INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

NOTE TO USERS

The original manuscript received by UMI contains pages with slanted print. Pages were microfilmed as received.

This reproduction is the best copy available

UMI

Bandwidth Allocation in ATM Networks Using Neural Networks

By

Sameh Ahmed Youssef

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

1999

This manuscript has been read and accepted for the Garduate Faculty in Engineering in stisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

4/21/1999
Date

Chair of Examining Committee

4-23-1999
Date

Mumtaz K. Kamis
Excutive Officer

Ibrahim Habib

Yi Sun

Ken Young
Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

Bandwidth Allocation in ATM Networks Using Neural Networks

By
Sameh Ahmed Youssef

Advisor: Professor Tarek Saadawi

ATM network architecture are capable of supporting a wide range of connections with different quality of service requirements and traffic characteristics. This dynamic environment creates difficult traffic control problems when trying to achieve efficient use of network resources. One of such problem is bandwidth allocation. In this thesis, we suggest the neural network(NN) as a new approach to handle the bandwidth allocation problem in ATM. The proposed algorithms employ neural network to calculate the bandwidth required to support multimedia traffic with multiple quality of service (QoS) requirements. Two types of neural network have been used to design the bandwidth allocation controllers, backpropagation neural network and radial basis function neural network. The hierarchical structure has been used to simplify the design of the neural network controllers and to extend the ability of the neural network to learn a huge number of patterns. The NN controllers calculate the bandwidth required per call using on-line measurements of the traffic via its count process, instead of relying on simple parameters such as the peak, average bit rate and burst length. Furthermore, to enhance the statistical multiplexing gain, the

controllers calculate the gain obtained from multiplexing multiple streams of traffic supported on separate virtual paths (i.e., class multiplexing). In this thesis, we will explain the design of the controllers have been used and also we will compare the results from neural network controllers with the results from the traditional methods.

To My Wife, My Brothers and My Muslim Brothers at Islamic Center of Westchester

Contents

Lists of Figures

Chapter 1: Introduction

Asynchronous transfer mode (ATM) is ITU-T's transfer mode of choice for B-ISDN[1]. The B-ISDN is designed to provide multimedia traffic services (e.g., voice, data, image and video,..). These services are likely to have a wide range of both traffic characteristics and performance, or quality of services (QOS) requirements. The QOS defines a set of guaranteed performance measurement parameters (e.g., cell loss rate, delay, delay variability) that each type of the multimedia traffic requires from the network.

The ATM solution provides the flexibility to integrate the transport of services with a wide range of bandwidth requirements by assigning fixed-length cells to virtual connections on as needed basis. ATM also provides the potential to obtain improved bandwidth efficiency through the statistical multiplexing of variable bit-rate (VBR), or bursty traffic streams. In ATM networks, cells from different connections are multiplexed or switched using a common fabric, independent of the confections'bit-rates or burstness.

1.1 ATM Traffic Management

Traffic management deals with the problem of preventing the congestion in communication networks while achieving high utilization for it. In particular, when network resources are allocated to more connections or traffic than they can effectively support, network performance for users degrades (i.e., buffers start to overflow and delays increase beyond acceptable levels). Therefore, its necessary to manage resource utilization and control the traffic so that the network can operate at acceptable levels even at times when the offered load to the network exceeds its capacity. Traffic control in ATM is completely different than the traffic control in traditional networks for many reasons. ATM has to support variable bit rate (VBR) sources, which generate traffic at significantly different rates. A single source may generate multiple types of traffic (i.e., voice, data, and video) with different characteristics. ATM networks have to deal with cell delay variation and maximum delay. Traffic characteristics of various types of services are not well understood. The Call Admission control (CAC), procedure, which is, based upon the bandwidth allocation algorithms provides the main method of controlling the traffic performance of the network. In following section

we are discussing the different types of call admission control algorithms have been suggested for ATM traffic.

1.2 Call Admission Algorithms

In ATM, call admission control decides whether to accept or reject a call based upon availability of capacity required to support its QoS. Although this function is quite simple, tremendous difficulties have been reported in implementation [2]-[16] due to several reasons. 1) It is difficult to characterize and model the high variability of the multimedia traffic by simple parameters such as peak, average bit rate and burst duration since such set of parameters may not accurately capture the correlation properties of the traffic. 2) It is difficult to obtain accurate estimates of such parameters in real time. 3) Most existing approaches in the literature, for example see [2]-[8] and the many references therein, resort to approximations in order to obtain tractable queuing analysis which is used to calculate the bandwidth. Moreover, most of the existing approaches are not adaptive in the sense that the derived algorithms can not be adopted in order to deal with new traffic scenarios.

For example in [2]-[8], the bandwidth allocation problem has been solved using either queuing or simulation analysis for multiple types of traffic. Results from these solutions are stored in bandwidth allocation tables or sets of curves used to allocate the bandwidth. Although such methods obtain good results, they require excessive amount of storage and are not adaptive to the dynamic nature of the traffic. In [9]-[13] the concept of effective bandwidth was used to solve the bandwidth allocation problem. In [9], the bandwidth allocation problem was solved by calculating an "effective capacity" or bandwidth requirement for both individual and multiple calls. Two different approximations were used. The first one relies on the fluid-flow model, whereas the second focuses on the distribution of the stationary bit rate on a link. Similarly, in [10], the "effective capacity" was obtained for general Markovian sources. It was found to be the maximal real eigen value of a matrix, which is obtained from source characterization and the admission criterion. Both Fluid and Point process models were considered to characterize individual and superposition traffiç sources. In [11]-[13] large deviation theorem was applied in an attempt to obtain more accurate characterization for the traffic in order to estimate the equivalent capacity. In [14], Fuzzy logic hs been used to solve the CAC problem. The CAC has been solved based on possibilty distribution of cell loss ratio. The

possibility distribution is estimated in a fuzzy inference scheme by using observed data of cell loss ratio. Finally, in [15] and [16] a neural network model was used to solve the problem of call admission control and link capacity control. In that model, neural networks were trained to differentiate between two types of traffic: high bit rate and low bit rate patterns. Based on the bandwidth allocated to each type of traffic, neural networks decide whether to accept or reject the call. But as a result of using simple traffic patterns with low variance, the structure of model was very simple (10 inputs, 10 hidden neurons, and 1 output). Hence, it may not accurately capture the characteristics of the complex multimedia traffic.

1.3 Neural Networks

In this thesis we present a call admission control (CAC) algorithm for ATM networks using Neural Networks (NN). The goals of our design are to achieve an efficient algorithm (i.e., allows for statistical multiplexing gain), simple (in terms of processing and storage requirements), and robust (i.e., suitable for all types of traffic). The first goal was achieved by training NNs to learn the variability of the traffic bit-rate from its count process instead of relying on simple parameters. Furthermore, class multiplexing among different classes

of traffic (i.e., video, voice, etc) has been included in the design. The second goal has been achieved using a hierarchical NN architecture with separate NN units for each type of traffic. Finally, because our approach does not rely on specific traffic parameters, robustness was achieved. A major advantage of NNs approach is their ability to learn from experience. When properly trained, NNs can generalize and extrapolate results by mapping complex non-linear functions between its inputs and outputs. Moreover, NNs do not require an accurate mathematical modeling of the system, in contrast to conventional mathematical approaches. Finally, NNs have a highly parallel architecture that makes their processing speed extremely fast.

Chapter 2 : Backprogation Neural Network Controller

Backpropagation neural networks have basically three layers, referred to as input layer hidden layer(s) and output layer. Each layer contains a number of processing elements and is fully connected to the next layer.. The input data vector presented via the input layer, which fans out the input data without making calculations. The data flows along the connections toward the hidden layer (s0 and the output layer. The final result of this operation is that the input data vector is transformed (mapped) into some corresponding output vector at the output layer. Each processing element in a hidden or output layer has a connection from each processing element in the preceding layer. Associated with each of these connections is an adaptive weight. The output of a processing element in a hidden or output layer is calculated by applying an activation function to the weighted sum of the input to that processing element. Various activation functions such as S-shaped functions (sigmoid) and bump function (Gaussian) are available.

During the learning phase of the network, the actual output data vector is compared to the desired output data vector and the errors between these two vectors are calculated. The error values are then used to calculate

the new weights for all output and hidden layers processing element and thereby reduce the error in the network output. This process is repeated until the mapping from the input vector to the output vector has been trained to the desired accuracy. The idea is to find a set of weigh values that result in maximum accuracy and minimum error. The error criterion used by backpropagation networks is the mean squared error (MSE).

2.1 Call Admission Controller

Before we explain the NN controller, we shall discuss the admission control algorithm. We consider an ATM link with capacity C_{link} bits per sec. We assume multiple sources with multiple classes of traffic (i.e., different traffic characteristic and QoS requirements). Our measure of QoS is the cell loss rate (CLR). Other measures such as cell delay, delay variability are also possible. We consider that there are N virtual paths supported over the link. Each virtual path supports sources with similar class of traffic. For example, two types of video traffic sources with different QoS will be supported on separate virtual paths. In this paper we consider only two virtual paths, however the model can be easily extended to include multiple virtual paths. Also, if two traffic sources share the same QoS but have different characteristics each will be supported on a separate virtual path.

The reason for this classification is to simplify the controller design since it is easier to design smaller size neural networks each trained to learn a specific traffic pattern, rather than a complex architecture capable of learning a large set of traffic patterns.

In this section we provide the detailed design of our model. It consists of two modules: Module I and Module II and a decision-maker (see Fig. 1). Module I calculates the capacity required per call on each virtual path using on-line traffic measurements from its count process N (t, t+Ts) which counts the number of cell arrivals during an interval Ts. Module II calculates the capacity required for the combined multiple virtual paths on the link. The purpose of calculating the capacity on the link is to include the capacity gain due to class multiplexing and determine the amount of unused link capacity that can be allocated to individual virtual paths. As shown in Fig. 1, Module I consists of two levels of neural networks. It learns the relationship between actual traffic behavior, QoS and the required capacity. It calculates the capacity per individual virtual path C_{vpi}. The values are then used as input to module II that calculates the capacity for the combined multiplexed virtual paths. In subsections II.1 and II.2, we shall elaborate on the details of each module. The decision-maker in Fig. 1 consists of adders and subtractors. It makes the decision

to accept or reject the calls based on the values of its inputs.

We designed our call admission control model to be used in the case of static or dynamic capacity allocation. In case of static capacity allocation, the network management layer will permanently assign the capacity of each virtual path. The decision-maker in this case will make a decision to accept or reject the call based on the following equation:

$$\alpha_i = C_{vpi} + X - C_{ai} - C_{new} \qquad (1)$$

Where :

α_i is the output value from the call admission controller. If this value is greater than or equal zero, then the new call will be accepted otherwise it will be rejected.

C_{vpi} is a fixed capacity assigned for virtual path (I) by network layer.

C_{link} is the total allocated capacity for all virtual paths on the link. Module II is used to calculated this capacity from the used capacity of each individual virtual path C_{ai}.

C_{ai} is the actual capacity allocated on virtual path (I). Module I is used to calculated this capacity using on-line traffic measurement. Note that according to number

of calls supported by the virtual path, C_{ai} is always less than or at most equals C_{vpi}.

C_{new} is the capacity calculated for a new call.

X is the gain due to class multiplexing and is calculated using the following equation:

$$X = \frac{\sum C_{vpj} - C_{link}}{N} \qquad (2)$$

Where N is the number of virtual paths.

In the case of dynamic capacity allocation, the capacity of each virtual path will be requested from the network management layer dynamically. The decision maker in this case will generate a request for the network layer to increase (decrease) the capacity required for the virtual path based upon requests from incoming calls requesting bandwidth over this virtual path. The capacity required for the virtual path to accept an incoming call is determined from:

$$C_{vpj} = C_{ai} - C_{new} \qquad (3)$$

If the request for C_{vpi} is granted, then the call is accepted, otherwise it is rejected.

In the following subsections the detailed design of modules II and I is described.

2.2 Neural Networks Module I

In this module, multiple three layered back propagation NN were trained to learn the relationship between the characteristics of the cell arrival process, the QoS, and the capacity required to support it. The inputs to the NN are the former two while the latter is the output. The arrival process is measured via its count process which keeps account of the number of cells arriving in consecutive time periods. For this work, we are interested in enabling the NN to predict future samples of this high time-varying process which is also very bursty with significant correlation. For that, we have trained the NN to capture the correlation that exists among cells' arrivals. As we shall elaborate, our choice of a measurement period and a sampling period has been dictated by our need to teach the NN the correlation of the arrival process. Previous research studies [17][18] have proved that this multimedia process can be characterized by its index of dispersion for counts (IDC) since it is a good indication of the correlation and variability of the process. Hence, in validating the results we used the IDC as a measure of the accuracy of the predicated process. The QoS was presented via the cell loss rate, which also indicates the size of the buffer used. It is easy to add two more inputs to

indicate the cells delay and delay variability (jitter), if desired. The output from the module is then an accurate estimate of the bandwidth, which is required to support the input arrival process. It is important to mention that this estimate is accurate regardless of the type of the input traffic or the number of multiplexed sources. This is because our design is scalable and parallel in the sense that the NN module is composed of a bank of n parallel smaller NN units, each was trained to learn the characteristics of a certain class of traffic with similar bandwidth requirements. To add on a new service or a new type of multimedia traffic, one has to add on a simple NN unit to the already existing bank. The NN module is represented by:

$$\text{Capacity} = NN_f\{\mathbf{H}(I), [W]\} \qquad (4)$$

Where:

Capacity represents the output from the neural network model.

NN_f denotes neural network transfer function.

[W] Represents the weight matrices of the hidden and output layers.

[H (I)] is an input vector with K elements. Each element is a value of the cells count process N $(0,T_s)$ during a sampling period T_s measured over a measurement period (T_m) where $T_m=KT_s$.

The two parameters T_m and T_s were determined in order to capture the correlation of the traffic while keeping the size of the NN simple. For example a large value of T_s would imply a smaller size input vector **H (I)**, hence smaller number of input PEs and simple NN structure. However, the price is that the NN would not be able to capture the statistical characteristics of the traffic, leading to poor estimate of the required bandwidth. In this work, T_s was set to the duration of one video frame (1/30 sec.) since we neglected spatial variations of the bit-rate process per frame. One could also choose the duration of a slice (1/900 sec.) to be T_s since the slice is the unit of information processing in variable bit-rate (VBR) coding techniques. However, the correlation among consecutive slices is much smaller than those among consecutive frames. This is due to the fact that variations reflected by scene changes are more likely to happen over longer frames than over the very small slices. Similarly, the duration of the measurement period T_m must be chosen such that the effect of correlation, reflected by a suitable measure such as the autocovariance or IDC, among consecutive video frames is well captured. In Fig. 2 we show the effect of the variability of the cell arrival process for a VBR video traffic source. Experiment data from 10 minutes MPEG-I "traces" data files are down loaded from University of

California Berkeley, which is available in public domain. The data is coded using UCB coder and recorded in bits per frame. There are 30 frame per second with 160x120-frame size and it is coded using IBBPBB frame pattern. It is clear, that as the NN learns the characteristics of the traffic, by capturing the correlation, the variations of the calculated bandwidth decreases down to less than 3 Mbps out of the total link capacity (155 Mbps) after approximately 600 frames (percentage of error is less than 2%). This is verified by Fig. (2.b) which shows that the IDC also saturates at approximately the same value (600 frames). Hence, we concluded that the NN has learned the relevant traffic statistics for the estimation of the bandwidth. Furthermore, to validate our model, we used another set of MPEG-I traces, which are coded using Futuretel hardware coder. We used two traces each one of them is 90 minutes long with a frame size 320x240 and a frame rate 30-fps. This coder uses variable distortion coding so that during a high action or colorful scene, the picture quality is lowered so that the coder can maintain its target rate, which in this case is 1.2 Mbps. The frame pattern IBBPBBPBBPBBPBB is used in this hardware coder. In this case, the correlation was much shorter than the former case, and about 80 frames were enough to capture that correlation. Also, we used the Auto-regressive simulation model introduced in [21]. That model simulated head-and-shoulder pictures without scene

changes. We expected that correlation of much shorter duration than those of the 10 minutes MPEG-I trace. Indeed, in this case we needed only 30 frames to capture the correlation of the traffic and produce accurate estimation of the required bandwidth. Hence, we selected our measurement period Tm to be 600 frames with 600 input to the NN, since this range would be enough to cover all the cases under study.

Now after determining the size of the inputs, we had to determine the number of hidden layer PEs. We realized that there could be infinite number of traffic patterns (whether from a single or multiple sources) that could cover the range of bandwidth from 0 to 155 Mbps (OC-3 interface standard). This implies a huge number of PEs in the hidden layer, adding to the complexity of the NN architecture. Moreover, results reported in [19] proved that the number of patterns that a NN can deterministically recognize is equal to twice the number of its weights. Hence, to keep the architecture of the NN simple, adopted a hierarchical structure using banks of parallel simple NN units. The hierarchical structure has been used before in [20], and proved to be effective in learning different features of patterns that the single neural network approaches. Another advantage of the hierarchical approach is that the PEs of the hidden layer only have to process a limited number of traffic patterns

one step at a time, hence the hierarchical model can provide capabilities that otherwise would not be possible to deliver.

To use the hierarchical approach in our model, we divided the elements of the count process input vector into two levels or groups. The first level represents the cell arrival rate in 20 frames together with their equivalent bandwidth, whereas the second level contains 30 elements representing the bandwidth output from the first level and the required bandwidth for the whole traffic. Equivalently, these 30 elements contain the information that represent the cell arrival rates in 600 frames of video traffic. However, the neural network model is still facing a problem. Since the number of inputs for a single neural network is 20 and the range of these inputs should cover a bandwidth range from 0 to 155 Mbps. This means that a large number of traffic patterns would still be learned by the neural network implying a massive number of neurons in the hidden layer. To simplify the design we decided to divide the traffic into multiple ranges according to its bandwidth requirement and assign a neural network unit to learn each range. Therefore we assigned a neural network for traffic requiring bandwidth from 0 to 15 Mbps, another neural network for traffic requiring bandwidth from 15 to 30 Mbps and so on until 155 Mbps. This design gave us an extra important feature,

which is flexibility to extend and build upon it. Since all that is required to add more traffic patterns with new requirements is to add on an extra NN unit, or more, according to the new requirements.

Figure 3 depicts the model used to calculate the bandwidth from the real traffic. This model has two phases of operation. The training phase and the "on-line" operation phase. In the training phase, the first of neural networks was trained to recognize the relationship between a count process vector consisting of 20 elements, QoS (cell loss rate) and the capacity requirement for these 20 elements. The input training data set consisting of traffic measurement, QoS and the output (capacity) were obtained using extensive simulation studies. Discrete-event simulations, written in C, were used to accurately model the statistical multiplexer performance under different class of video traffic with different characteristics and QoS requirements. We were interested in teaching the NNs as many different traffic situations as possible. The important feature of NNs is that once it is trained to associate a certain output with a specific input, it can easily generalize this association to new inputs and generate an accurate estimate of the corresponding new output.

In the on-line operation phase, outputs from module I represent the allocated bandwidth for on-line traffic based on its count process vector. The comparators in Fig. 3 are used to allow only one output at one time from different neural networks in the first level. The function of the comparators is to compare whether the output capacity is from the neural networks falling within its specified range, then this output will be passed to the OR function. Otherwise the comparator will generate a zero output. The shift registers in this model will hold the input to each neural network level.

