

# Utilizing Neural Networks to Reduce Packet Loss in Self-Similar Teletraffic Patterns

Homayoun Yousefi'zadeh

Center for Pervasive Communications

Electrical Engineering and Computer Science Department

University of California, Irvine

hyousefi@uci.edu

Edmond A. Jonckheere

John A. Silvester

Department of Electrical Engineering - Systems

University of Southern California

[jonkhee,silvester]@usc.edu

**Abstract**— Reducing packet loss and increasing overall efficiency in multiple source queuing systems is one of the most important issues in the design of traffic control algorithms. On the contrary, the other important issue in such systems is to provide every individual source with the ability to take advantage of a fair portion of the shared available resources such as buffer space or server bandwidth. In this paper a novel technique for reducing packet loss in a class of queuing systems with self-similar traffic patterns is introduced. The technique takes advantage of the modeling power of neural networks to offer a dynamic buffer management scheme capable of efficiently addressing the trade off between packet loss and fairness issues.

**Index Terms**— Perceptron Neural Networks, Teletraffic Modeling, Self-Similarity, Buffer Management, Server Scheduling, Packet Loss, Dynamic Neural Sharing.

## I. INTRODUCTION

Analysis of traffic data from networks and services such as Ethernet LANs [14], Variable Bit Rate (VBR) video [3], ISDN traffic [10], and Common Channel Signaling Network (CCNS) [4] have all convincingly demonstrated the presence of features such as long range dependence, slowly decaying variances, and heavy-tailed distributions. These features are best described within the context of second-order self-similarity and fractal theory approach.

Neural networks are a class of nonlinear systems capable of learning and performing tasks accomplished by other systems. Their broad range of applications includes speech and signal processing, pattern recognition, and system modeling. Systems with neural network building blocks are robust in the sense that occurrence of small errors in the systems does not interfere with the proper operation of the system. This characteristic of neural networks, makes them quite suitable for traffic modeling. Considering this characteristic in [19], we utilized neural networks in modeling self-similar traffic patterns.

Reducing packet loss in queuing systems is one of the most important issues in the design of traffic control algorithms. Reducing packet loss in the queuing systems is equivalent to improving efficiency and is usually considered as a performance evaluation tool. For the systems consisting of more than one source, there is another major issue worth considering known as fairness. Fairness provides each individual source with the ability to take advantage of a fair portion of the shared available resources such as buffer space or server bandwidth. The combination of buffer management and scheduling algorithms specifies the fairness and the efficiency of a multiple source queuing system.

In this study, two different scheduling algorithms are considered. These are namely Fixed Time Division Multiplexing (FTDM) and Statistical Time Division Multiplexing (STDM). While in FTDM each source takes advantage of a fair portion of the server bandwidth also known as the service rate and there is no bandwidth sharing, in STDM the unused portion of the bandwidth assigned to each source might be used to service packets generated by other sources. While FTDM is typically used for ATM switching systems with a number of Virtual Paths, STDM is typically used in ATM queuing systems with a number of Virtual Channels.

There are a number of different buffer management algorithms studied in the literature as described in [15], [8], [11], [12], and [7]. These are namely Complete Sharing (CS) with no enforced capacity allocation mechanism, Complete Partitioning (CP) with equal partitioning of the available buffer capacity, and Partial Sharing (PS) with dedicated portions of the buffer space assigned to each source as well as a common shared portion. A dynamic buffer management algorithm is classified under PS methods with the ability to adjust the buffer size of each source dynamically. More specifically, a dynamic buffer management algorithm can address the trade off between fairness and efficiency by assigning a fair portion of the buffer space to each source with the ability to adjust the buffer space partitions according to system conditions.

The algorithm introduced in this paper is, in fact, a dynamic buffer management algorithm. It is capable of improving the loss performance of Static Partial Sharing (SPS) [15] while considering fairness versus loss trade off. The algorithm relies on the power of neural networks to model traffic patterns of individual sources in multiple source queuing systems and dynamically adjust the portion of the buffer space assigned to each source according to the corresponding traffic generation pattern. Relying on the prediction power of neural networks, the technique can outperform other threshold algorithms studied in the literature.

