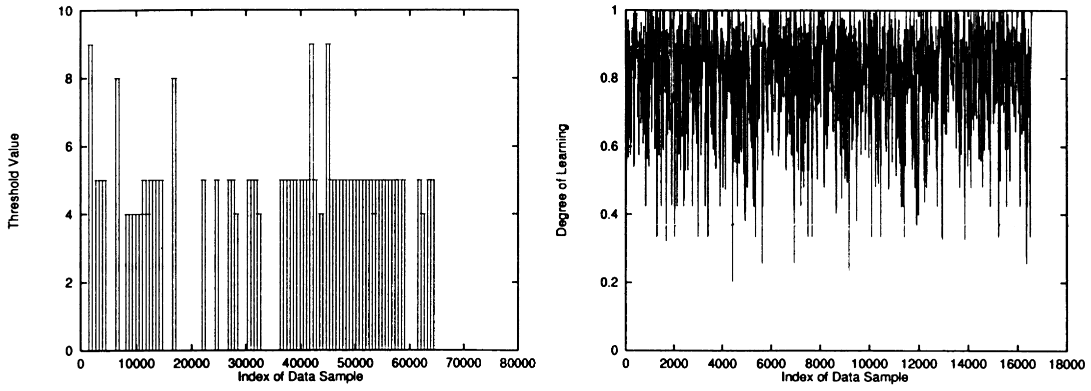

Fig. 6. Threshold value versus data index (left) DoL with loadgene (right) size = 10.

the loadgene L that represents the bandwidth samplings of the next n minutes. Clearly if the EGA takes as parameters X, Y and Z, some portion (may be all) of L will already have been sampled before the EGA is able to make a prediction. To overcome this problem, we passed as parameters to the EGA the loadgenes X, Y and $Z_{\text{predicted}}$. This execution of the EGA was called even before the sampling of Z_{actual}. The results using such a scheme (called the second method) are presented in Fig. 4 (right) which clearly indicates that there is no difference in DoL compared to Fig. 4 (left) called the first method.

Thus, if we consider Z to be starting at time t_0, the EGA uses the bandwidth samplings of the duration $(t_0 - 2n)$ to t_0 and the predicted bandwidth samplings for the duration t_0 to $t_0 + n$ to predict the bandwidth samplings for the period $t_0 + n$ to $t_0 + 2n$. The EGA is called at time t_0 and if one carefully chooses the length of the loadgenes, i.e. the duration of sampling for one loadgene, one can ensure that the EGA predicts well before the actual event occurs at time $t_0 + n$.

The threshold value and α: The DoL of the EGA depends on the threshold value described in Section 2.1. To illustrate how the threshold value dynamically changes with respect to time we have plotted the threshold value against time in Fig. 6 (left). The threshold value is influenced by the factor α, the relationship of which is as follows:

$$T = \lfloor (\delta_{\min} + \delta_{\max})\alpha \rfloor \tag{2}$$

where T is the threshold value, $\delta_{\min}$ the minimum difference between successive bandwidth-demands $\delta_{\max}$ the maximum difference between successive bandwidth-demands.

Loadgene length: The length of the loadgene is an important parameter in implementing an efficient VP bandwidth manager. The selection of an optimum loadgene length is influenced by the following factors: the DoL and the time taken to predict. It has been found that DoL is erratic for smaller values of loadgene length as shown in Fig. 6 (right) which leads to instability in network operation. For larger values of the loadgene length the time taken to predict is large which would make the VP manager less sensitive to rapid changes in demand.

In order to select an optimal value for loadgene length and α we plotted the statistical characteristics of DoL when using different values of loadgene length and α as shown in Fig. 7 (right). As is seen from Fig. 7 (right) the DoL of the EGA is very consistent except for very small values of the loadgene length. This lends credence to our contention that the EGA predicts the statistical behavior of the bandwidth-demand with marginal standard deviation. However for