The second level of neural networks learns the relationship between the 30 output from the previous level and the equivalent capacity assigned for the traffic. Hence, the number of inputs for all neural networks in this model is kept to 30 inputs.

2.3 Neural Networks Module II
Statistical Classes Multiplexing

The objective of Module II is to include the gain achieved from statistical multiplexing of multiple virtual paths each with different class of traffic and different bandwidth. For example, consider two virtual paths VP1 and VP2, each is supporting a certain number of calls, let us assume that the bandwidth computed for each

virtual path, from Module I, is 10 Mbps and 34 Mbps respectively. The total assigned capacity for both virtual paths I not 44 Mbps (the sum) but a value that is less than that, say 40 Mbps. Hence the 4 Mbps gained, is called class multiplexing gain.

Module II also uses three layered back propagation neural networks to learn the relationship between its inputs and outputs. A NN in module II is responsible to learn the relationship between the capacity assigned for each virtual path (from Module I) and the actual assigned capacity for these paths on the link. For two virtual paths, the structure of the NN is simple. It has two inputs representing the bandwidth assigned for each virtual path, four hidden neurons, and one output which is the required capacity to support these paths. It is worthwhile mentioning that the controller, also, computes the requested increase (or decrease) in the capacity for each virtual path based upon a suitable entries such as a desired cell loss rate per virtual path. To show the gain achieved from using this model, it will be compared with other four rules for bandwidth assignment. These four rules are peak assignment, bit-rate stationary approximation, class-related rule and simulation. To make this comparison the following set of parameters are defined for each class:

W_i is the bandwidth assigned for each class

(B_p, B_m) are the source traffic descriptors, peak and average bit-rates respectively.

n_i is the number of connected sources

$A_i=n_iB_{mi}$ is the total average bandwidth of the class I.

$A_O=\Sigma n_i*B_{mi}$ is the total offered average bandwidth of the m classes

W is the total bandwidth assigned for all classes.

The main problem will be solved by using these rules is to find the minimum total bandwidth needed to serve these different classes with the required cell loss rate for each class. In the following, we describe the bandwidth rule for each technique.

a) Peak Assignment: In this method the statistical multiplexing is not considered among the sources in the same class nor among different type of classes. The bandwidth assignment rule is:

$$W=\sum_{i=1}^{n} W_i \qquad (5)$$

$$W_i=n_i*B_p \qquad (6)$$

b) Stationary State Approximation: In this method statistical multiplexing among sources in the same class is considered. However, statistical multiplexing between

different types of classes is not. The bandwidth assignment rule is:

$$W_i = W_{ss} \quad (7)$$

$$W = \sum_{i=1}^{n} W_i \quad (8)$$

Where:

W_{SS} is the bandwidth assigned for each class based on stationary state approximation [9].

c) Class related Rule: In this method statistical multiplexing among sources in the same class is considered and statistical between different type of classes is partially considered. The bandwidth assignment rule is [22]:

$$W = MIN(W_w A_o, \sum_{i=1}^{n} W_i) \quad (9)$$

Where:

W_W is the maximum value obtained on the assignment curve representing the behavior of the class with the largest burstiness in correspondence with total offered average bandwidth.

W_i is bandwidth assigned for a class calculated from simulation.

d) Simulation: In this case, simulation is used to calculate the total bandwidth assigned for all classes. Clearly, statistical multiplexing among the sources in the same class and the statistical multiplexing among different types of classes are fully considered.

e) Neural Networks: In this case, our neural networks model has been used to calculate the bandwidth required for two video classes multiplexed on the same link.

These five techniques are used to calculate the bandwidth required for two virtual paths assigned for video traffic. The first virtual path is assigned for class video traffic with cell loss rate 10^{-4} while the other virtual path is assigned for class B video traffic with cell loss rate 10^{-6}. Results from these techniques are compared in Fig. 4. In this figure the two axes represent the offered mean bandwidth [Mbps] of the two-video classes. Each point in this graph represents a combination of xa Mbps offered by class A and yb Mbps offered by class B. in this figure we can draw the limit curve of the total acceptable average bandwidth according to the previous five methods. In Fig. 4, the limit curve is drawn for video sources with average=3.9 Mbps and variance = 1.9 Mbps. As shown in the figure, results

from neural networks are more accurate in comparison with other methods.

2.4 Simulation and Numerical Results

Extensive simulations were performed to obtain the data set, for both training and operation phases, and also to assess the performance of various neural networks architectures. The video sources were simulated using first order Autoregressive Markov model in [21]. The equation used to simulate this model is:

$$\lambda(n)=a\lambda(n-1)+bw(n) \qquad (10)$$

The steady state average E (λ) is:

$$E(\lambda)= b*\eta/(1-a) \qquad (11)$$

And discrete autocovariance is:

$$C(n)=b^2*a^n/(1-a^2) \quad n>=0 \qquad (12)$$

Where:

$\lambda(n)$ Is the bit rate during nth frame.

$w(n)$ is a sequence of independent Gaussian random variables

With mean η=3.9 Mb/s and variance=1.

a=0.8781 and b=0.1108.

Using this autoregressive model, we tried to generate video traffic with all possible variation in single link with a capacity 155 Mbps. The number of video sources supported by a single link with maximum capacity 155 Mbps range from 1 to 27 sources. Furthermore, experimental

data-traces from two MPEG-I sources were also used to validate the results. The neural networks were simulated using HNC Inc. EXPLORENET 3001 package [23]. Several experiments were performed to tune up the parameters of the neural networks. The activation function of these neural networks were selected to be sigmoid where the steepness parameter (a factor that determines the non-linearity of the sigmoid function) was selected by trial and error method to obtain good results.

In this section we demonstrate the effectiveness of the proposed models to allocate the effective bandwidth for VBR video traffic. We decided to compare results from our models with those of stationary state model because both use the same type of traffic.

Results from simulation, neural networks and stationary state model are shown in figures 5 and 6. Results from stationary state model are obtained using the following equation:

$$C_{(S)} = \lambda + a\sigma \qquad a = \sqrt{-2\ln(\varepsilon) - \ln(2\pi)} \qquad (13)$$

Where:

$C_{(S)}$ Equivalent capacity

λ Mean aggregate bit

ε Cell loss rate.

σ Standard deviation of the aggregate bit rate.

For neural network model, it was found that only 40 neurons in the hidden layer provided very good results. Actually we started with 20 neurons in the hidden layer in our experiments and then we increased them until we obtained very good results at 40 neurons in the hidden layer.

In Fig. 5 the effective bandwidth allocated for different number of MPEG-I sources with buffer size 40 cells and cell loss rate 10^{-6} is shown. It is clear from this figure that results obtained from neural networks model are more accurate than results of stationary state model. At the same time, the percentage of error in bandwidth allocation is small compared with the simulation results. Although in real-life ATM networks, the CLR for video services are less than 10^{-9}, it was not possible to simulate such rare events with reliable results. Hence, we used 10^{-6} as figures of merit.

To further assess the validity of the model, we applied different sequences of video traffic with different characteristic (i.e., variance average) to this model. To maintain the cell loss rate at the required value, different effective bandwidth should be allocated to each sequence. Results are shown in Fig. 6. It is clear that the effective bandwidth from simulation is very close to

that calculated by the model. The small error introduced is due to limited number of inputs that characterize the variability of the traffic, to only 600 frames. Clearly, there is a trade off between the complexity of the NN architecture and the accuracy of results. For example, if the number of inputs were increased, the error would be even smaller; however, the price paid would be a complex the NN structure.

Figures 7 and 8 show the effective bandwidth calculated for two classes of video traffic generated using autoregressive model. In Fig. 7, we show result of multiplexing two sources of class B with sources of class A. In Fig. 8 we increased the number of sources in class B to 14. It is clear that statistical multiplexing gain is more effective in the latter case than the former one.

Chapter 3: Radial Basis Function Controller

A Radial-basis-function (RBF) network is a 3-layer feed-forward NN that is based on traditional RBF approaches to multi-variable interpolation[25]. It consists of an input-hidden nonlinear mapping followed by a hidden-output linear mapping. Each hidden unit (neuron) computes a nonlinear function of a measure of the distance between the network inputs and the unit's weight vector (usually called the "center" of the unit). Thus, the linear output of the network is expanded on a baisis given by a set of radially symmetric functions. In practice, the output of a hidden unit peaks when the input is at its center and falls off monotonically as the input moves away from it. Thus, each hidden unit may be viewed as a classifier that is triggered over some specific range of the input data. In our model we used the Gaussian basis function for the hidden units. The output from each hidden unit is given by :

$$Z_j(x) = \exp\left(\frac{\|x - \mu_j\|^2}{2\sigma^2}\right) \qquad (14)$$

As we can see from equation (1) The output of hidden unit Z_j is obtained by calculating the "closeness" of the input x to an n-dimensional parameter vetcor μ_j

associated with the jth hidden. The output from the RBF is given by:

$$Y_j(x) = \sum_{j=1}^{J} W_{ij} Z_j(x) \qquad (15)$$

The RBF neural network has been used in our model to estimate the bandwidth based on the following equation:

$$BW_n = F(BW_{n-1}, H, CLR, B) \qquad (3)$$

where Bw_n is the required bandwidth estimated at the end of monitoring period n. BW_{n-1} is the required bandwidth estimated at the end of period n-1. Each BW_n and BW_{n-1} are calculated over a monitoring period T_m where $T_m = KT_s$ where T_s is the sampling interval which is set to 10 frames. The choice of 10 frames is done through trail and error process. As the sampling period decreases, the bandwidth calculation time decreases but the overlaping among patterns for different number of sources is increases which cuase a difficuty for classifying the patterns. As the number of frames increases, the overlaping between patterns decreases but the bandwidth calculation time increases. We tried different sampling period starting from one and up, until we found the overlaping between patterns fanish at 10 frames. This result is clearly shown in figure 9 and figure 10. H is a

value of the cells count process $N(0,T_{s1})$ during sampling period T_{s1}, CLR is the cell loss rate, B is the buffer size and F is the nonlinear function that the RBFNN is trained to estimate.

One of disadvantages of RBF neural network is that it require excessive number of hidden units to achieve high precision, specially when the dimensions of the input is high. To simplify the design of our model and to reduce the dimension of the input to the RBF neural network, we simplified equation (16) to :

$$BW_n = F(BW_{n-1},H)+F(BW_{n-1},CLR) +F(BW_{n-1},B) \qquad (17)$$

Using equation (17) we designed our model using three RBF neural network modules: namely, Module I, II and III as shown in figure 11. Using three modules instead of one module reduces the dimesions of the inputs and consequently, the number of hidden neurons in each of them. Module I is used to calculate the bandwidth required for cell loss rate CLR_i and buffer size B. The inputs to this module are BW_{n-1} and H. Module II is used to calculate the difference between the required bandwidth to achieve cell loss rate CLR_j and the required bandwidth to achieve cell loss rate CLR_i. The inputs to

this module are BW_{n-1} and CLR_j. Module III is used to calculate the difference between the required bandwidth when buffer size is equal to B_j and the required bandwidth when buffer size is equal to B_i. The inputs to this module are BW_{n-1} and B_j. By adding the output from these three modules we obtain the bandwidth required to achieve cell loss rate CLR_j when buffer size B_j is used. To elaborate a typical operation of the model consider that it is required to calculate the bandwidth required for a video traffic with CLR of 10^{-10} and buffer size 100 cells. Also, let's assume that module I has been trained to calculate the bandwidth required for the same video traffic but for a different CLR of 10^{-6} and buffer size 40. In this case the output from module I will always be the required bandwidth for cell loss rate 10^{-6} and buffer size 40. However, the output from module II will compensate the difference between the required bandwidth for CLR 10^{-10} and the required bandwidth for CLR 10^{-6}, whereas Module III will compensate the difference between the required bandwidth for buffer size 40 and the required bandwidth for buffer size 100. By adding the output from these modules, we obtain the required bandwidth for cell loss rate 10^{-10} and buffer size 100. In the following, we will describe the details of learning and training phases of each one of these modules.

3.1 RBF Neural Network Module I

Module I is used to calculate the capacity required for cell arrival rate at cell loss rate CLR_i based on the following equation:

$$BW_i = (BW_{i-1}, H) \tag{18}$$

The inputs to this module are two parameters, the bandwidth required for traffic BW_{i-1} calaculated from previous samples and the cells count process H at time T. The output is the bandwidth required to support the cell loss rate CLR_i over measurment period T_m. This module has two phases of operations: the learning phase and the operation phase. The learning of the RBF neural network is a process by which the free parameters (means of the hidden layer Gaussian units, widths (standard deviation), and the output layer weights) are updated in an adaptive way to minimize a cost function. There are different learning strategies that can be followed in the design of a RBF network. In our model, we used a training strategy that decouples learning at the hidden layer from that at the output layer. This strategy has been proved to be very effective in terms of training

speed[26]. In the following we will describe the method has been used to locate the centers and widths of neurons. Then we will describe the algorithm that has been used to compute the weights of the output layer.

Several schemes have been suggested to find centers and widths of a RBF neural networks[27][28]. First, we have to use an algorithm to find a proper means of the RBF then use a another heuristics method to find their widths. The method we used to locate the means is suggested by Broomhead and Lowe[29]. In this method, the centers of of the RBF neurons has been placed according to some coarse lattice defined over the input space. Assuming a uniform lattice with k divisions along each dimension of an n-dimensional input space, this latice would require k^n basis functions to cover the input space. Although this method normally requires many hidden units, especially for high dimension inputs, it is very proper to solve the link capacity problem for two reasons: First, the inputs are just two dimensions. Hence,. the number of hidden neurons should not be that much. Secondly, distribution of the input data of the problem space is known as shown in figure 12. Hence, we can select the centers of the hidden neurons more accurately. To simplify the design of our module we used

15 video sources only covering the range (0-18 Mbps) since each source bandwidth approximately 1.2 Mbps. However, the model can be extended easily to cover the space of bandwidth and cell arrival rate (0-155 Mb/s) dividing the 155 Mbps into fixed equally spaced ranges(18 Mbps each). Hence, nine more modules (similar to the one shown in figure 11) will be needed.

To select the centers of the RBF neural networks, we draw the bandwidth required for this 15 video sources verses the bit arrival rate per sampling period T_s as shown in figure 12. In selecting the centers of neurons, we tried to achieve the best accuracy with minimum number of neurons. To achieve that we conducted two experiments: In the first one, we gave equal accuracy for all the points located in the space of the problem, and divided the space of inputs into four squares. Then, we used their centers to be the centers of the RBF neural networks. The values of the centers in this experiment were <(4,4.5), (4,13.5), (12,4.5), and (12,13.5) >. In second experiment, we divided the input space to four areas,, A, B, C and D. If we carefully examine at these areas, we can classify them into two classes. the first one is the "steady state operation area" and it includes areas A and C. The reason for calling this area "steady state operation area" is that the RBF will usually go to

transient state then will find an accurate value for the bandwidth required within this area. As a result, areas A and C will need more number of neurons to increase the accuracy of RBF. The second class is called "transient operation area" and it includes areas B and D, they are called "transient areas" since the final solution for the bandwidth will never be in this area. As the RBF operates, it will go through some points in this area, and then it converges to the final solution which is located in steady state area. To achieve high accuracy while keeping the number of neurons as small as possible, we located four neurons in each of areas A and C. Also we placed one neuron in each of areas B and D. We consider the centers of the squares shown in figure 13 as the centers of the RBF neurons. The values of these centers were <(2,2.25), (2,6.75), (6,2.25), (6,6.75), (4,13.5), (12,4.5), (10,11.25), (10, 15.75), (14, 11.25), and (14, 15.75)>.

Once we determined the centers for the previous two experiments, we had to find a proper value for the widths. To determine the widths in experiment 1, we used the average width $\sigma = < \| \mu_i - \mu_j \| >$, which represents a global average over all Euclidean distances between the center of each unit I and that of its nearest neighbor j which

has been suggested in [28]. By applying this method in our model and by using Fig. 4 we can find that the value 8 represents a good estimation for the widths of RBF for experiment 1. For experiment 2. We calculated the width for each unit separately using the following equation:

$$\sigma = \alpha \| \mu_i - \mu_j \| \quad (19)$$

Where α is taken between 1.0 and 1.5. In our experiments we found that a good results are obtained when α equals 1.3. Using Fig. 13, we can find that a good estimation for the widths of RBF for experiment 2 is <.5.2, 5.2, 5.2, 5.2, 15, 15, 5.2, 5.2, 5.2 and 5.2>.

Once the hidden layer parameters are determined, supervised learning is employed to choose the weights of the linear output layer such that a specific cost function is minimized. We used Least-mean-square (LMS) algorithm to adjust the output layer weights and biases by performing gradient descent on the cost function to find its minimum value. Its steps can be described as follows:

The output weights and biases are intialized with small random values. An input pattern is randomly picked up from the training set and propagated forward through the network to compute the outputs bw_i . The weights are updated based on the following equation:

$$w_j(l)= w_j(l-1)+\varepsilon(s\_bw - nn\_bw)h_j \qquad j=1,...,c \qquad (20)$$

where w_j is the weight of the output layer connected to hidden neurons j , ε is the learning rate, s_bw is the bandwidth calculated from simulation, nn_bw is the bandwidth calculated by the neural network, h is the output from the hidden layer and l is the iteration number. these steps are repeated until the cost function is converged to minimum value. By selecting the output layer parameters, the training procedure of the RBF is completed..

The other problem we have to solve is to find the proper data to train our RBF neural network. As there are no analytical models that can adequately describe VBR video traffic in high speed networks, we used a finite sequence of data coded using MPEG-I to train the RBF neural network. We used two traces of MPEG-I. The first one is coded using University of California Berkley coder and recorded in bit per frame. There are 30 frames per second with 160x120-frame size and it is coded using IBBPBB frame pattern and it is 10 minutes long. The second one is coded using Futuretel hardware coder. There 30 frames per second with 320x240 frame size and it is coded using IBBPBBPBBPBBPBB frame pattern and it is 90 minute long. As previously explained, to simplify the RBF neural network we had to preprocess the data . As we can see an

overlapping between the data that representing a single source, three sources or five sources (see figure 10). This overlapping implies that more input samples and consequently more hidden neurons. To solve this problem we had to smooth the traffic before applying it to the RBF neural network. To smooth the traffic, we used the following equation:

$$x(t) = \frac{1}{Ts}\int_{0}^{Ts} x(t)dt \qquad (21)$$

Where x(t) is the cell arrival rate and Ts is the sampling period. Equation (21) is equivalent to a low pass filter operation. By passing the traffic through the low pass filter, we removed higher frequency components of the traffic. The implementation of this equation is simple. All that is required is to calculate the average over period Ts. We found that by using Ts=0.3 sec that is ten times the frame period, we got very smoothed traffic as shown in figure 10. Now, the RBFNN can easily classify different traffic patterns using a small number of inputs and using a small number of hidden neurons. Also, by avoiding pattern overlapping, convergence of the RBFNN occurs more rapidly since it can predict the traffic more accurately within a monitoring period T_m.