An outline of the paper follows. In Section II, we briefly review neural network modeling of teletraffic patterns. In Section III, we discuss the application of our modeling scheme in packet loss reduction of multiple source queuing systems. We also compare the performance of our proposed Dynamic Neural Sharing (DNS) scheme with other buffer management schemes. Finally, we conclude the paper in Section IV.

## II. SELF-SIMILAR TRAFFIC MODELING

### A. Self-Similar Traffic

In [19], we provide an analytical framework for self-similarity as a statistical property of the time series. Here, we provide a brief summary of that discussion. Suppose  $X = (X_t : t = 0, 1, 2, \dots)$  is a covariance stationary stochastic process with mean  $\mu$ , variance  $\sigma^2$ , and autocorrelation function  $R(n)$ ,  $n \geq 0$ . Particularly, assume the autocorrelation function of  $X$  has the form

$$R(n) \sim k_1 n^{-\beta}, \quad \text{as } n \rightarrow \infty \quad (1)$$

where  $0 < \beta < 1$  and constants  $k_1, k_2, \dots$  are finite positive integers. For each  $m = 1, 2, 3, \dots$  let  $X^{(m)} = (X_n^{(m)} : n = 1, 2, 3, \dots)$  be the covariance stationary time series with corresponding autocorrelation function  $R^{(m)}$  obtained from averaging the original series  $X$  over the non-overlapping time periods of size  $m$ , i.e., for each  $m = 1, 2, 3, \dots$  the moving average  $X^{(m)}$  is given by

$$X_n^{(m)} = \frac{1}{m}(X_{nm-m+1} + \dots + X_{nm}), \quad n \geq 1 \quad (2)$$

The process  $X$  is called exactly second-order self-similar with the self-similarity parameter  $H = 1 - \beta/2$  if the corresponding  $X^{(m)}$  has the same correlation function as  $X$ , i.e.,  $R^{(m)}(n) = R(n)$  for all  $m = 1, 2, 3, \dots$  and  $n = 1, 2, 3, \dots$ .  $X$  is called asymptotically second-order self-similar with self-similarity parameter  $H = 1 - \beta/2$  if  $R^{(m)}(n)$  asymptotically approaches to  $R(n)$  given by (1), for large  $m$  and  $n$ . Hence, if the correlation functions of the aggregated processes  $X^{(m)}$  are the same as the correlation functions of  $X$  or approach asymptotically to the correlation functions of  $X$ , then  $X$  is called exactly or asymptotically second-order self-similar. Fractal Gaussian Noise (FGN) is a good example of an exactly self-similar process with self-similarity parameter  $H$ ,  $1/2 < H < 1$ . Fractional Arima processes with the parameters  $(p, d, q)$  such that  $0 < d < 1/2$  are examples of asymptotically second-order self-similar processes with self-similarity parameter  $d + 1/2$ .

Mathematically, self-similarity manifests itself in a number of ways.

- The variance of sample mean decreases more slowly than the reciprocal of the sample size. This is called slowly decaying variance property with the meaning  $\text{var}(X^{(m)}) \sim k_2 m^{(-\beta)}$  as  $m \rightarrow \infty$  with  $0 < \beta < 1$ .
- The autocorrelations decay hyperbolically rather than exponentially fast, implying a non-summable autocorrelation function  $\sum_n R(n) = \infty$ . This is called long range dependence property.
- The spectral density  $f(\cdot)$  obeys a power-law near the origin. This is the concept of  $1/f$  noise with the meaning  $f(\lambda) = k_3 \lambda^{-\gamma}$  as  $\lambda \rightarrow \infty$  for  $0 < \gamma < 1$  and  $\gamma = 1 - \beta$ .