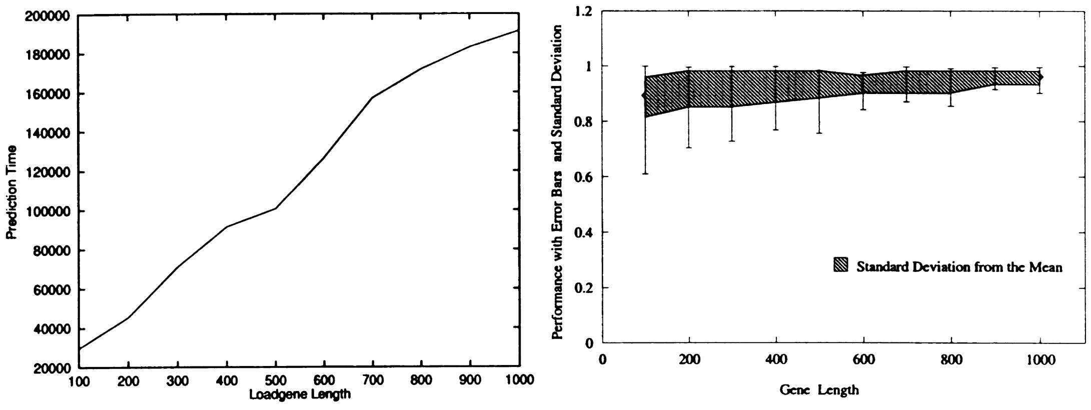

Fig. 7. Prediction time versus loadgene size (left) DoL as performance versus loadgene length (right) for $\alpha = 0.5$.

```
if (Measured_Previous_Peak > 1.2 * Previous_Predicted_Peak) then
    Previous_Peak = 1.2*Measured_Previous_Peak;
else
    Previous_Peak = Previous_Predicted_Peak;

if (Sum_of_Negative_Changes > Sum_of_Positive_Changes) then
    Allocated_Bandwidth = Previous_Peak - Mean + 0.5*Stddev;
else if (Sum_of_Negative_Changes < Sum_of_Positive_Changes) then
    Allocated_Bandwidth = Previous_Peak + Mean + 1.2*Stddev;
else
    Allocated_Bandwidth = Previous_Allocated_Bandwidth;
```

Fig. 8. Bandwidth allocation estimation algorithm.

implementation purposes we need to optimize the loadgene length based on the time taken for prediction. The time taken for prediction is directly proportional to the number of times the random function is called in our EGA simulator. Fig. 7 (left) illustrates the relationship between the time taken for prediction and the loadgene length. On the basis of our simulations, we suggest that the optimal loadgene length for the current context is 600. However, this figure is implementation specific and should be tuned after installing the EGA.

Bandwidth allocation estimation: After generating the predicted loadgene, the next step is to estimate the bandwidth to be allocated for the VP. The bandwidth to be allocated is estimated from the statistical properties of the predicted loadgene. The mean and standard deviation of the negative and positive changes in the predicted loadgene is computed. The peak demand during the previous measurement period is also recorded. The algorithm to compute the allocated bandwidth is given in Fig. 8. The allocated bandwidth with respect to time for Poisson arrival of calls is shown in Figs. 9 and 10. For a self-similar call arrival, the bandwidth allocation with respect to time is shown in Figs. 11 and 12. The allocation smoothly varies with respect to the traffic needs for a 4-node network for both Poisson and self-similar traffic. For the 11-node network the allocation is not able to follow the demand as in the case of the 4-node network because, more VPs share the physical links in the 11-node network. Hence in the competition for getting the allocated bandwidth, the chance of getting the requested allocation by the VPs is lesser for a 11-node network than the 4-node network.

In order to study the performance of adaptive allocation through the EGA method we compare its performance with the fixed allocation method. Call blocking probability is used as the metric for comparing the performance of the fixed and the adaptive allocation methods. In the 4-node network there are four VPs that share a physical link (maximum capacity of 24 bandwidth units), hence the bandwidth allocated per VP in the case of fixed allocation is assumed to be six units each. In the 11-node network there are 11 VPs that share a physical link (maximum capacity of 92 bandwidth units), hence the bandwidth allocated per VP in the case of fixed allocation is assumed to be eight units each. The performance of the two methods is shown in Table 2,