3.2 RBF Neural Network Module II and Module III

Module II is used to calculate the difference between the required bandwidth to achieve cell loss rate CLR_j and the required bandwidth to achieve cell loss rate CLR_i based on the following equation:

$$\delta BW_i = F(BW_{i-1}, CLR) \qquad (22)$$

In this module, we have two dimensional inputs (BW_{i-1},CLR)and one output. To select the centers of the RBF neural networks, we draw the bandwidth required for 15 video sources versus the CLR. Then we divided the space of inputs to four squares and used their centers to be the center of the RBFNN. The widths of the RBF neural networks have been calculated using the global average method that is the same one used in module I. The output layer of this module has been trained using supervised learning. To obtain the training data for this model we had to find the bandwidth required for very small values of CLR such as 10^{-8} and 10^{-10} . Practically it is very difficult to calculate the bandwidth for these values. Hence, we used an interpolation technique. We calculated the bandwidth required for cell loss rates up to 10^{-6}. Note that the bandwidth required for zero cell loss rate is the peak rate of the traffic. Hence, a RBF neural

network has been used to interpolate the missing points between 10^{-6} and zero CLRs as shown in figure 15. Module III is used to calculate the difference between the required bandwidth when buffer size is equal to B_j and the required bandwidth when buffer size is equal to B_i based on the following equation:

$$\delta BW_i = F(BW_{i-1}, B) \qquad (18)$$

In this module, we have two dimensional inputs (BW_{i-1}, B) and one output. To select the centers of the RBFNN, we draw the bandwidth required for 15 video sources versus the buffer size. We divided the space of inputs to four squares and we used their centers to be the center of the RBFNN. The widths of the RBFNN has been calculated using the global average method which is same method in modules I and II. The output layer of this module has been trained using supervised learning.

3.3 Simulation and Numerical Results

In this section we demonstrate the effectiveness of the proposed model to allocate the effective bandwidth for VBR video traffic. Also, we will evaluate our model when it is used for VBR voice traffic. First, we explain how

the neural network has been trained, then, we demonstrate the capability of neural network to allocate proper bandwidth for real video traffic with different characteristics. To train the neural network we used experimental data-traces from two MPEG-I sources. These MPEG-I traces consist of two files, pbride.t and cnn.t and they are 90 minutes long with a frame size of 320x240 and a frame rate of 30 fps. The file cnn.t is CNN news including commercials, and pbride.t is the movie Princess Bride. Using these files, we generated 60 video sources, each is 2000 frames long. We divided these 45 sources into three groups: each consists of 15 video sources. The first group has been used to train the neural networks, whereas the second group has been used to test it, and the third group has been used to validate the results. We generated from these sources 4000 patterns for training, 4000 patterns for testing and 4000 patterns for validating the neural network model.

Beside selecting the training data for RBFNN, we have to select the monitoring period T_m. The parameter T_m is selected to reduce the error in calculating the bandwidth. To find the relationship between the bandwidth error and T_m , we calculated the bandwidth error for different values for T_m as shown in figure 15. From this

figure we can see that as T_m increases the error in bandwidth decreases and as T_m decreases the error in bandwidth increases. Also, we can see from figure 15 that the bandwidth error reduces to minimum value after approximately 600 frames (percentage of error is less than 3%).

To evaluate our model, we decided to compare the results from our models with those of stationary state model because both use the same type of traffic. In figures 16 and 17 the effective bandwidth allocated for different number of sources with buffer size 100 and cell loss rate 10^{-6} and 10^{-8} is shown. In figures 18 and 19, the bandwidth is allocated for the traffic when buffer size equal to 400 cells and the cell loss rate 10^{-6} and 10^{-8}. In figure 20, the bandwidth has been calculated for 10 video sources with different values for buffer size. As shown in this figure, the bandwidth calculated by the stationary state has not been affected by the size of buffer while the bandwidth calculated by neural networks follows the bandwidth calculated by simulation and it decreases as the buffer size increases. It is clear from these figures that the results obtained from neural networks model are more accurate than those of stationary state model. At the same time, the percentage of error in

bandwidth allocation is small compared with the simulation results.

To see how the quality of video will be affected by the bandwidth allocated by our model, we applied our model on a video source generated from MPEG coder developed by University of California Berkley coder. We transmitted the video traffic generated by the MPEG coder through our model and we calculated the bandwidth required for different quality of service. three experiments have been done on our model. In each experiment we used different value for cell loss rate. For experiment 1, cell loss rate was 10^{-4}, for experiment 2, cell loss rate was 10^{-6} and for experiment 3, cell loss rate was 10^{-8}. In each experiment we calculated the bandwidth required for the video traffic using our model. This bandwidth value has been used by the coder , using the source rate control capability, to limit the video traffic and makes it fits into the specified bandwidth calculated by our model. The bandwidth calculated for experiment 1 is 0.9 Mb/S, while the bandwidth calculated for experiment 2 is 1.3 Mb/S and for experiment 3 the bandwidth is 1.8 Mb/S. Figures 21,22,23 show the quality of video for cell loss rate 10^{-4} , 10^{-6}, 10^{-8} respectively. As we can see from these

figures, when cell loss rate decreases we got better quality.

To extend the evaluation for our model, we used it to allocate the bandwidth for VBR voice traffic. The voice traffic has been generated using ON-OFF model. The ON and OFF time periods are assumed to be exponentially distributed random variables with mean 350 ms and 650 ms respectively. Using this model we generated a voice traffic for 300 voice sources which require a bandwidth in range from 0-18 Mb/S. To use the neural networks to allocate the bandwidth for voice traffic, the output layer should be re-trained to get an accurate results. The values for centers and widths will be the same as in video traffic and that is because the video sources and the voice sources cover the same range of bandwidth(0-18 Mb/S). To train the output layer for voice traffic, we use the same technique as in video traffic. We generated three sets of voice traffic. The first set has been used to train the neural networks, whereas the second set has been used to test it, and the third set has been used to validate the results. We generated from these voice sources 8000 patterns for training, 8000 patterns for testing and 8000 patterns for validating the neural network model.

To validate our model for voice traffic, the bandwidth has been calculated using simulation, neural networks and stationary state model as shown in figures 24 and 25. As shown in these figures, the bandwidth has been allocated for voice traffic using simulation, neural networks and stationary state model for buffer size equal to 100 and 400 respectively. From these figures we can see that the neural networks produce more accurate results than stationary state model for voice traffic.

To see how long it takes from the RBFNN to converge to the correct value of bandwidth, traffic patterns have been applied to RBFNN and they have been changed suddenly as shown in figure 26. From this figure we can see that the error in calculating the bandwidth from the RBFNN reduces to less than 3% within three sample periods.

To further assess the validity of the model, we applied different sequences of video traffic from different number of sources. To maintain the cell loss rate at the required value, different bandwidths should be allocated to each sequence. Results are shown in Figures 27 and 28. Figure 27 shows the output for RBFNN and simulation when

4 hidden neurons are used in module (I). Figure 28 shows the output for RBFNN and simulation when 10 hidden neurons are used in module (I). We can see from these figures that the RBF neural network converges to the accurate value for bandwidth from the second samples and also the results of using 10 hidden neurons is better than using 4 hidden neurons.

Chapter 4: Conclusions and Future Work

In this proposal, we presented a neuro-computing approach to the problem of bandwidth allocation in ATM networks. In particular we used neural networks to compute the effective bandwidth required to support MPEG-I VBR video calls with different QoS requirements. Our approach tackles the problem using two types of neural network, backpropagation neural network and radial basis function neural network. Performance evaluation results proved that the neural network approaches are more accurate in calculating bandwidth than traditional methods. In these models, adaptability to new traffic situations has been achieved by adopting a hierarchical approach to the design, whereby newly traffic patterns could be easily accommodated by adding extra neural network modules. These models also have the ability to make fast decision about congestion in ATM networks due to its dependency on on-line traffic measurement.

A natural extension for using neural network for video traffic is to use it for different type of sources such as voice and data. Also, the dynamic bandwidth allocation can be implemented using recurrent neural network which we will investigate its capability in our next work.

Publication

1. S. Youssef,I. Habib, T. Saadawi, " Bandwidth Allocation of Variable Bit Rate Video in ATM Networks Using Radial Basis Function Neural Networks," in Proc. ICC '1999.

2. S. Youssef,I. Habib, T. Saadawi, "Bandwidth Allocation in ATM using RBF Neural Network," Proc. SPIE '1999.

3. S. Youssef,I. Habib, T. Saadawi, "Allocation of Resources for Quality of Service Guarantees in ATM Networks" in Proc. ARTIP '1999.

4. S. Youssef,I. Habib, T. Saadawi, " Radial Basis Functions For Bandwidth Estimation in ATM Newtorks," in Proc. MILCOM '1998.

5. S. Youssef,I. Habib, T. Saadawi, "A Neurocomputing Controller for Bandwidth Allocation in ATM networks," IEEE J. Select. Areas Commun., January, 1997

6. S. Youssef,I. Habib, T. Saadawi, " A Neurocomputing Controller for Admission Control in ATM Networks," in Proc. ICC '1996.

7. S. Youssef,I. Habib, T. Saadawi, " A Neurocomputing Controller for Admission Control in ATM Networks," in Proc. Milcom '1995.

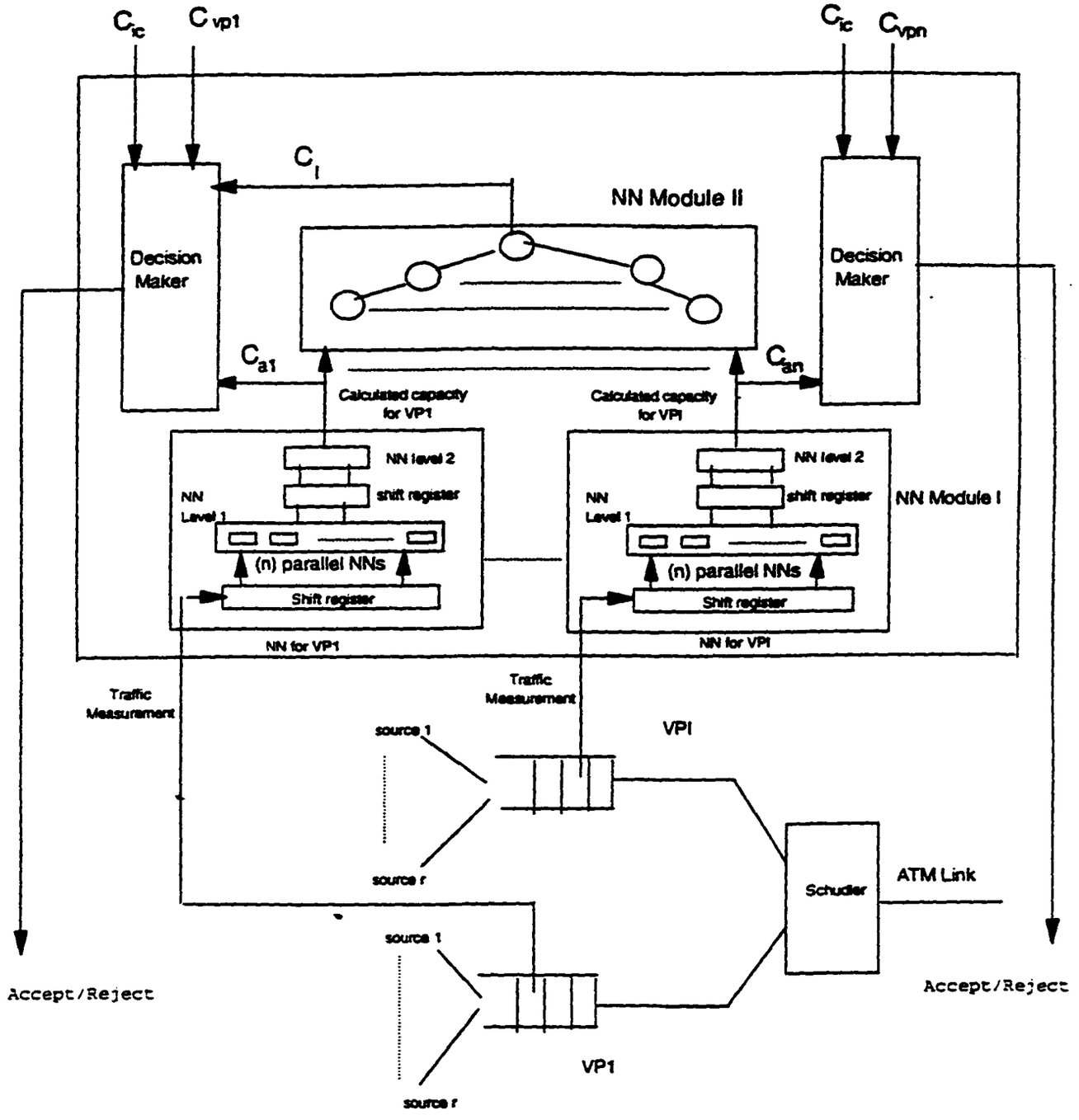


Figure 1: Call admission control model using backpropagation neural networks

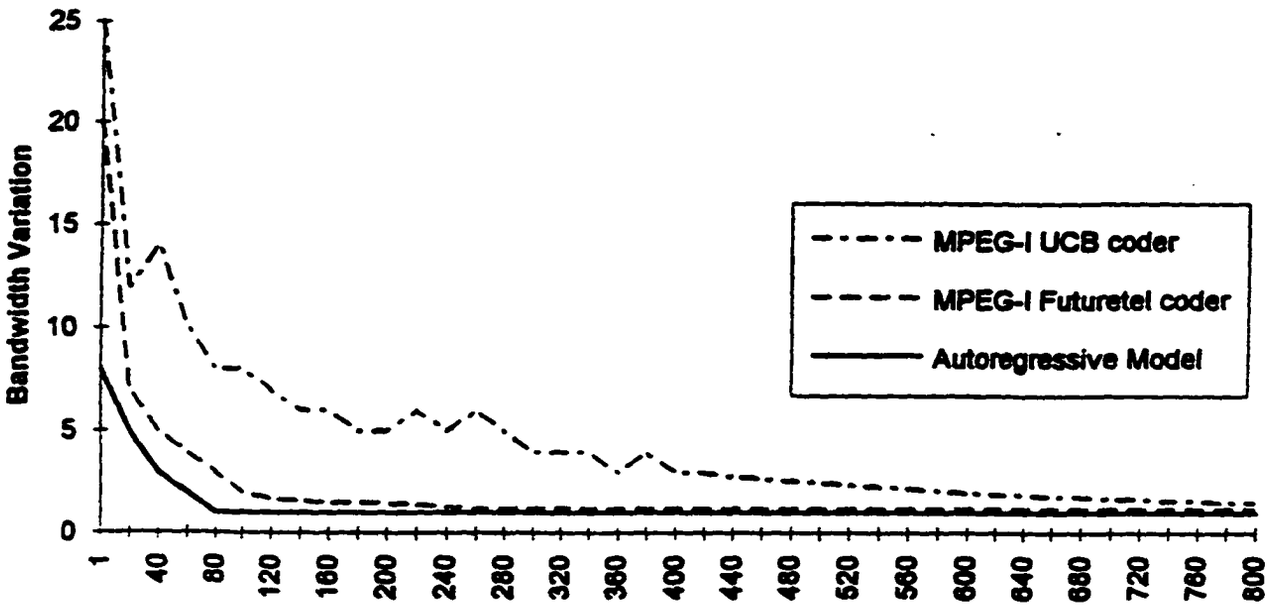


Figure 2: Bandwidth variation verses number of frames in count vector process

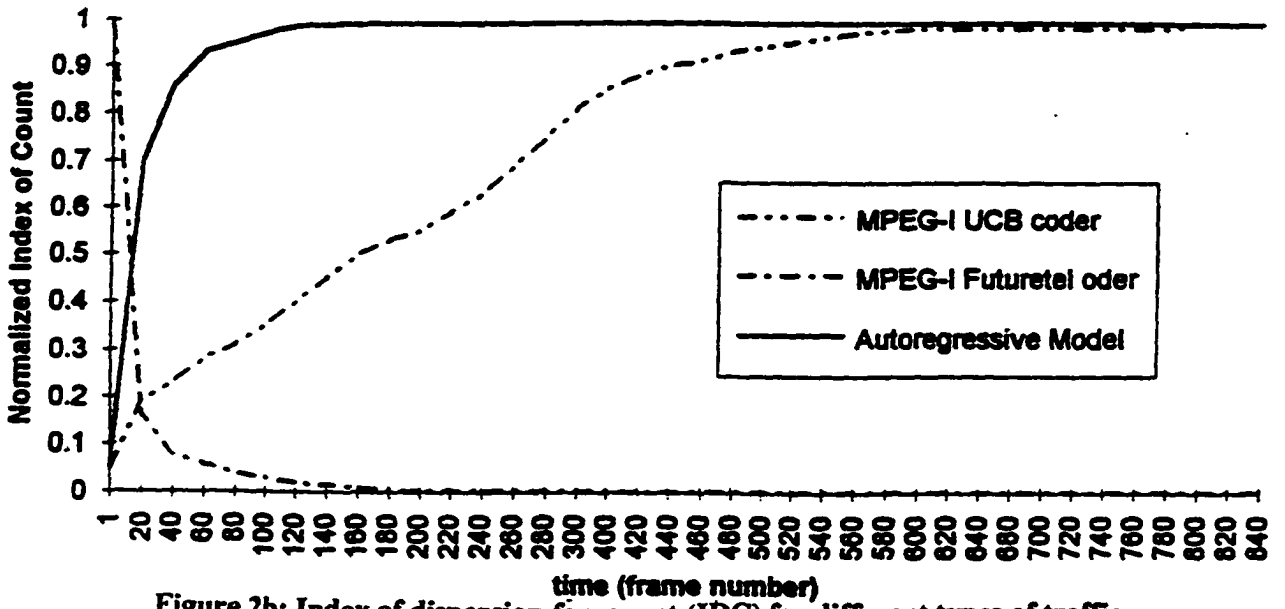


Figure 2b: Index of dispersion for count (IDC) for different types of traffic

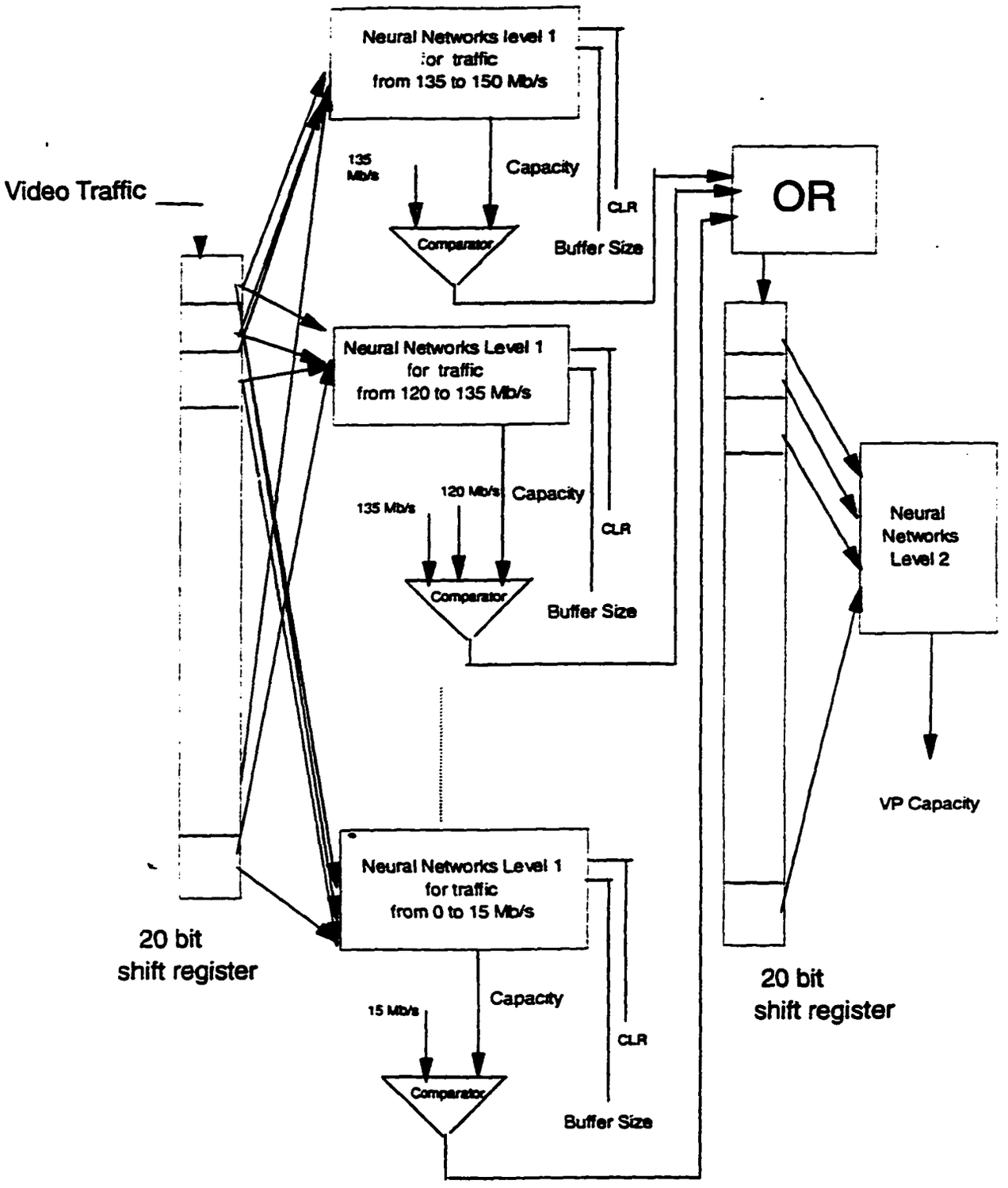


Figure 3: Hierarchical multi-level neural networks architecture that calculate VP capacity