The most important feature of self-similar processes is seemingly the fact that their aggregated processes  $X^{(m)}$  possess a non-degenerate correlation function as  $m \rightarrow \infty$ . This is completely different from typical packet traffic models previously considered in the literature, all of which have the property that their aggregated processes  $X^{(m)}$  tend to second order pure noise, i.e.,  $R^{(m)} \rightarrow 0$  as  $m \rightarrow \infty$ .

### B. Neural Network Modeling of Self-Similar Traffic

In [19], we utilize the proposed methods of [16], [6], and [17] to describe how a fixed structure feed forward perceptron neural network with back propagation learning algorithm can be used to model aggregated self-similar traffic patterns as an alternative to stochastic and chaotic systems approaches proposed in [13], [5], [2], and [1]. We note that although the emphasis of our work is on self-similar traffic modeling, our proposed neural network modeling approach can nevertheless be used for any traffic pattern independent of self-similarity. In what follows we briefly review the neural network modeling technique of [19] in which an elegant approach capable of coping with the fractal properties of the aggregated traffic is introduced. The approach provides an attractive solution for traffic modeling and has the advantage of simplicity compared to the previously proposed approaches namely stochastic and deterministic chaotic map modeling. The promise of neural network modeling approach is to replace the analytical difficulties encountered in the other modeling approaches with a straightforward computational algorithm. As oppose to the other modeling approaches, neural network modeling does not investigate identification of appropriate maps neither does it introduce a parameter describing the fractal nature of traffic. It, hence, need not cope with the complexity of estimating Hurst parameter and/or fractal dimensions. The proposed neural networking approach of this article simply takes advantage of using a fixed structure nonlinear system with a well defined analytical model that is able to predict a traffic pattern after learning the dynamics of the pattern through the use of information available in a number of traffic samples. Interestingly and as proposed by Gomes et al. [9], neural networks can also be utilized as appropriate estimators of the Hurst parameter.

The fixed structure, fully connected, feed forward perceptron neural network utilized for the task of modeling in our study consists of an input layer with eight neurons, three hidden layers with twenty neurons in each layer, and an output layer with one neuron. The number of neurons in each layer reflect our best practical findings leading to a balance between complexity and accuracy. Fig. 1 illustrates the structure of the neural network. The sigmoid transfer function defined below

$$f(z) = (1 + e^{-z})^{-1} \quad (3)$$

is utilized to generate the output of each neuron from its combined input. The output of each neuron is connected to the input of all of the neurons in the layer above after being multiplied by a weighting function. The specific neural network used for the task of modeling relies on the so-called back propagation learning algorithm described in [19], [16] and the references therein. In a nutshell, the back propagation learning algorithm (BPA) overcomes the mismatch between the actual and the generated outputs by adjusting the weightings of interconnections denoted by  $\Delta w_{ji}[\cdot]$  in the opposite direction of the gradient vector and its momentums in order. BPA minimizes the absolute error function  $E$  defined proportional to the square of the difference between the neural network output and the real output

as

$$\underbrace{\Delta w_{ji}[s]}_{(k+1)\text{-th step}} = lc \cdot e_j[s] \cdot \{x_i[s-1] + k \cdot e_i[s-1]\} + M(\underbrace{\Delta w_{ji}[s]}_{k\text{-th step}}) \quad (4)$$

In Equation (4)  $M$ ,  $w_{ji}[s]$ ,  $lc$ ,  $E$ ,  $e_j[s]$ , and  $x_i[s-1]$  denote the momentum, the weighting function of the connection between the  $i$ -th neuron in layer  $(s-1)$  and the  $j$ -th neuron in layer  $s$ , the learning coefficient, the absolute error function, the relative error function of the  $j$ -th neuron in layer  $(s)$ , and the present output state of the  $i$ -th neuron in layer  $(s-1)$  respectively. In the adjustment process, BPA propagates the output layer error to the preceding layer via the existing connections and repeats the operation until reaching the input layer. In other words, output error moves from the output layer -just in the opposite direction of the movement of original information- one layer at a time until reaching the input layer.