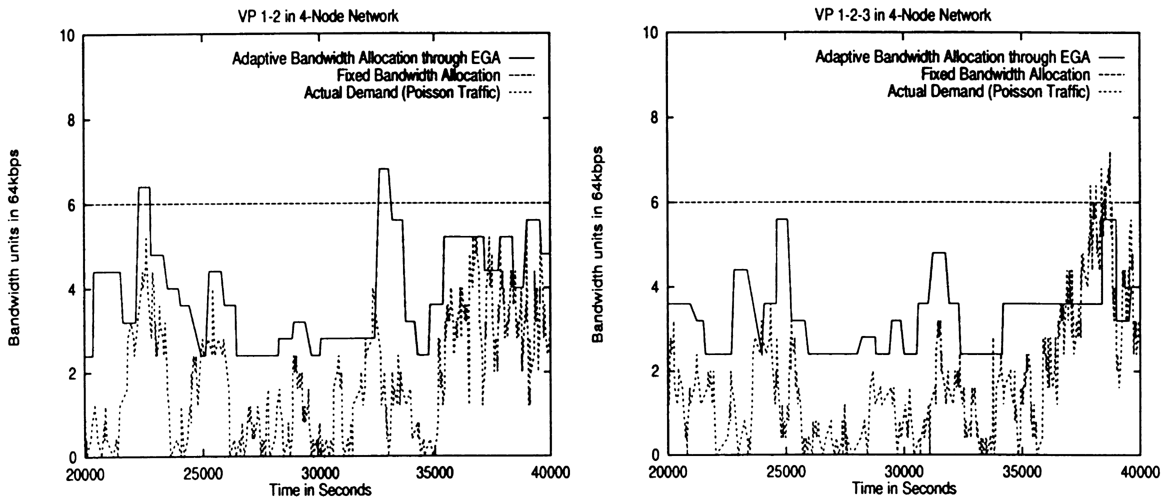

Fig. 9. Bandwidth allocation for a 4-node network through EGA for Poisson call arrival.

Table 2
Call blocking probability for fixed and adaptive bandwidth allocation methods

Traffic type	Poisson		Self-similar	
Network (traffic)	4-Node (20 Erlangs)	11-Node (250 Erlangs)	4-Node (20 Erlangs)	11-Node (250 Erlangs)
Adapt. alloc.	0.04	0.032	0.07	0.072
Fixed. alloc.	0.09	0.091	0.09	0.091

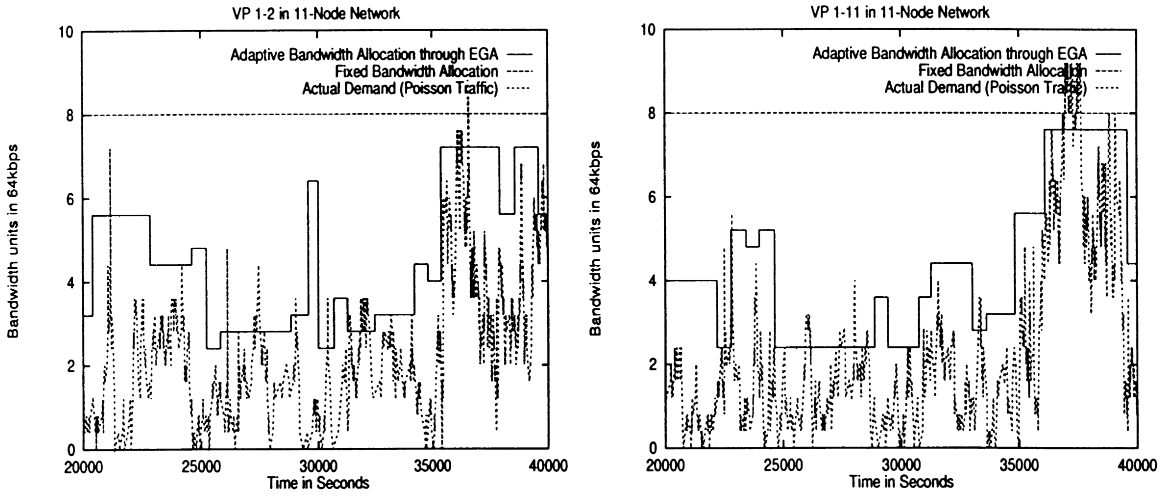

Fig. 10. Bandwidth allocation for a 11-node network through EGA for Poisson call arrival.