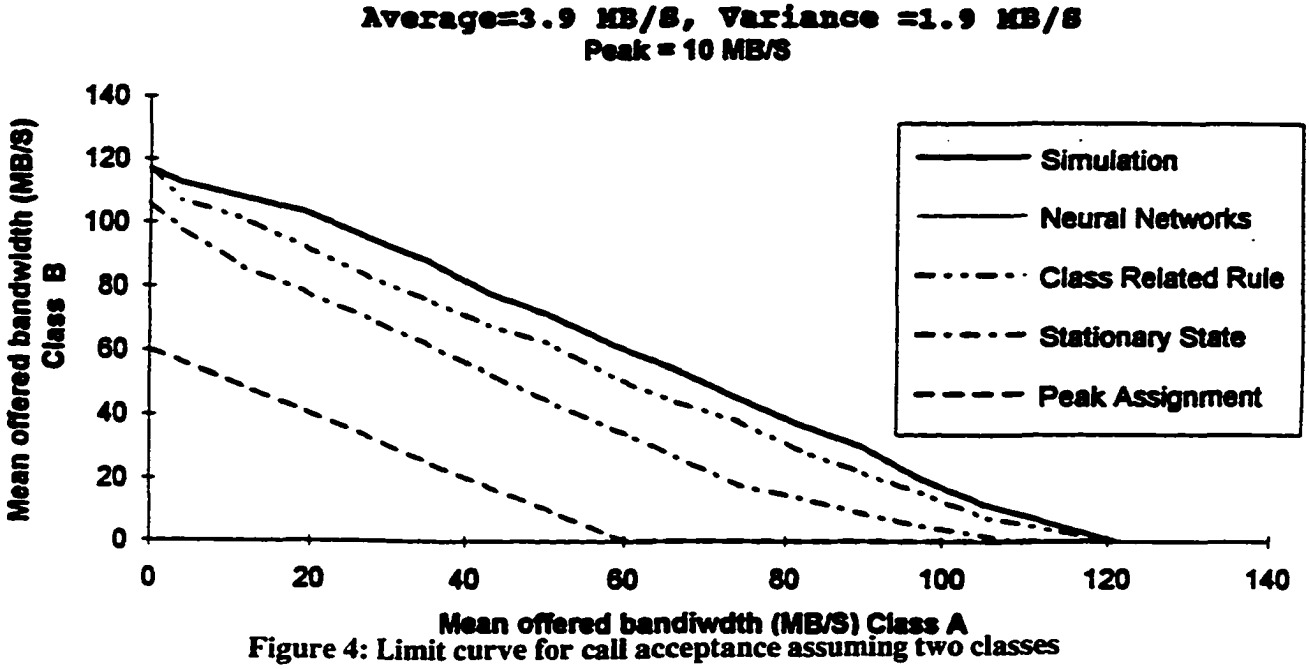


Figure 4: Limit curve for call acceptance assuming two classes

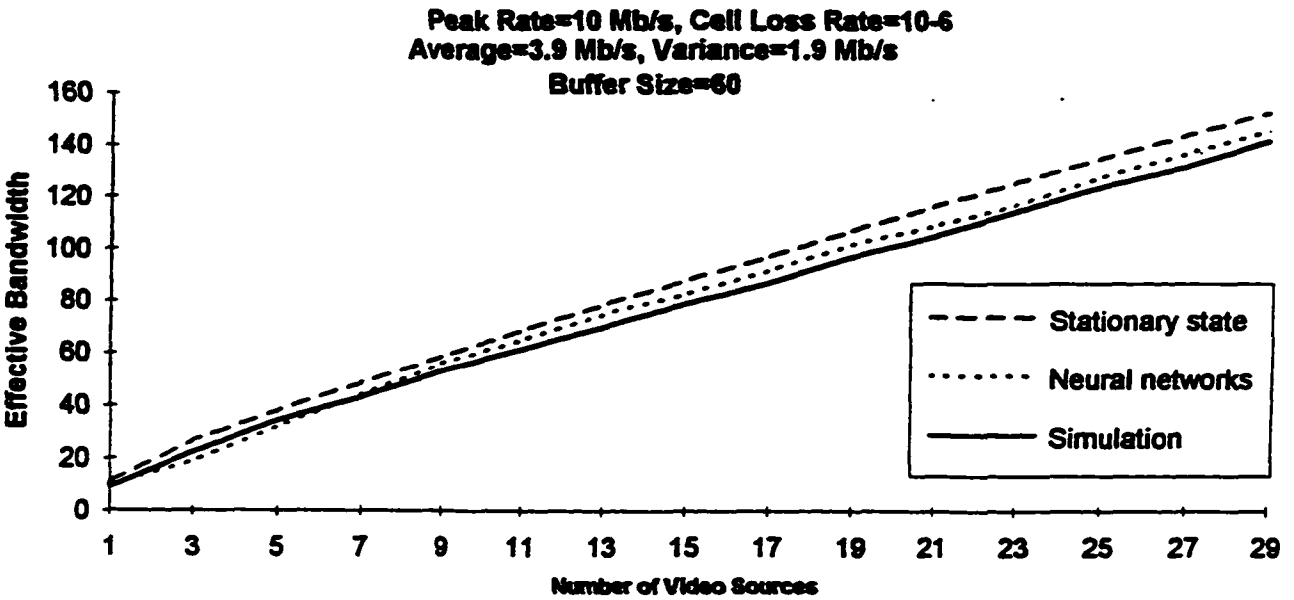


Figure 5: Effective bandwidth for simulation, neural networks and stationary state models Calculated by backpropagation neural networks model

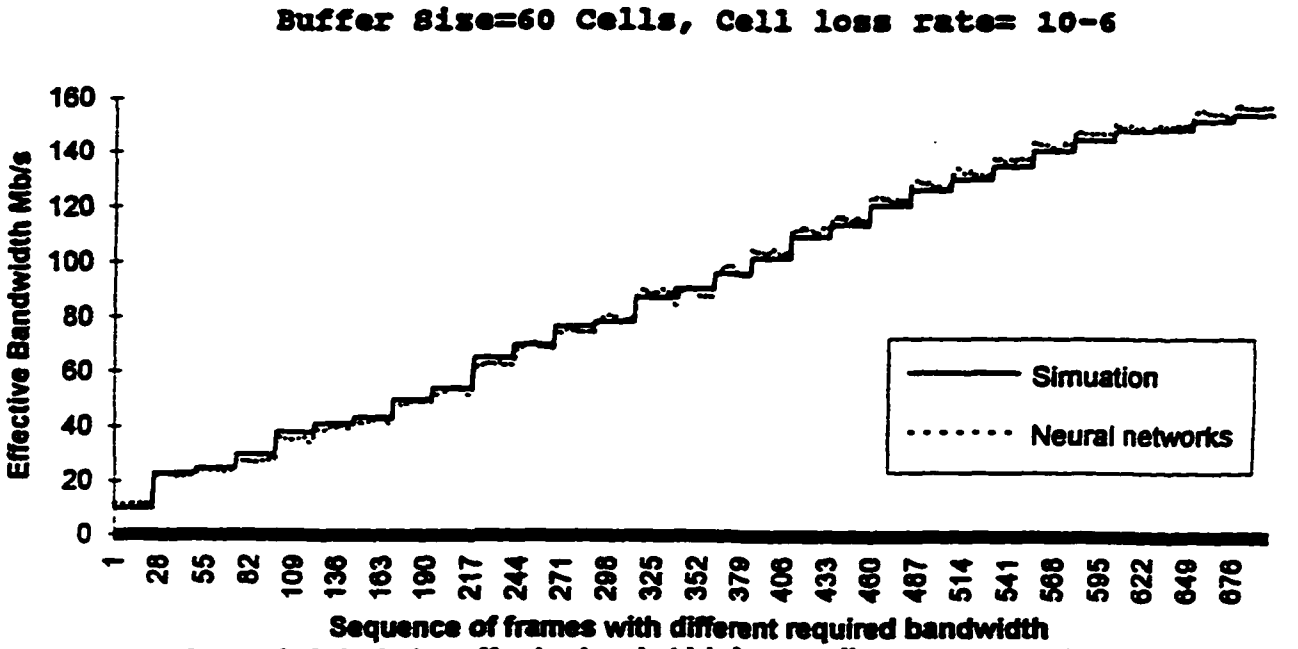


Figure 6: Calculating effective bandwidth from on-line measurements using backpropagation neural networks model

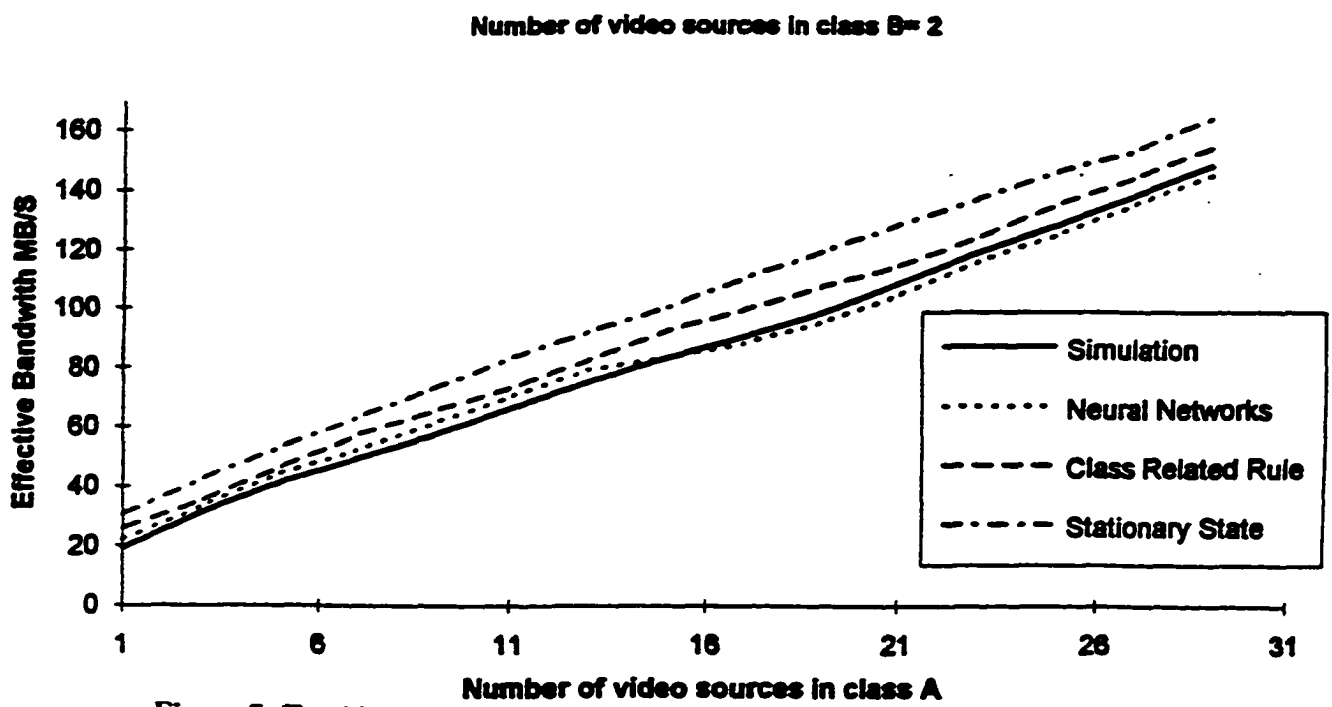


Figure 7: Total bandwidth assigned for class A and class B (2 video sources)

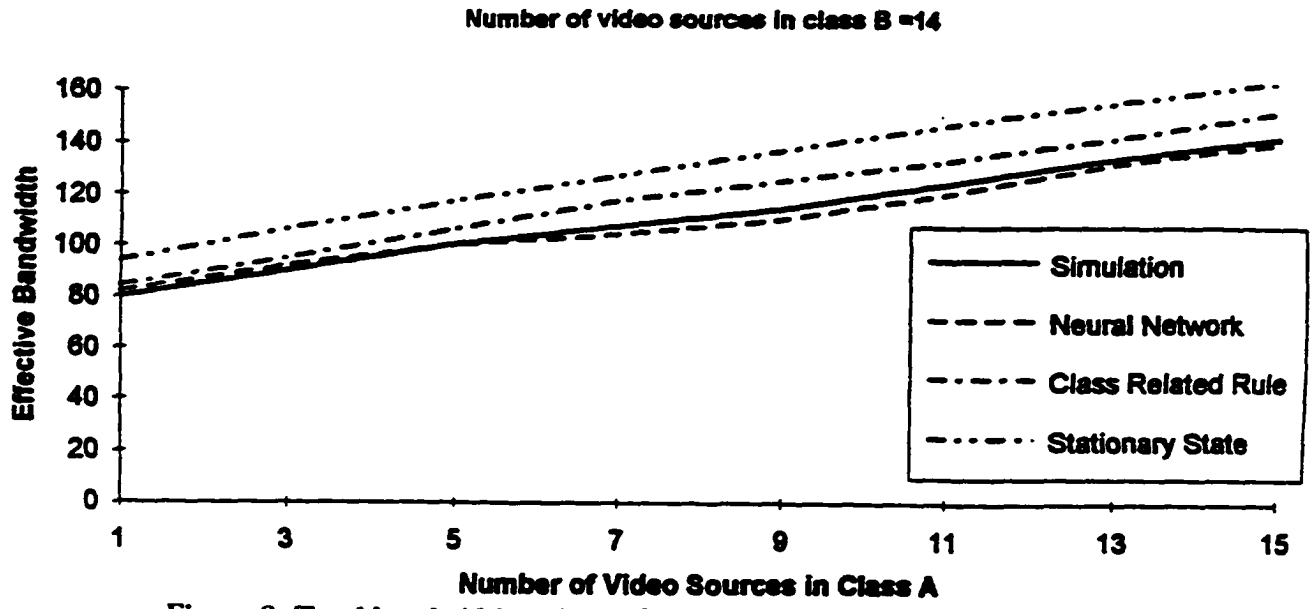


Figure 8: Total bandwidth assigned for class A and class B (14 video sources)