In a typical iteration of the learning phase, the neural network is provided with samples  $x[k-8]$  through  $x[k-1]$  of the real traffic pattern. The difference between sample  $x[k]$  of the real traffic pattern and the neural network output is then used to adjust the weighting functions of the network accordingly. In the next iteration, sample  $x[k-8]$  of the real traffic pattern is discarded, samples  $x[k-7]$  through  $x[k]$  of the real traffic pattern are used as the new input sample set, and sample  $x[k+1]$  is used as the new real traffic sample. The neural network continues processing more information in consecutive iterations of the learning phase until the absolute error is less than a specified error bound,  $\epsilon$ . The learning phase of the perceptron neural network is directly followed by the recalling phase when the network output is able to follow the real traffic within the acceptable error bound,  $\epsilon$ . In each iteration of the recalling phase, the neural network independently generates the samples by discarding the oldest input sample, shifting the input samples by one, and using its output as the most recent input sample. It is important to note that while the training and recalling phases of our modeling scheme rely on standard feedforward and recurrent feedback methods respectively, other combinations of training and recalling phases such as utilization of pure recurrent schemes are also possible. The same sequence of following a learning phase by a recalling phase is repeated when and if the neural network output difference exceeds the acceptable error bound,  $\epsilon$ . The number of samples required for the training of the neural network depends on the complexity of the traffic pattern dynamics. The time complexity and the space complexity of the back propagation algorithm are respectively  $\mathcal{O}(IN)$  and  $\mathcal{O}(N)$  where  $N$  is the number of weighting functions in the network and  $I$  is the number of iterations. Although the complexity is typically better than the complexity of implementing statistical approaches such as fractional ARIMA processes or the complexity of calculating fractal dimensions such as correlation dimension, wide variations of  $I$  prevent us from making a strong claim about complexity advantage of the algorithm compare to other algorithms. Nonetheless combining the straight forward way of implementation with the analysis of complexity, we claim that the neural network modeling approach provides

an elegant approach for the task of traffic modeling.

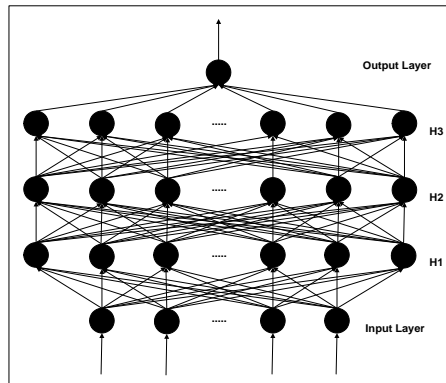


Fig. 1. Fixed structure neural network used for the task of modeling.

In the following section, we apply the proposed neural network modeling technique to reduce the packet loss rate of a shared buffer in a typical multiple source system.

### III. REDUCING PACKET LOSS IN MULTIPLE SOURCE SYSTEMS

Our application test bed relies on a multiple source queuing system as illustrated by Fig. 2. A multiple source queuing

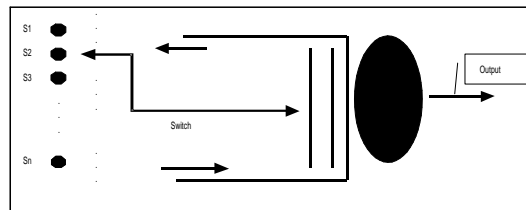


Fig. 2. The structure of a multiple source queuing system.

ing system consists of a number of sources sharing an available buffer space. The traffic pattern of each source includes the packets generated by a number of ON-OFF chaotic maps. An ON-OFF source model is generating traffic at a peak rate when it is active and becomes active as soon as the state variable of the describing chaotic map goes beyond a threshold value. The source becomes passive as soon as the state variable goes below the threshold value. We utilize double intermittency map with the following specifications as it generates a self-similar traffic pattern according to [5]. We select initial conditions in the range of  $x_0 \in [0.1, 0.3]$  along with a fixed threshold value of  $d = 0.7$  and parameters  $\epsilon_1 = 0.01$ ,  $\epsilon_2 = 0.05$ ,  $m = 5$ ,  $c_1 = 1.73$ ,  $c_2 = 267.49$  to obtain different traffic patterns for different sources. In our experiments, we rely on the same discrete time scales for both the neural network and the traffic generating intermittency maps. As an alternative, one may use different threshold values with fixed initial conditions to achieve varying traffic patterns.