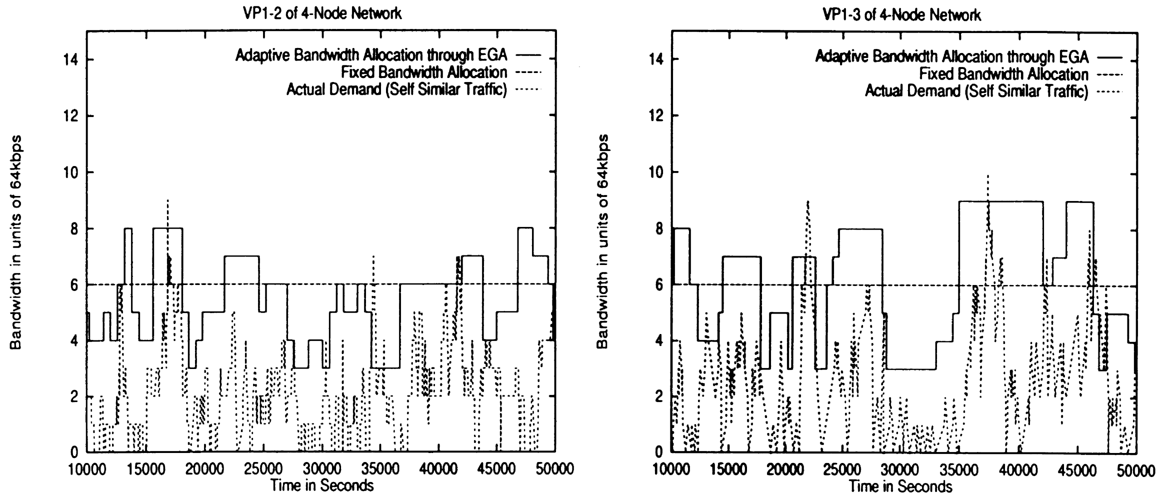

Fig. 11. Bandwidth allocation for a 4-node network through EGA for self-similar call arrival.

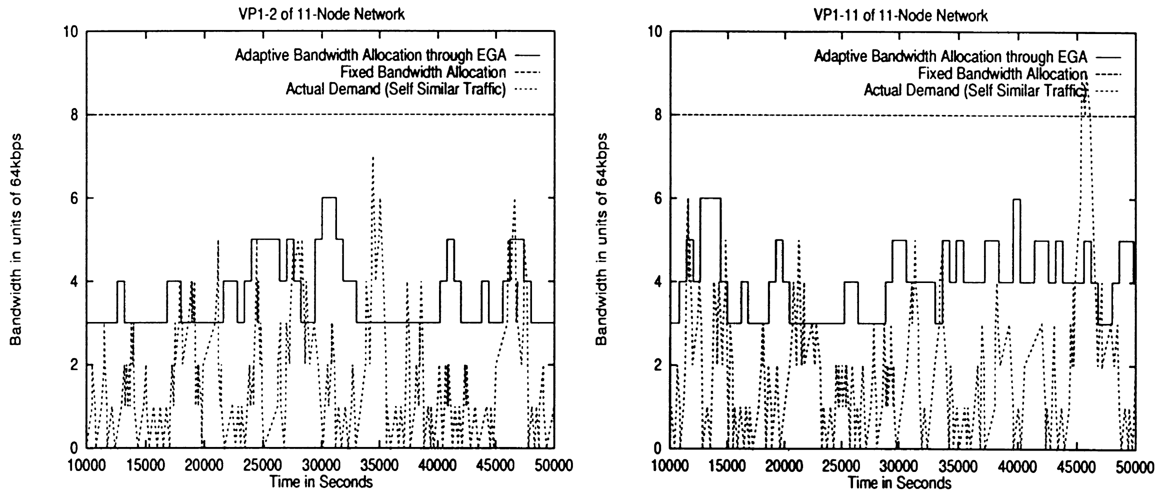

Fig. 12. Bandwidth allocation for a 11-node network through EGA for self-similar call arrival.

where the adaptive allocation method consistently exhibits a lower call blocking probability than the fixed allocation method. When comparing the performance under different traffic types in Table 2, it is found that the adaptive allocation shows at least 50% lesser call blocking probability than the fixed allocation method for Poisson call arrivals. In the case of self-similar call arrivals, the adaptive allocation though is showing higher call blocking probability than the Poisson arrivals, yet it is 20% lesser than fixed allocation method. Hence, the adaptive VP bandwidth allocation using the EGA method is found to utilize the physical bandwidth efficiently by adapting to the traffic situations for small and large networks with both Poisson and self-similar call arrivals.

4. Conclusion

Virtual Path (VP) bandwidth control improves transmission efficiency in an ATM network. An accurate estimate of the bandwidth-demand within a VP leads to efficient VP bandwidth control. So far the statistical methods were employed to predict the bandwidth-demand. In this paper we have presented an Evolutionary-Genetic Approach (EGA) to predict bandwidth-demand patterns within a VP. We have quantified the efficiency of this scheme in terms of the Degree of Learning. Factors affecting the implementation of the EGA were discussed and a methodology for identifying the optimal parameters was illustrated. Simulation studies on the effectiveness of the EGA on an ATM network and their results were presented. From our studies it was observed that the EGA captures the trend and variation in bandwidth-demand for both Poisson and self-similar call arrivals. The adaptive allocation through the EGA shows 50% lesser call blocking probability than the fixed allocation method for Poisson arrivals and 20% lesser for self-similar arrivals. It would be interesting to implement the EGA in actual hardware and test it in a real environment.