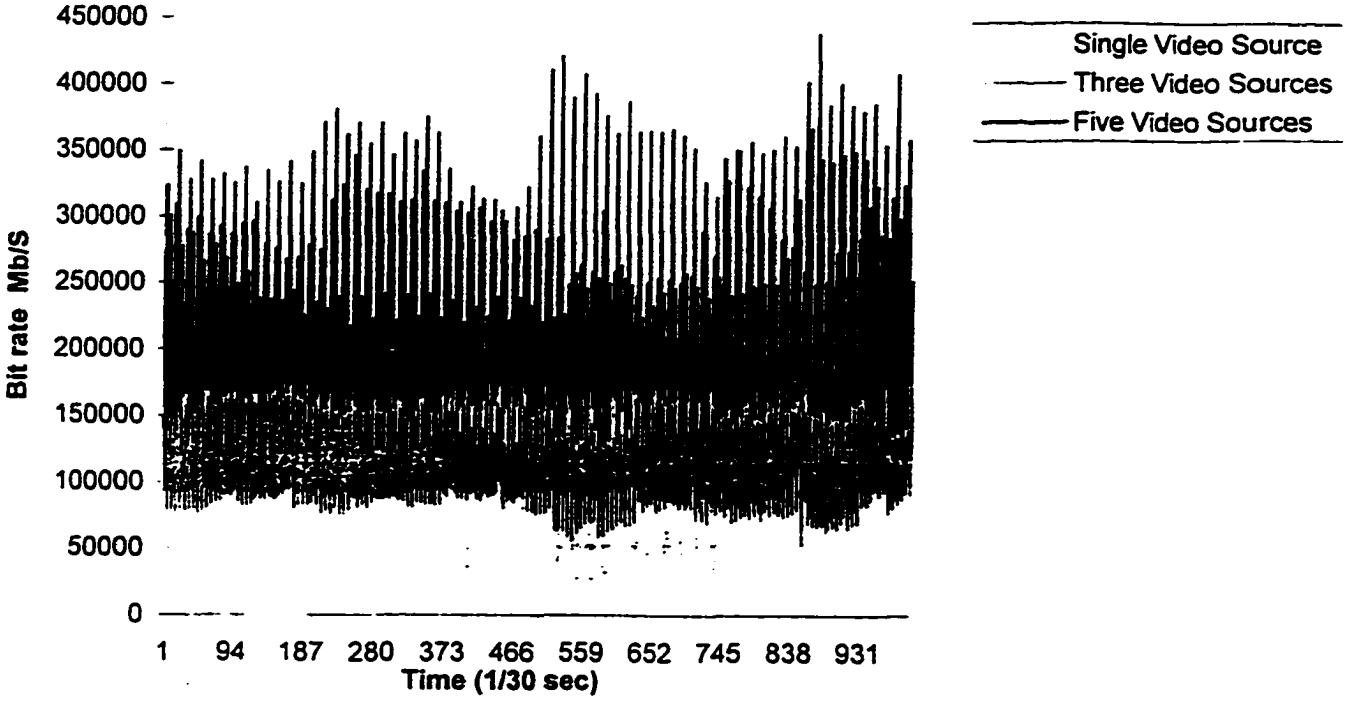


Figure 9: Bit rate for different number of video sources befor processing them

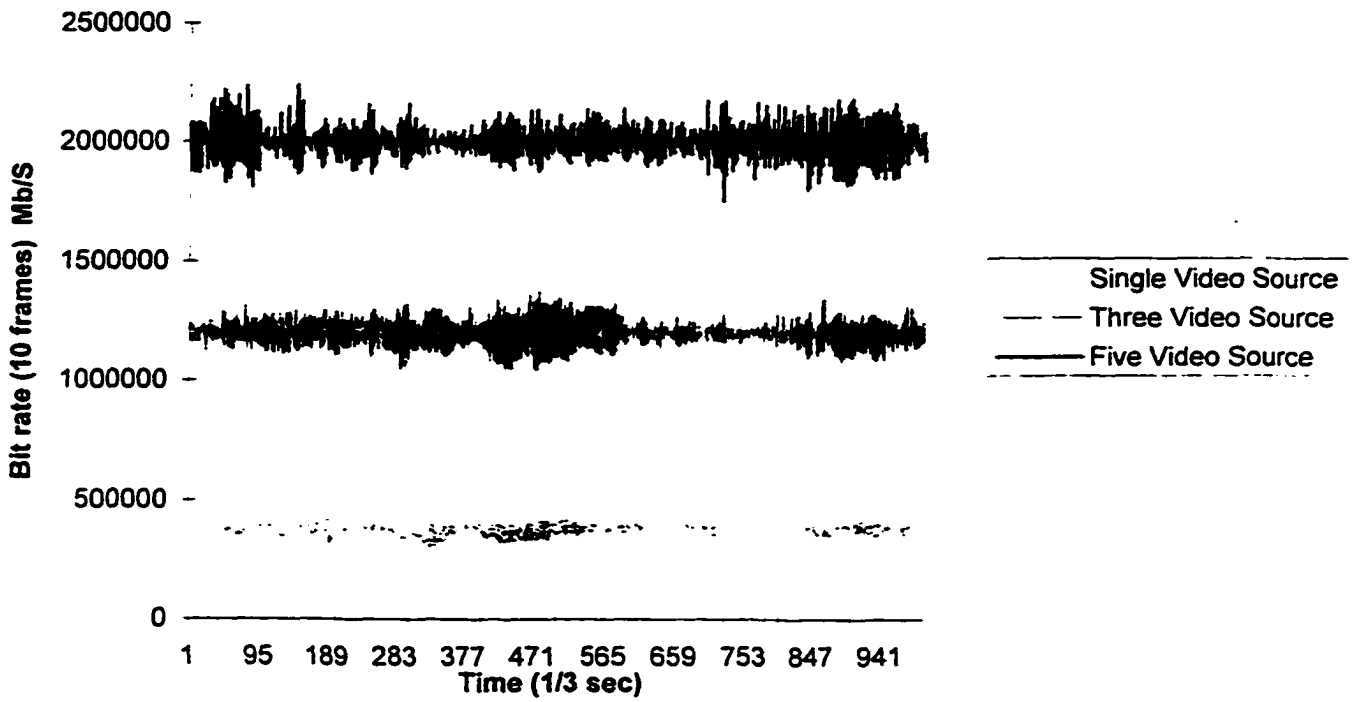


Figure 10: Bit rate for different number of video sources after processing them

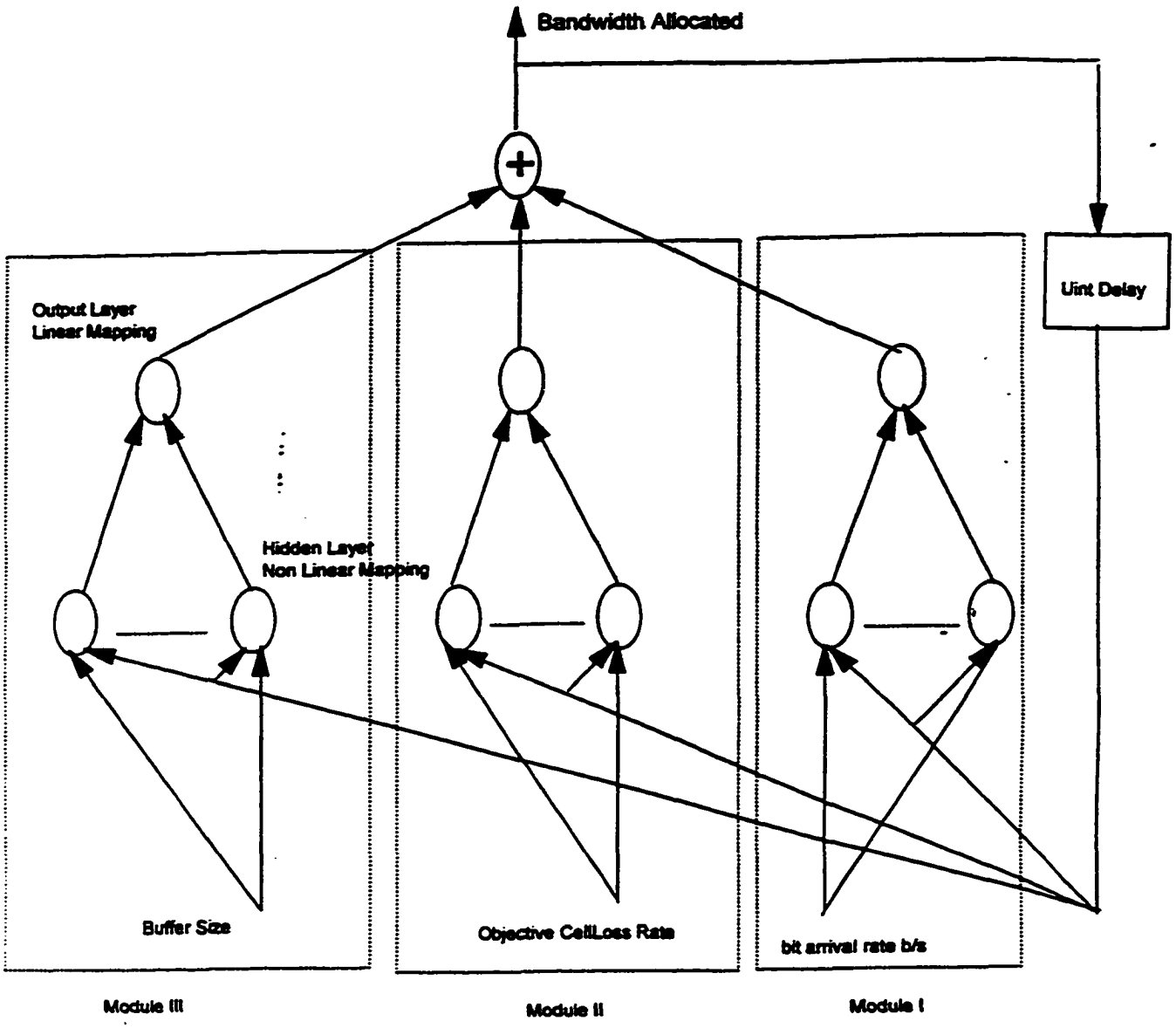


Figure 11: Radial basis function neural network model used to calulate the bandwidth

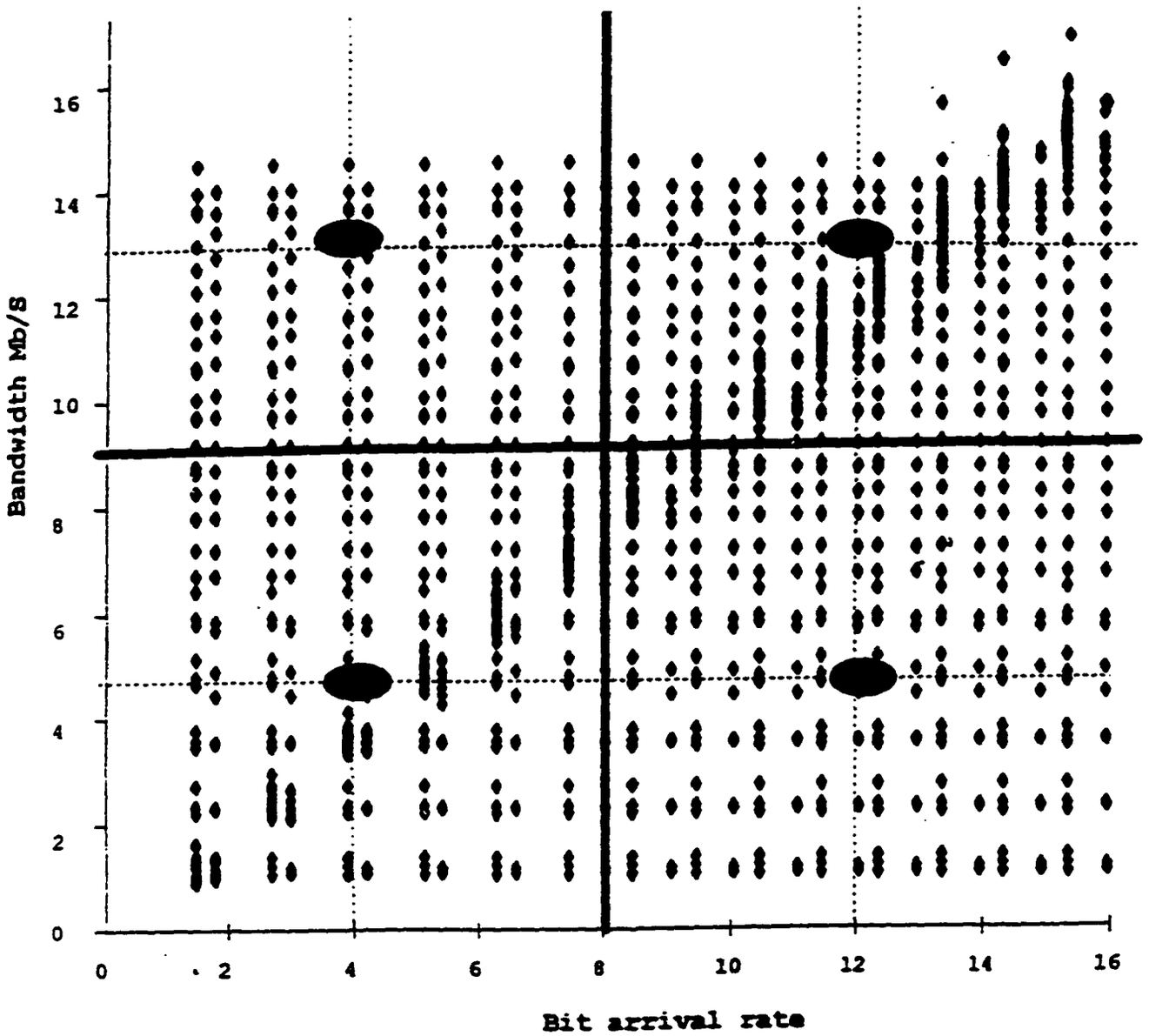


Figure 12: Locations of the 4 RBF neural network hidden neurons.

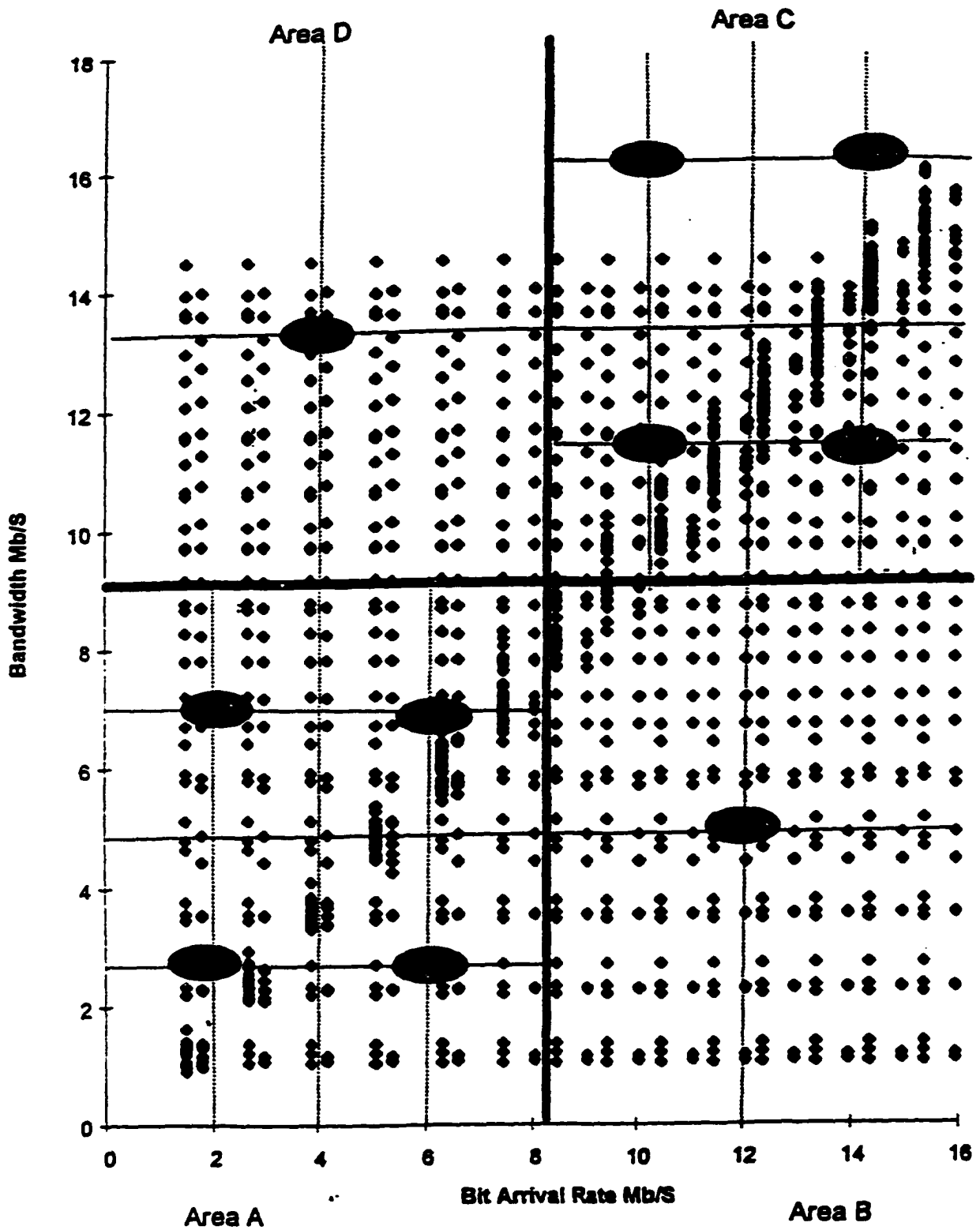


Figure 13: Locations of the 10 RBF neural network hidden neurons.

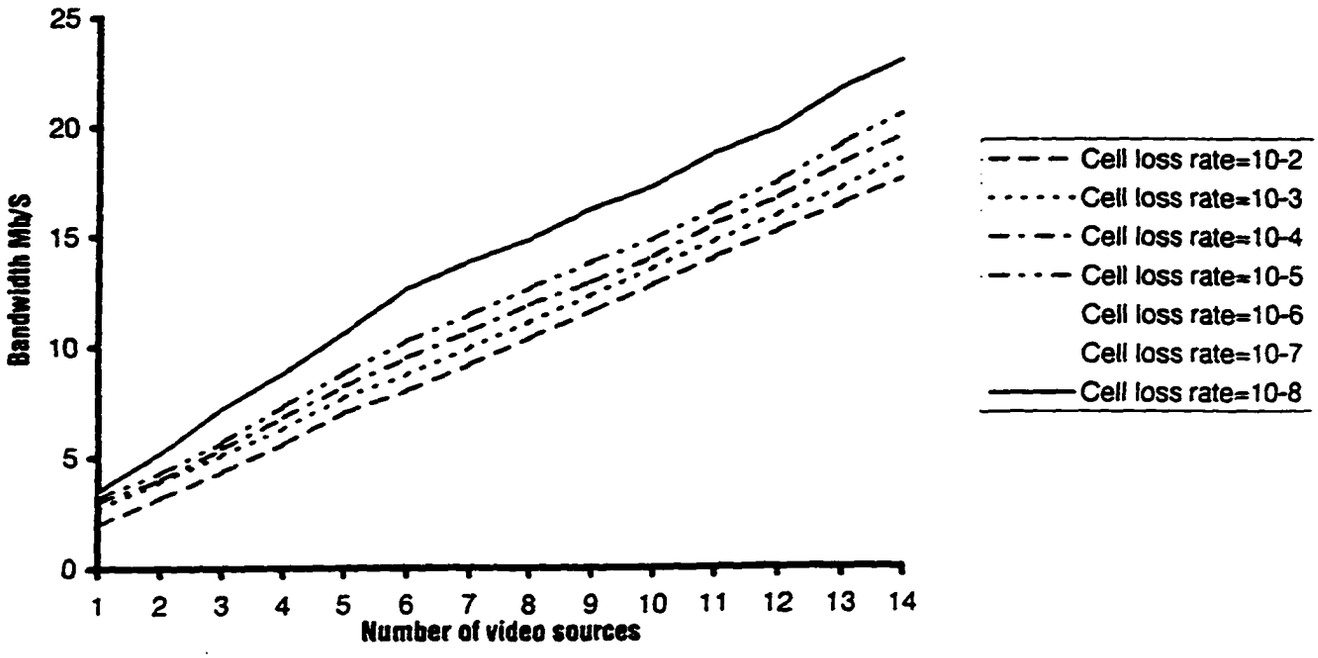


Figure 14: Bandwidth required for video traffic for different values of the cell loss rate

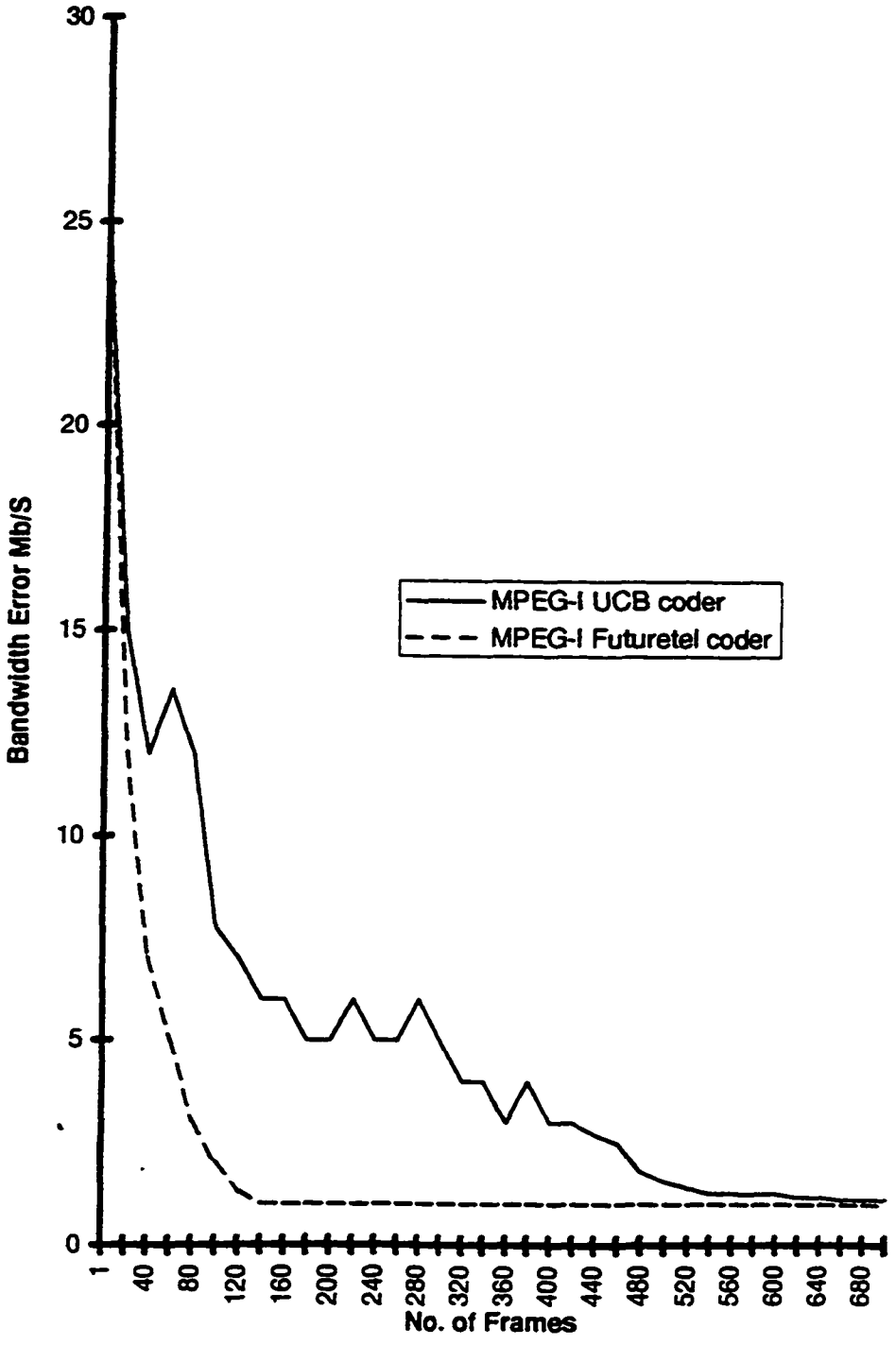


Figure 15: Bandwidth error verses the period length of on-line traffic measurement

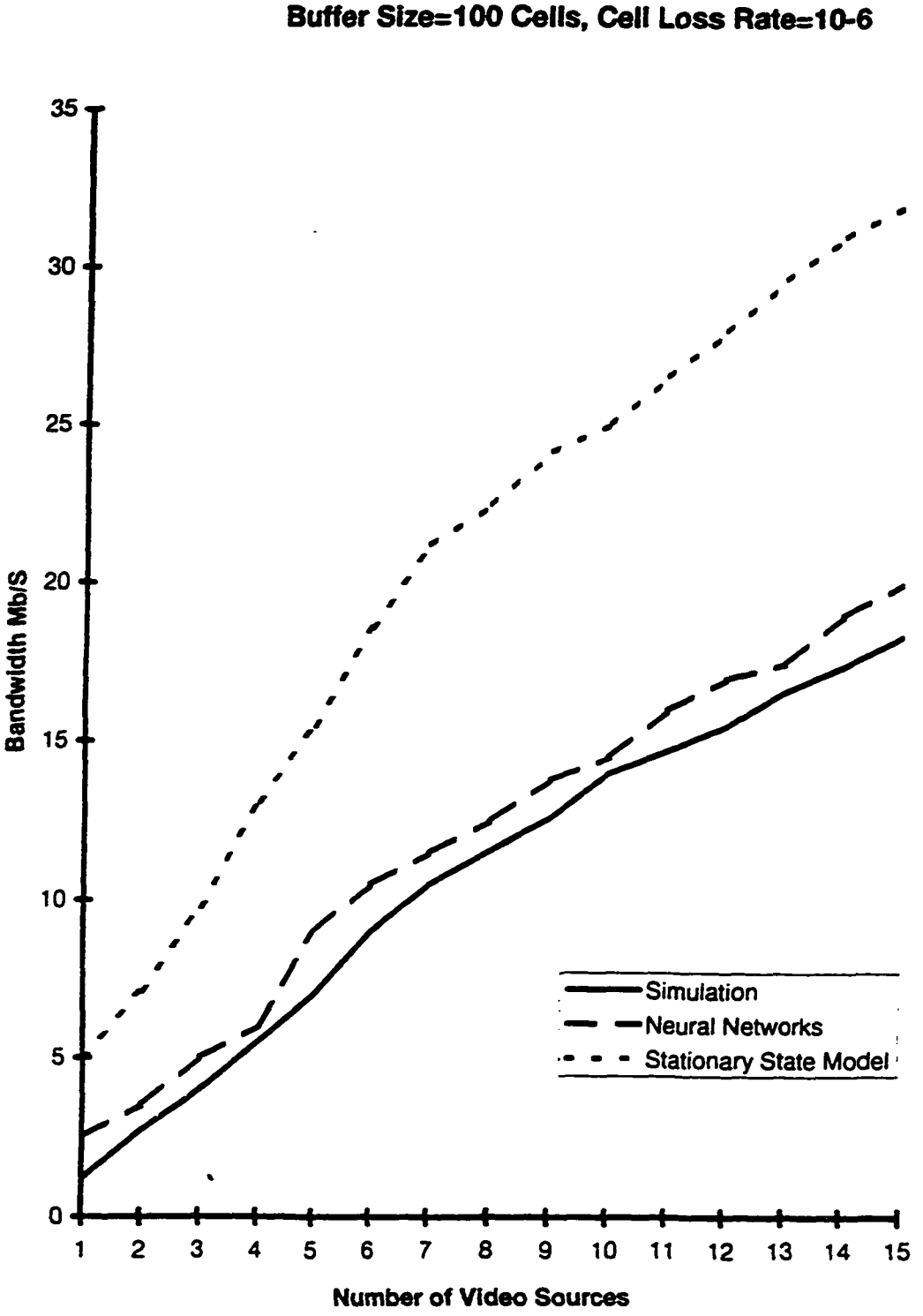


Figure 16: Bandwidth for video traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-6}$, buffer size= 100)

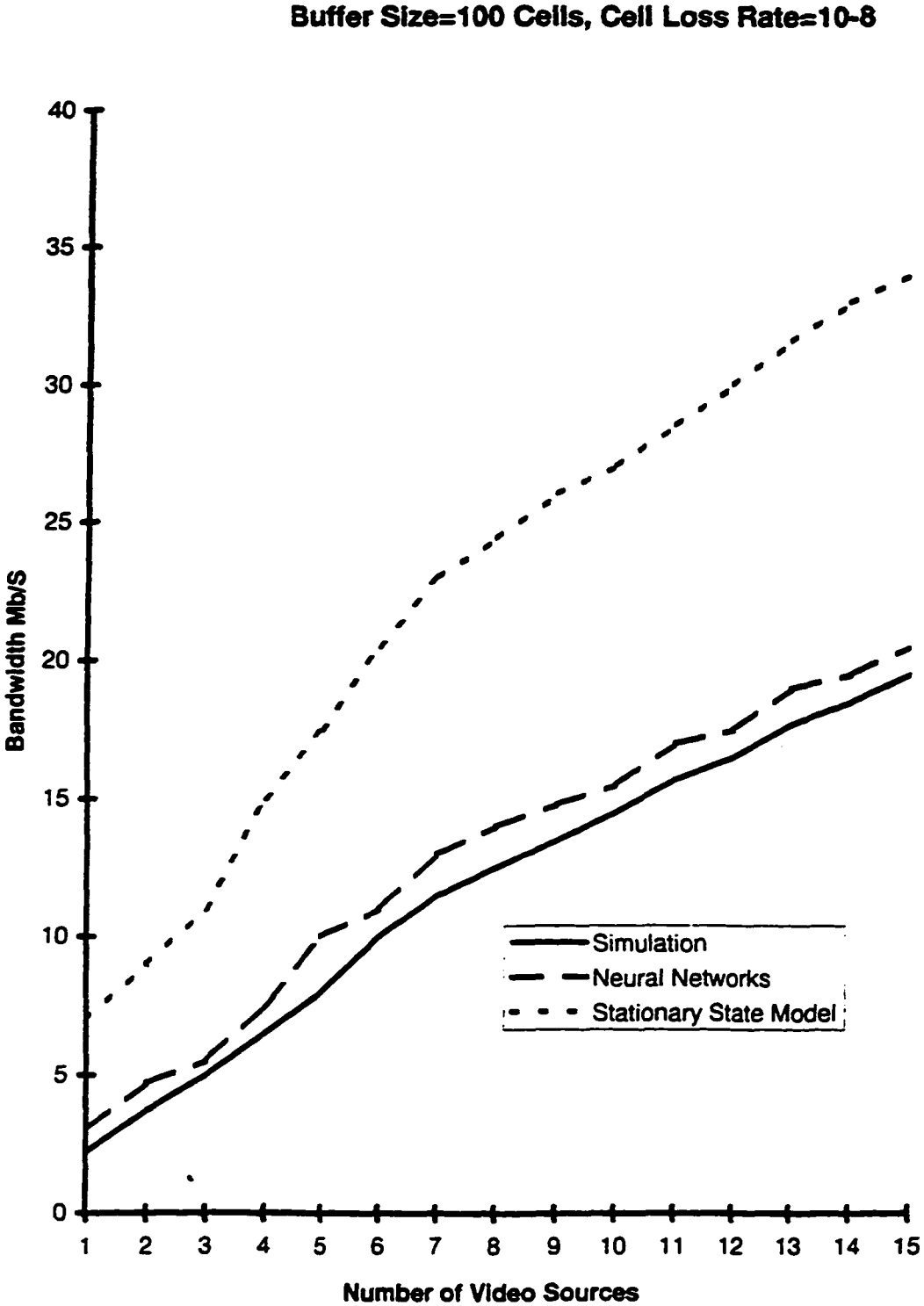


Figure 17: Bandwidth for video traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-8}$, buffer size= 100)

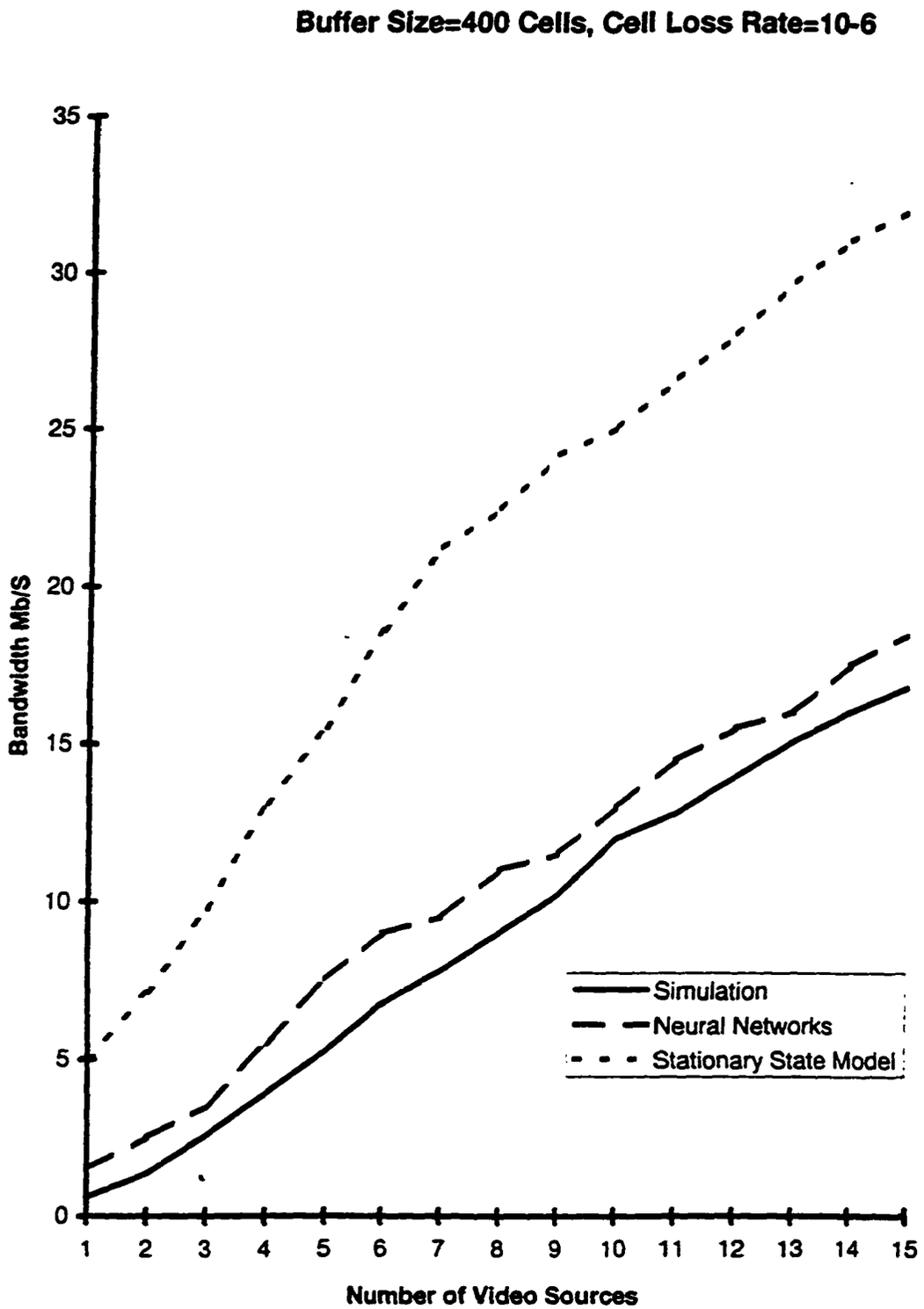


Figure 18: Bandwidth for video traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-6}$, buffer size= 400)

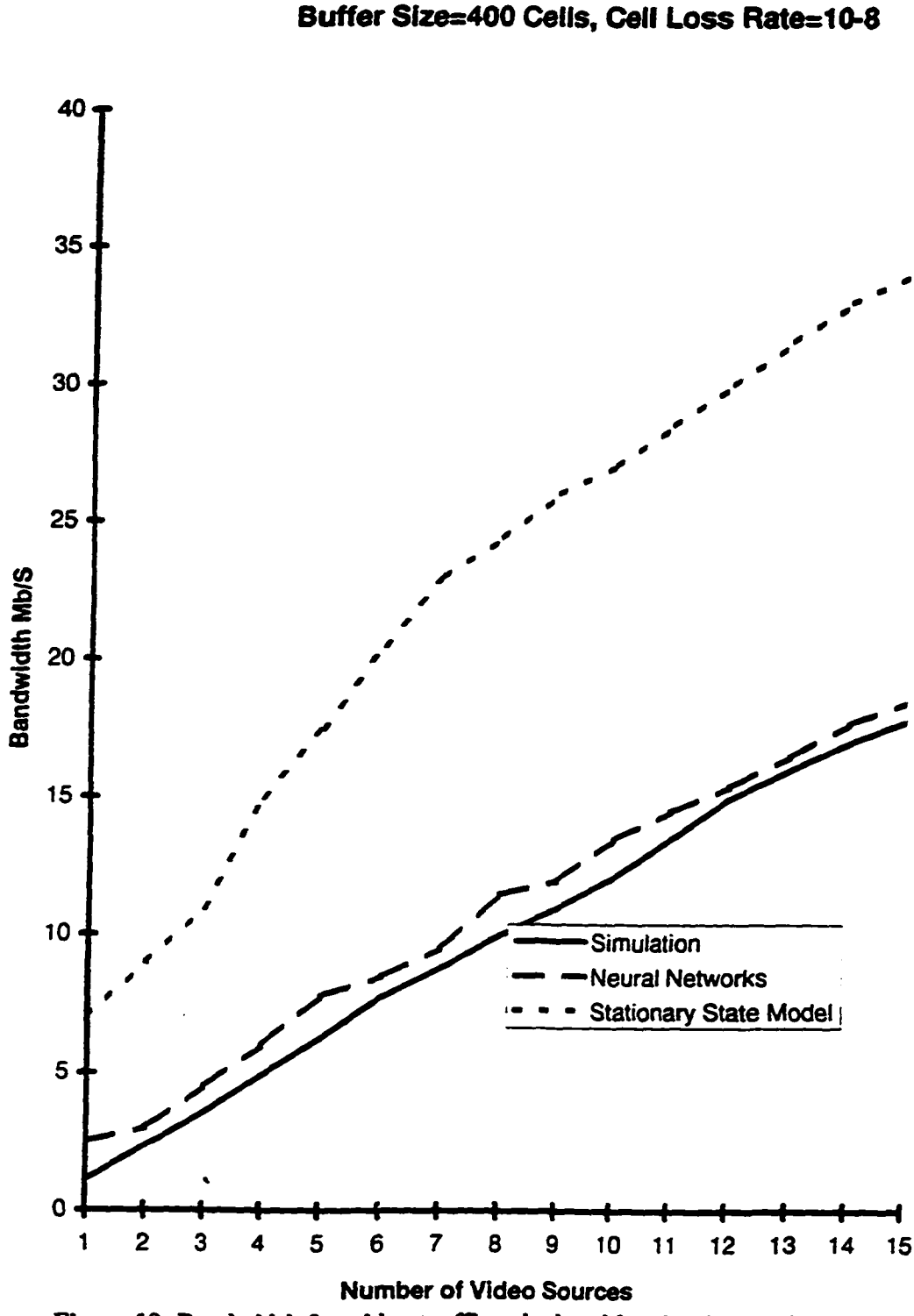


Figure 19: Bandwidth for video traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-8}$, buffer size= 400)