References

[1] S. Ohta, K. Sato, Dynamic bandwidth control of the virtual path in an asynchronous transfer mode network, IEEE Trans. Commun. 40 (7) (1992) 1239–1247.

[2] J. Burgin, D. Dorman, Broadband ISDN resource management: the role of virtual paths, IEEE Commun. Mag. September 29 (9) (1991) 44–48.

[3] R. Siebenhaar, Optimized ATM virtual path bandwidth management under fairness constraints, Proc. IEEE Globecom (1994) 321–329.

[4] K.T. Cheng, F.Y.S. Lin, On the joint virtual path assignment and virtual circuit routing problem in ATM networks, Proc. IEEE Globecom (1994) 777–782.

[5] G. Gordos, A. Farago, S. Blaabjerg, T. Henk, A new degree of freedom in ATM network dimensioning: optimizing the logical configuration, IEEE J. Selected Areas Comm. 13 (7) (1995) 1199–1206.

[6] M. Logothetis, S. Shioda, Medium-term centralized virtual-path bandwidth control based on traffic measurements, IEEE Trans. Comm. 43 (10) (1995) 2630–2640.

[7] Ake Arvidsson, High level B-ISDN/ATM traffic management in real time, Proc. of Second IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, Bradford University, vol. 1, 4–7 July 1994, pp. 177–207.

[8] Saito Hiroshi, Dynamic resource allocation in ATM networks, IEEE Commun. Mag. 35 (5) (1997) 146–153.

[9] M. Pioro, P. Gajowniczek, Stochastic allocation of virtual paths to ATM links, Proc. of Second IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, Bradford University, (vol. 1), July 4–7 1994, pp. 135–146.

[10] N. Anerousis, A.A. Lazar, Virtual path control for ATM networks with call level quality of service guarantees, IEEE Trans. Networking 6 (2) (1998) 222–236.

[11] S. Shioda, H. Uose, Virtual path bandwidth control method for ATM networks: successive modification method, IEICE Trans. J75-B-1 (5) (1992) 333–342.

[12] S. Srinivasan, M. Veeraraghavan, DIVA: a distributed and dynamic VP management algorithm, Proc. of the Fifth IFIP/IEEE International Symposium on Integrated Network Management, San Diego, California, USA, May 12–16 1997, pp. 287–298.

[13] Shioda Shigeo, Self-sizing network—algorithmic aspects, Intl J. Commun. Systems 11 (1998) 43–58.

[14] S. Crosby, I. Leslie, M. Huggard, J.T. Lewis, B. McGurk, R. Russell, Predicting bandwidth requirements of ATM and Ethernet traffic, Proc. 13th IEE UK Teletraffic Symposium (UKTS), February 29, 1996.

[15] S. Crosby, I. Leslie, J. Lewis, R. Russell, F. Toomey, B. McGurk, Practical connection admission control for ATM networks Based on Online Measurements, Proc. IEEE ATM'97, June, Lisbon, 1997.

[16] P. Aranzulla, J. Reeve, J. Mellor, P. Mars, Improved stochastic learning automata for routing in ISDNs, Symposium on Broadband Access Networks, at the European Conference on Networks and Optical Communications (NOC 97) Antwerp, Belgium, chap. 38, 1997, pp. 227–231.

[17] J.M. Reeve, P. Mars, T. Hodgkinson, Learning algorithms for multicast routing in communication networks, Proc. of the Tenth Yale Workshop on Adaptive and Learning Systems, Yale University, June 1998, pp. 70–75.

[18] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[19] V. Paxson, S. Floyd, Wide area traffic: the failure of Poisson modeling, IEEE Trans. Networking 3(3) (1995) 226–244.

[20] M.E. Crovella, A. Bestavros, Self-similarity in world wide web traffic: evidences and possible causes, IEEE/ACM Trans. Networking 5 (6) (1997) 835–846.

[21] F. Chen, J. Mellor, P. Mars, On burstiness of self-similar traffic models, Proc. of European Conference on Networks and Optical Communications (NOC'96), Heidelburg, vol. 2, 25–28 June, 1996, pp. 146–153.

[22] F. Chen, J. Mellor, P. Mars, Bursty measurement of Poisson and self-similar traffic, Proc. IEEE third Communication Networks Symposium, Manchester, 8–9 July 1996, pp. 168–171.