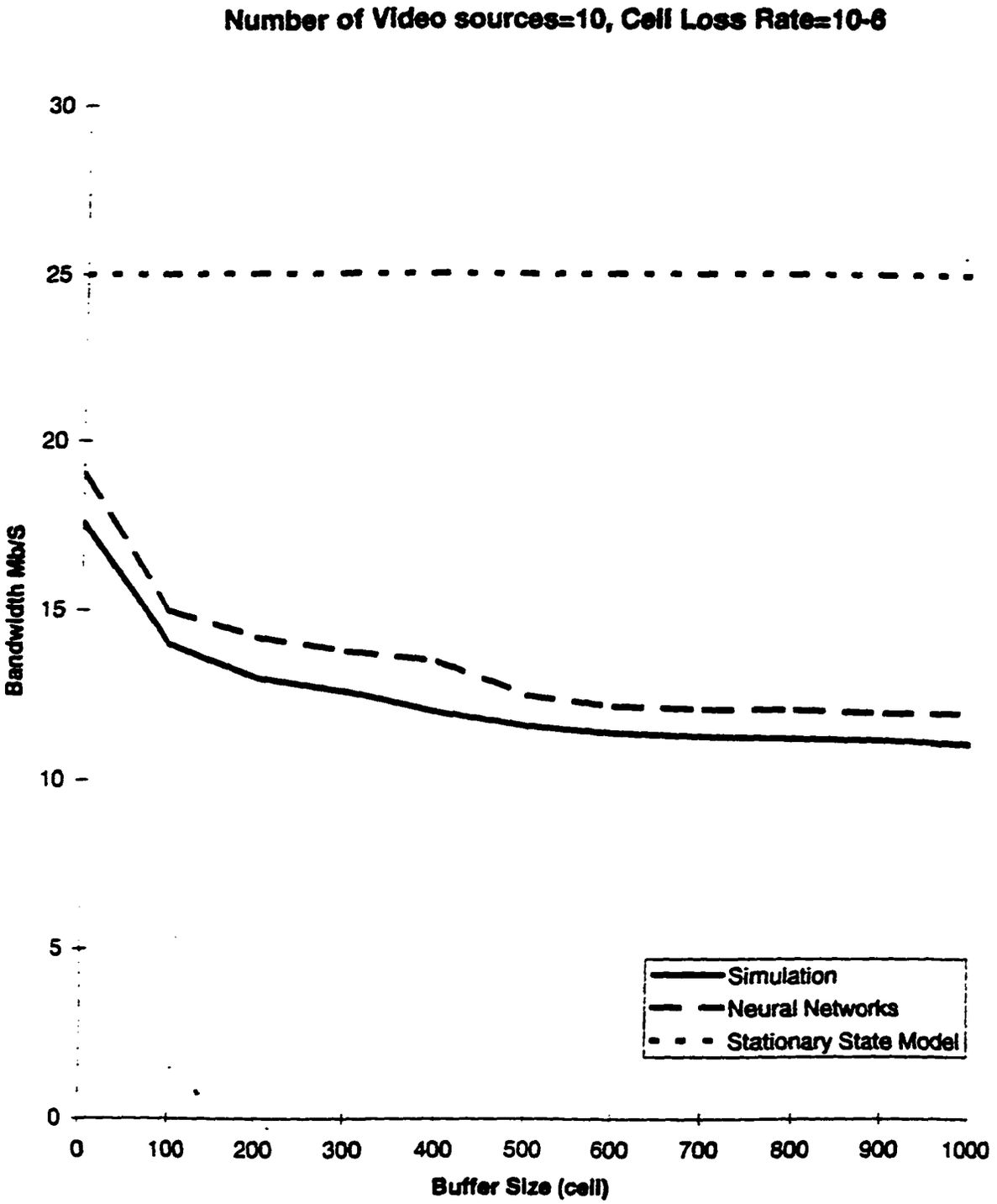


Figure 20: Bandwidth calculated from simulation, neuria networks, stationary state verses buffer size

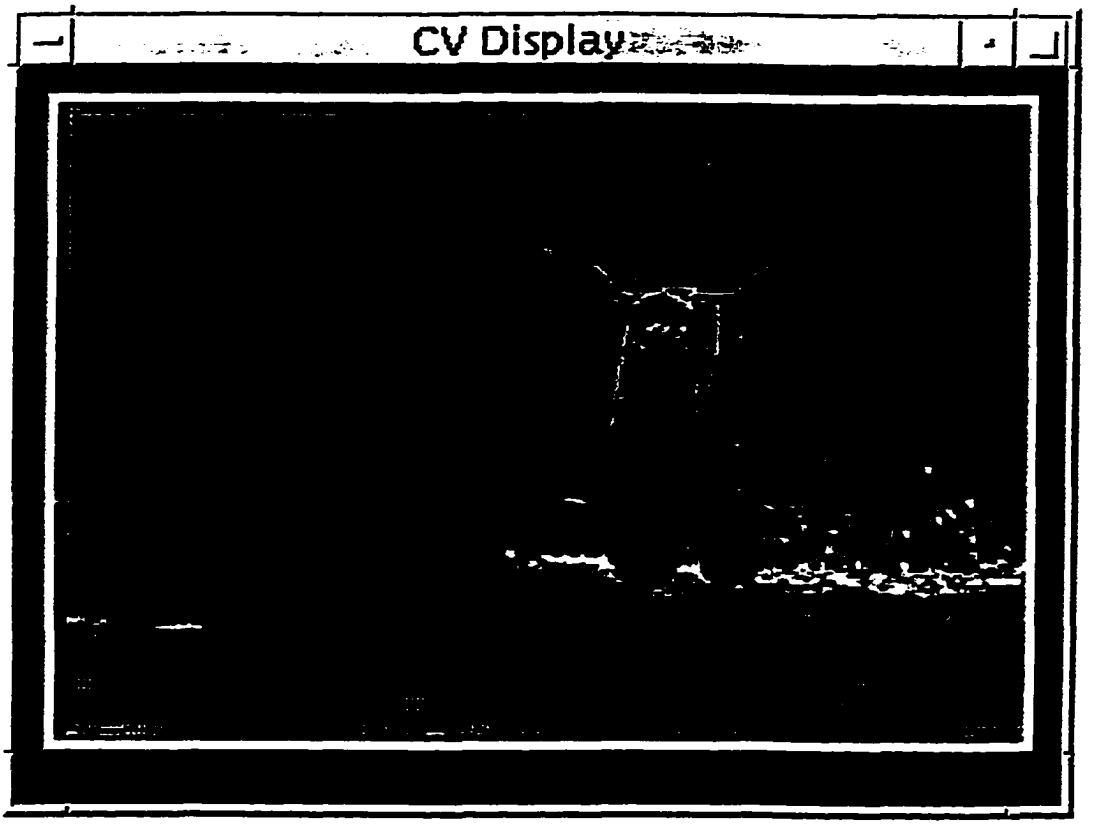


Figure 21: Video quality for cell los rate 10^{-4}

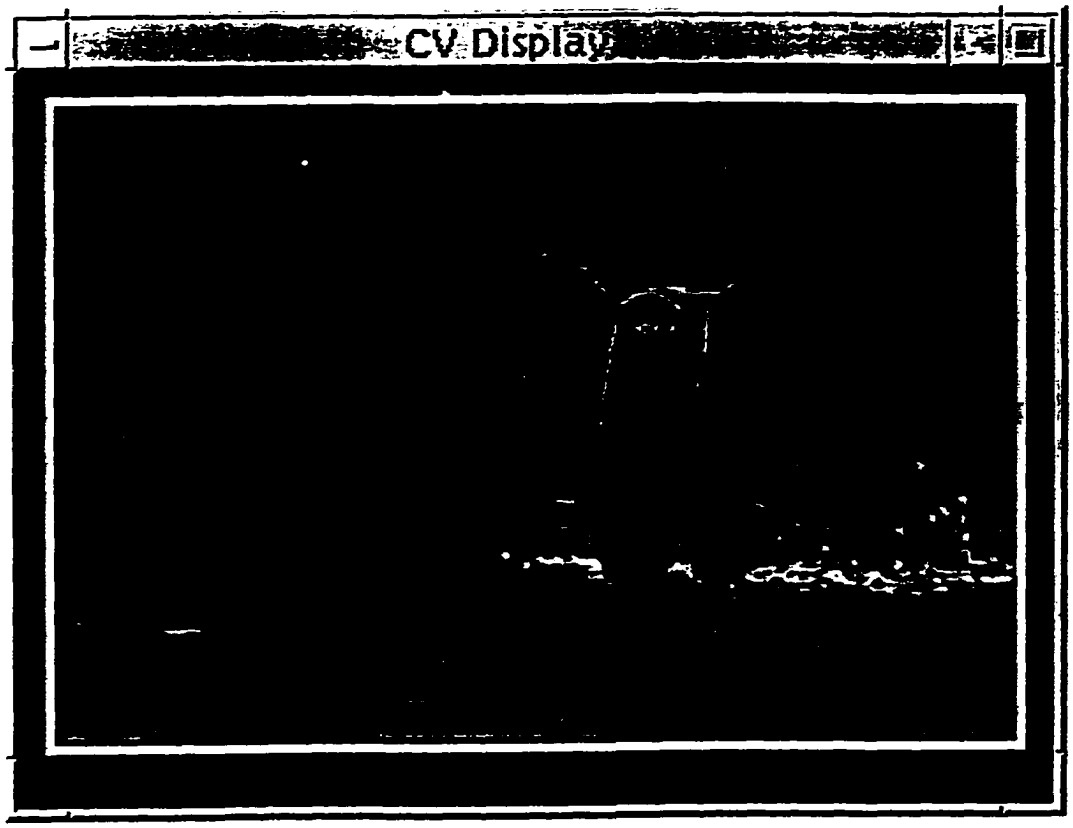


Figure 22: Video quality for cell los rate 10^{-6}

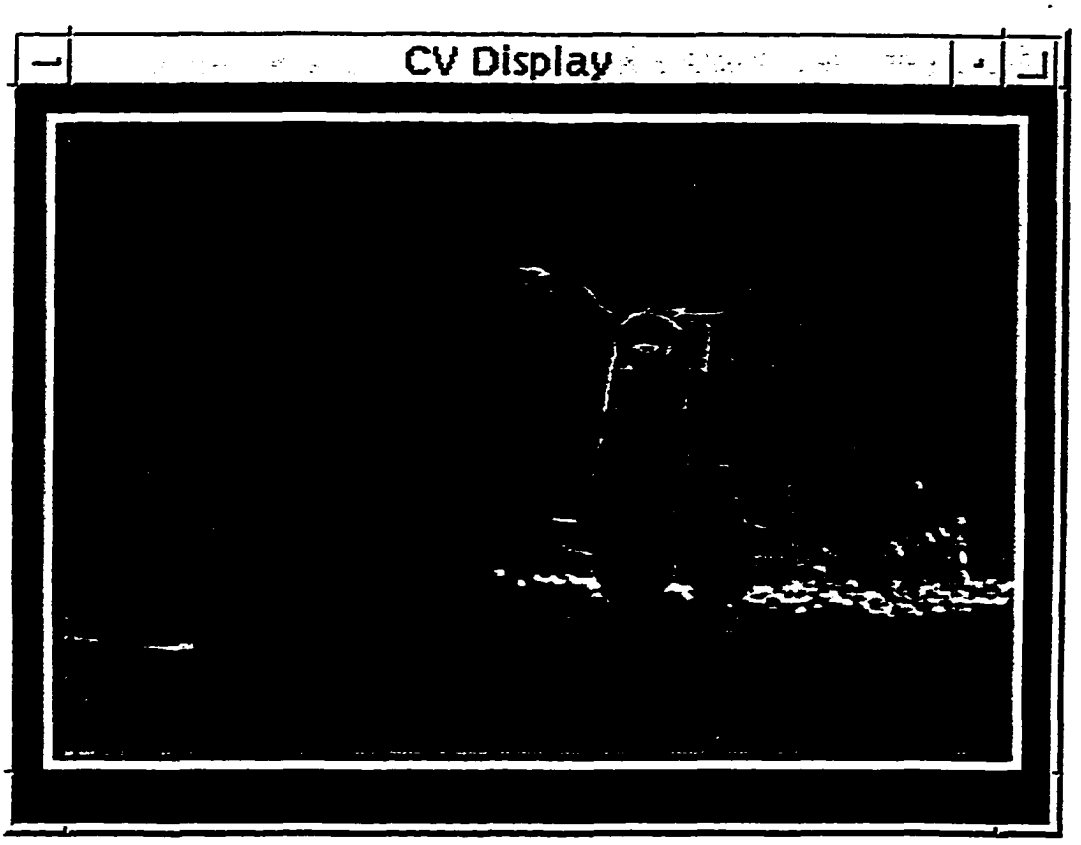


Figure 23: Video quality for cell los rate 10^{-8}

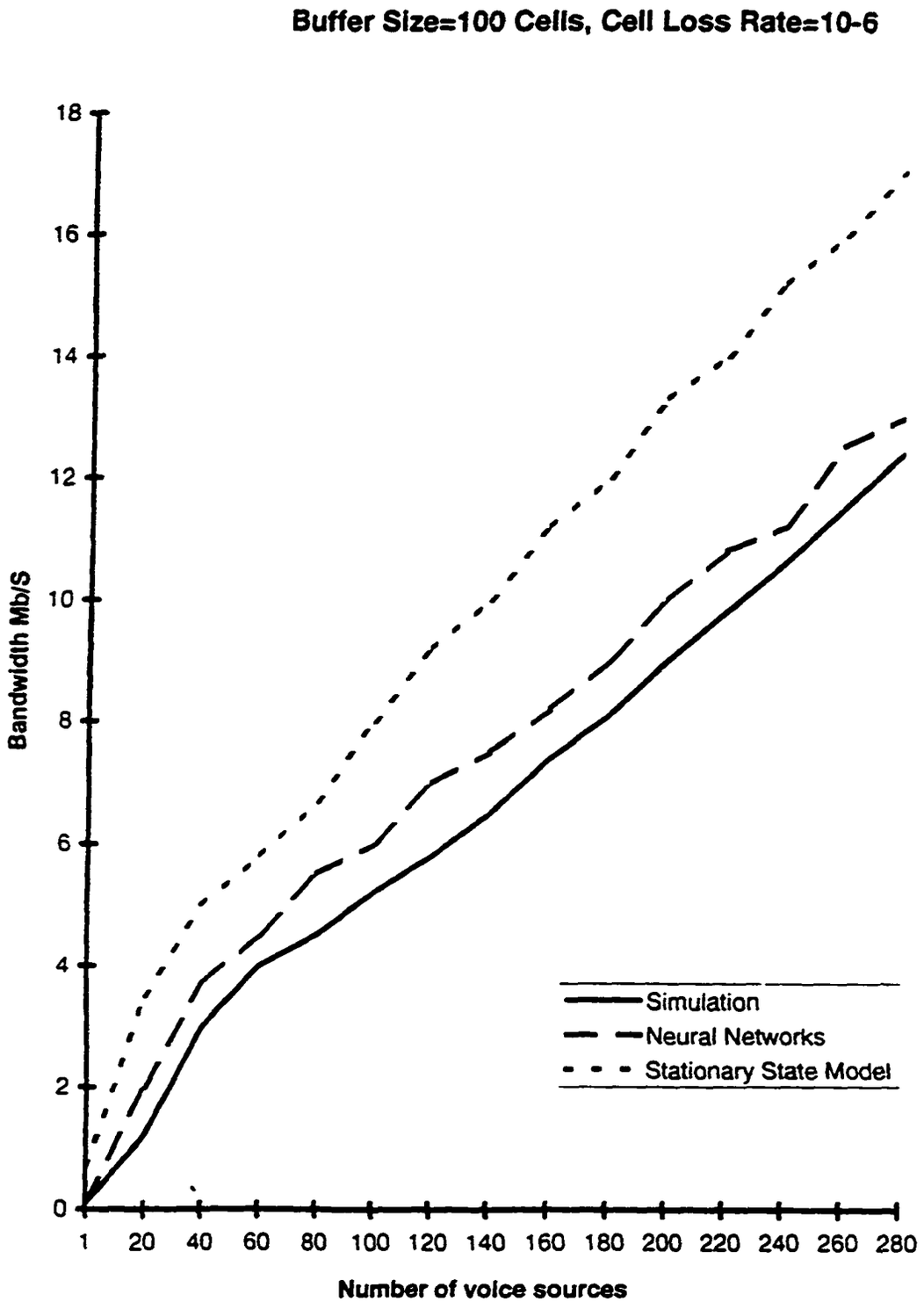


Figure 24: Bandwidth for voice traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-6}$, buffer size= 100)

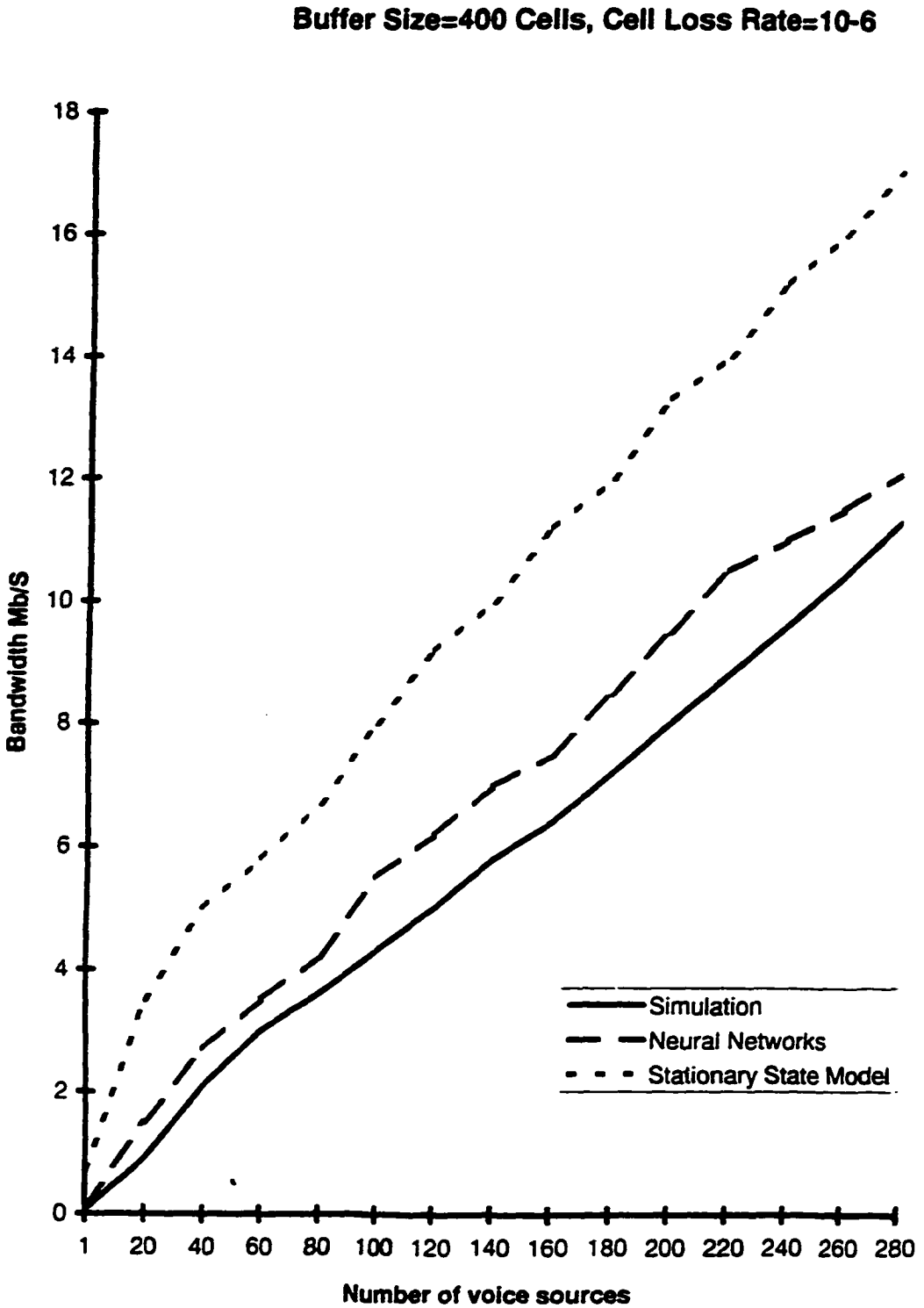


Figure 25: Bandwidth for voice traffic calculated by simulation, RBF neural networks, stationary state (cell loss rate $=10^{-6}$, buffer size= 400)

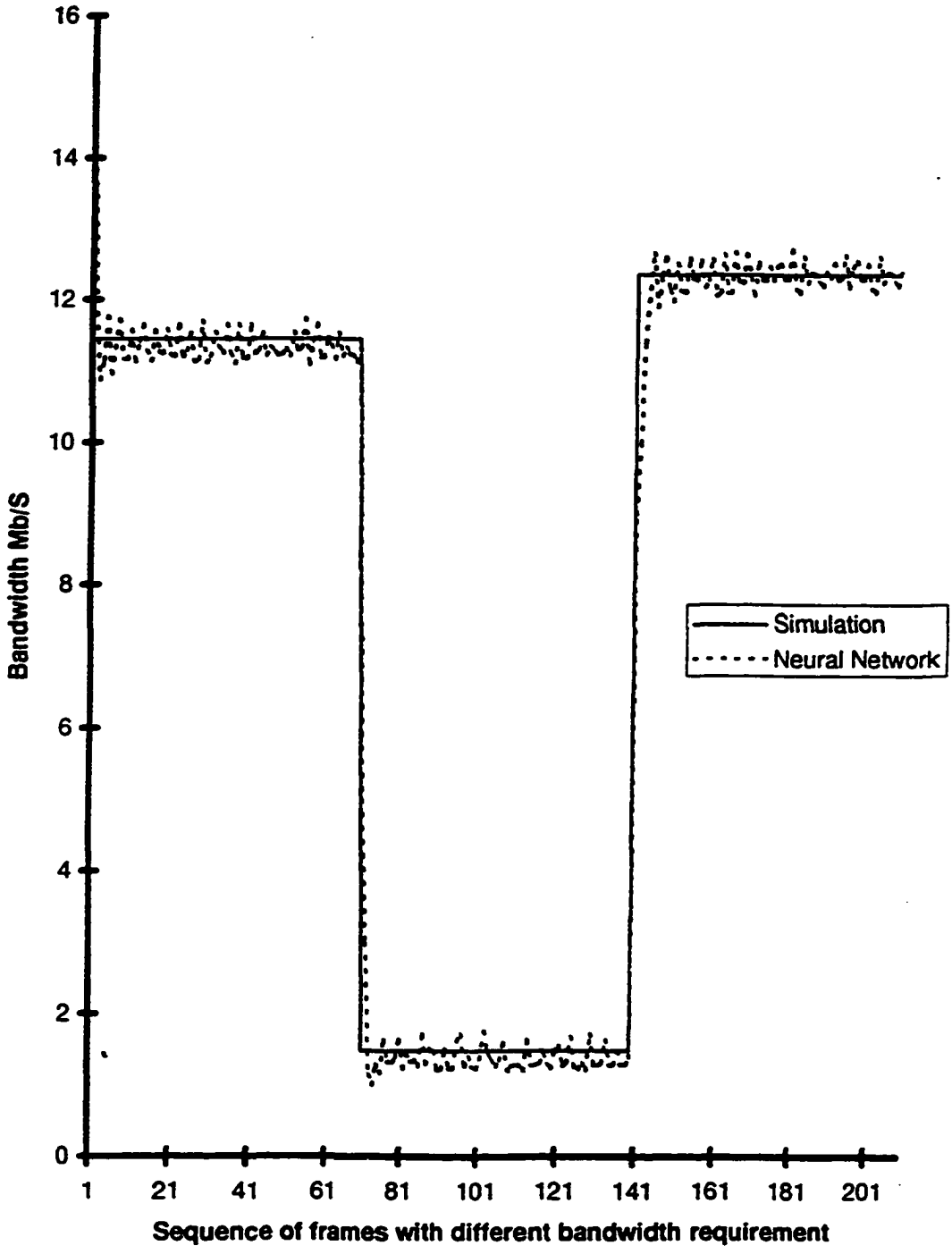


Figure 26: Convergence time for RBF neural network model

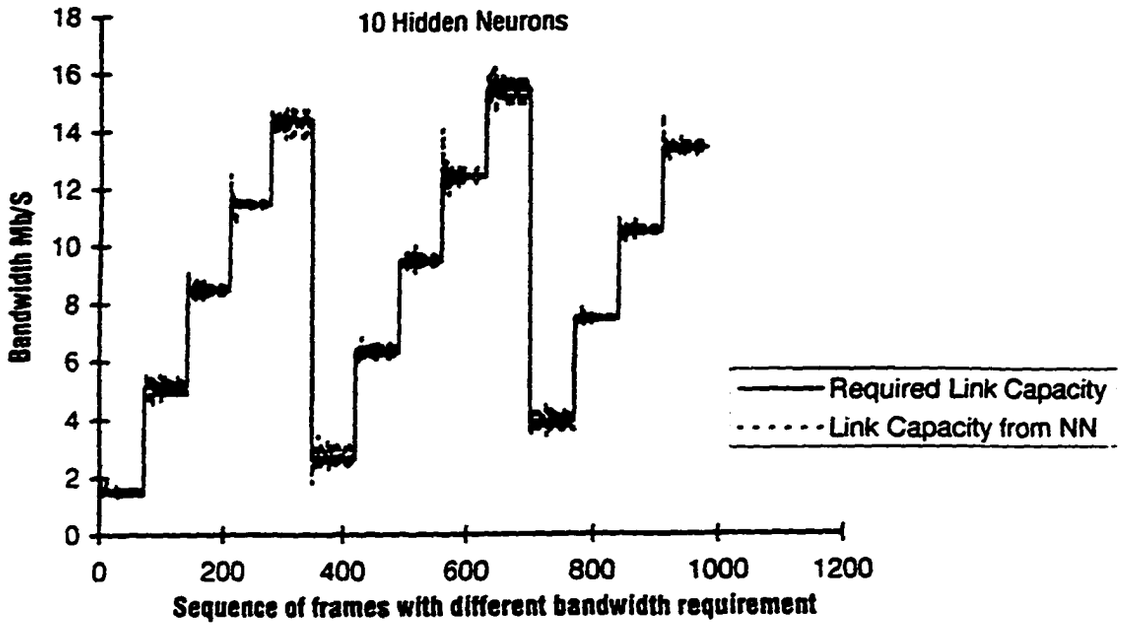


Figure 27: Calculating effective bandwidth from on-line measurements using RBF neural networks model using 10 hidden neurons

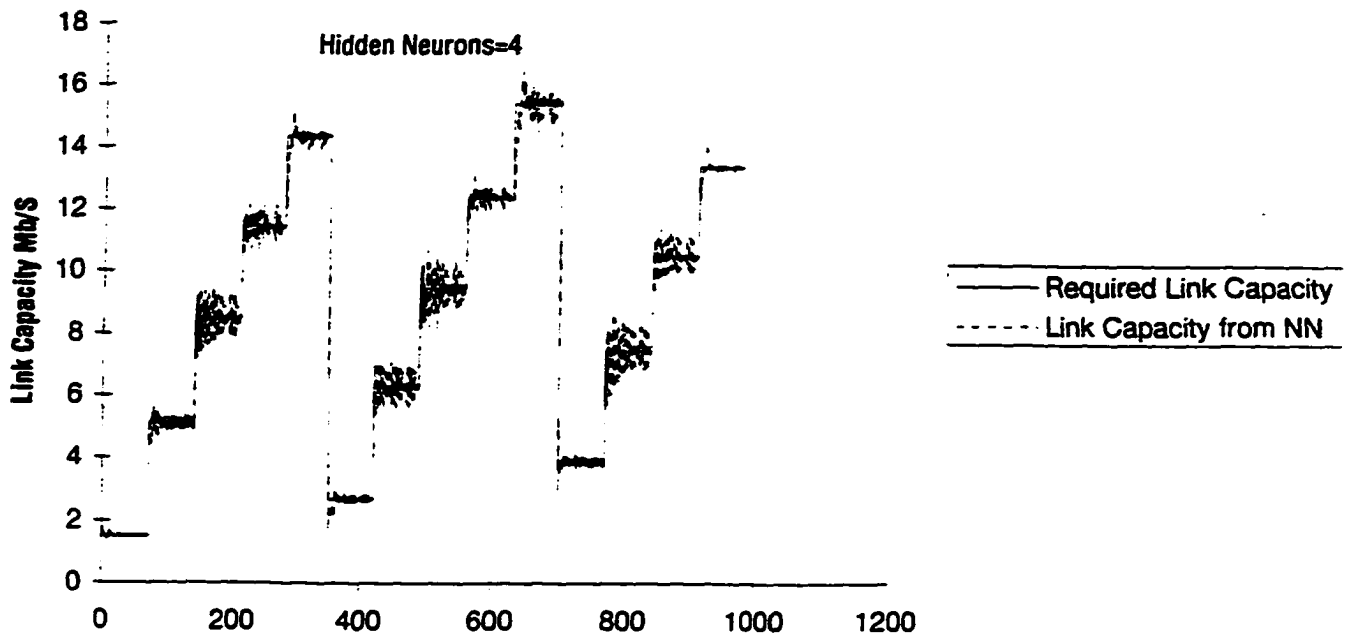


Fig. 28 Calculating effective bandwidth from on-line measurements using RBF neural networks model (4 hidden neurons)

References

[1] ITU Recommendation I.121, "Integrated Services Digital Network (ISDN), General Structure and Service Capabilities" , April 1991.

[2] Q. Ren, H. Kobayashi "Diffusion Approximation Modeling for Markov Modulated Bursty Traffic and Its Applications to Bandwidth Allocation in ATM Networks," IEEE J. Select. Areas Commun., Vol. 16, No. 5, pp. 679-691, June 1998.

[3] D. Mitra, M.I. Reiman, and J. Wang "Roboust Dynamic Admission Control for Unified Cell and Call QoS in Statistical Multiplexers," IEEE J. Select. Areas Commun., Vol. 16, No. 5, pp. 692-707, June 1998.

[4] I. Habib, T. Saadawi "Dynamic Bandwidth allocation and access control of virtual paths in ATM broadband networks," in Proc. 4th IFIP Conference on the High Performance Networks, December, 1992.

[5] M. Decina, T. Toniatti and P. Vaccari " Bandwidth Assignment and Virtual Call Blocking in ATM Networks," in Proc. INFOCOM'90, pp. 1015-1018.

[6] L.K. Reiss and L.F. Merakos, " A Capacity Allocation Rule for ATM Networks," in Proc. GLOBECOM'93, pp. 762-765.

[7] H. Saito, M. Kawarasaki and H. Yamada, " An Analyis of Statistical Multiplexing in an ATM Transport Network,"

IEEE J. Select. Areas Commun., Vol. 9, No. 3, pp. 812-817, pp. 861-869, April 1991.

[8] C. Shim, I, Ryoo, J. Lee and S. Lee, " Modeling and Call Admission Control Algorithm of Variable Bit Rate Video in ATM Networks," IEEE J. Select. Areas Commun., Vol. 12, No. 2, pp. 532-536, February 1994.

[9] R. Guerin, H. Ahmadi, and M.Naghshineh, "Equivalent Capacity and its application to Bandwidth Allocation in High Speed Networks," IEEE J. on Select. areas Commun., Vol. SAC-9, No. 7, pp. 969-981, December 1991.

[10] A.I. Elwalid and D. Mitra, "Effective Bandwidth of General Markovian Traffic Sources and Admission Control of High Speed Networks," IEEE/ACM Transactions on Networking, Vol. 1, No. 3, pp 329-343, June, 1993.

[11] G. Veciana, G. Kesidis, J. Warland, " Resource Management in Wide-Area ATM Networks Using Effective Bandwidths," IEEE J. Select. Area Commun., Vol. 13, pp. 911-923, No. 6, August 1995.

[12] C. Chang and J. Thomas, " Effective Bandwidth in High Speed Digital Networks," IEEE J. Select. Areas Commun., Vol. 13, No. 6, pp. 973-982, August 1995.

[13] G: Choudhury, D. Lucantoni and W. Whitt, " Squeezing the Most Out of ATM," IEEE Transactions on Communications, Vol. 44, No. 2, pp. 203-216,February 1996.

[14] K. Uehara and K. Hirota, "Fuzzy Connection Admission Control for ATM Networks Based on Possibility

Distribution of Cell Loss Ratio," IEEE J. Select. Areas Commun., Vol. 9, No. 7, pp. 542-554, September 1991.

[15] A. Hiramatsu, "Integration of ATM call admission control and link capacity control by distributerd neural networks," IEEE J. Select. Areas Commun., Vol. 9, No. 7, pp. 325-338, September 1991.

[16] A. Hiramatsu, "ATM Communications Network Control by Neural Networks," IEEE Transaction on Neural Networks, 1990.

[17] K. Siriram, and W. Whitt "Characterizing superposition arrival processes in packet multiplexers for voice and data," J. Select. Areas Commun., Vol. SAC-4. NO. 6, pp. 761-773, Sept 1986.

[18] R. Gusella "Characterizing the Variability of Arrival Processes with Indexes of Dispersion," J. Select. Areas Commun., Vol. 9, NO. 2, pp. 561-573, February 1991.

[19] W. Bernard "30 years of adaptive neural networks: Perceptron, Madaline, and Backpropagation," Proceedings of IEEE, Vol. 78, No. 9, pp. 1415-1442, Sept. 1990.

[20] K. Fukushima " A Neural Network for Visual Pattern Recognition," IEEE Computer, Vol. 21, NO. 3, pp. 65-75, March 1988.

[21] B. Maglaris, D. Anastassiou et al., "Performance models of statistical multiplexing in packet video communications", IEEE Trans. Comm., Vol 36, No. 7, pp. 834-844, July 1988.

[22] L.K. Reiss, L.F. Merakos, " A Capacity Allocation Rule for ATM Networks," in Proc. pp. 546-549 GLOBECOM'1993.

[23] HNC Inc. "HNC ExploreNET release 2.0 operating manual," 1991.

[24] S. Haykin, " Neural Networks: A Comprehensive Foundation," Macmillan College Publishing Company Inc., 1994.

[25] Mohamad H. Hassoun,"Fundamentals of Artificial Neural Netwroks," Massachusetts Institute of Technology, 1995.

[26] J.Y.Hui and E. Karasan "A Thermodynamic Theory of Broadband Networks with Application to Dynamic Routing," IEEE Journal on Selected Areas in Communication, Vol. 13, No.6, pp. 991- 1003, August 1995.

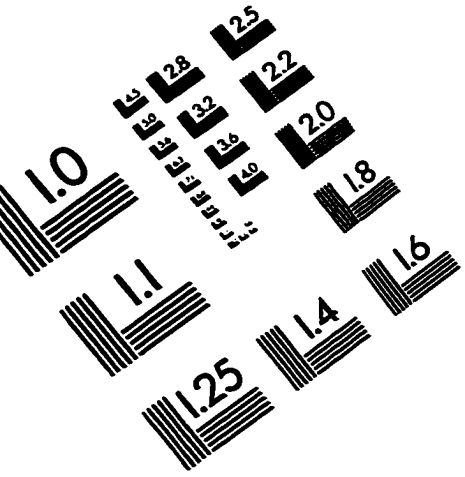
[27] M.J.D. Powell, "Radial Basis Functions for Mulivariable Interpolation: A Review, " Technical Report DAMPT 1985/NA12, Department of Applied Mathematics and Theoritical Physics, Cambridge University, England, 1985.

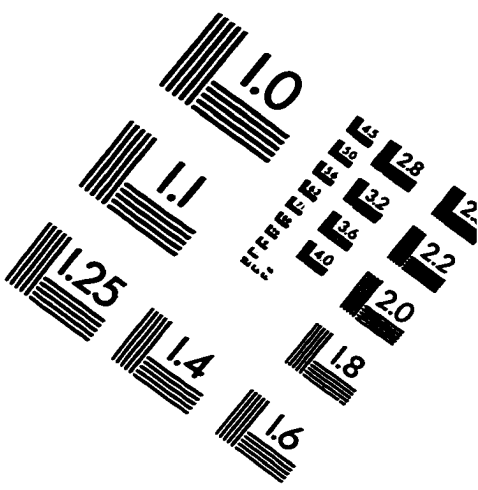
[28] J. Moody, C. Darken, "Fast learning in networks of locally-tuned processing units," Neural Computation,vol. 1, pp. 281-294, 1989.

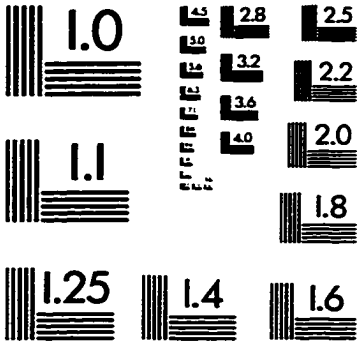
[29] D.S. Broomhead and D. Lowe "Mutivariate functional interpolation and adaptive vetwroks," Complex Systems, Vol. 2, pp. 321-355.

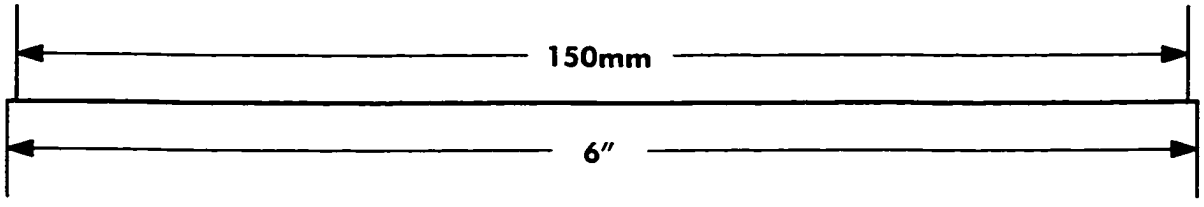
[30] S. Youssef, I. Habib, T. Saadawi "A Neurocomputing Controller for Bandwidth Allocation in ATM Networks," IEEE J. Select. Areas Commun., pp. 191-199, January, 1997.

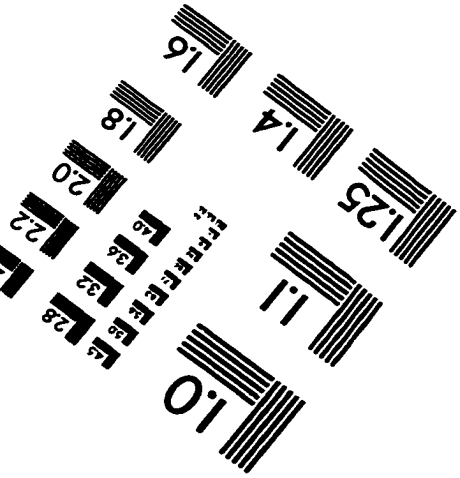
IMAGE EVALUATION TEST TARGET (QA-3)

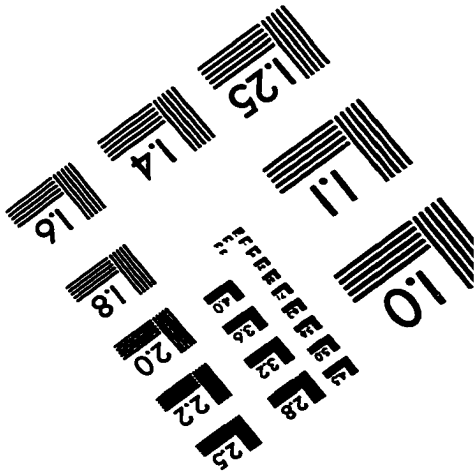












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989