We view each source and its corresponding buffer as a separate FIFO queuing system for different combinations of buffer management and service scheduling schemes. In our FIFO

model, there is a finite capacity buffer corresponding to each source storing generated packets before they get transmitted. The occupancy of each buffer is determined by the flow of the cells from the corresponding source and the rate at which the cells are serviced. A queue is identified by its buffer capacity, and its server capacity. In each queue, the arrival rate is compared with the service rate to determine whether the size of the queue is increasing or decreasing as well as whether the queue is losing cells. The model may be considered as the so-called burst scale queuing component of an ATM queuing system with a number of Virtual Channels (VCs) with each VC belonging to a traffic source as described by [18].

In order to show the performance of the modeling approach of Section II.B, four different buffer management scenarios are compared together in presence of FTDM and STDM scheduling algorithms. In the first method complete sharing mechanism is deployed, i.e., there is only one queue for all of the sources. The second method is a simple implementation of complete partitioning scheme in presence of FTDM and STDM in which the capacity of a central buffer is distributed equally among the sources. The third method is a simple implementation of partial sharing scheme that has three equal portions for the three sources with an additional shared portion available to all of the sources. The fourth method is the dynamic assignment of the buffer space relying on the results obtained from the neural network prediction algorithm, i.e., adjusting dedicated buffer space of each source according to its packet generation pattern. This is a generalization of the third method keeping the shared portion size fixed and adjusting the buffer space size of each source dynamically. The fourth method has a potential to outperform the other buffer management algorithms as it relies on predicted future information. It is important to mention that in case of the last three methods, there is a separate queue for each source storing the packets generated by that source.

In order to investigate the performance of the method, a triple source system is used. The traffic patterns of the first, second, and third source consist of an artificial traffic pattern generated by 30, 40, and 50 individual double intermittency map packet generators respectively. The traffic generated by each source is collected and sent to the corresponding buffer in a round robin manner. It is specially important to note that there is a difference among the number of packets generated by each source as the result of having a different number of ON-OFF packet generators per source. In order to evaluate the performance of different methods, the overall as well as per source loss rate of the system for different choices of buffer size with a fixed service rate are compared together. The buffer space can be shared among all of the sources or may be divided into equal portions for individual source usage. The server bandwidth may also be used according to FTDM or STDM scheduling mechanisms.

Fig. 3 and Fig. 4 show single source packet loss rate versus normalized buffer size diagram for the triple source queuing system in presence of FTDM scheduling algorithm. The single source is the source with the lowest generation rate to compare the fairness of different schemes. Fig. 5 and Fig. 6 show the same measures in presence of STDM scheduling algorithm. The packetized simulation results have been obtained from an iterative algorithm with a total number of ten million iterations

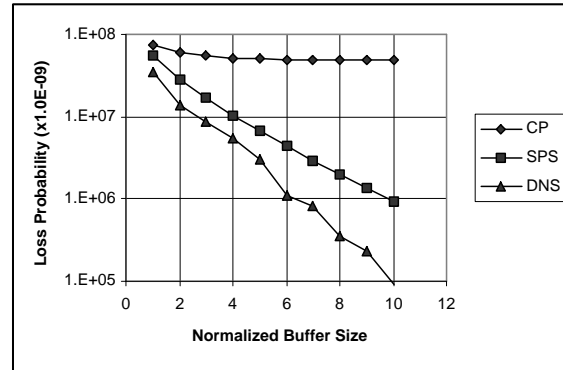


Fig. 3. Total packet loss probability versus buffer size diagram for the triple source queuing system using complete partitioning (CP), static partial sharing (SPS), and dynamic neural sharing (DNS) in presence of FTDM scheduling algorithm.

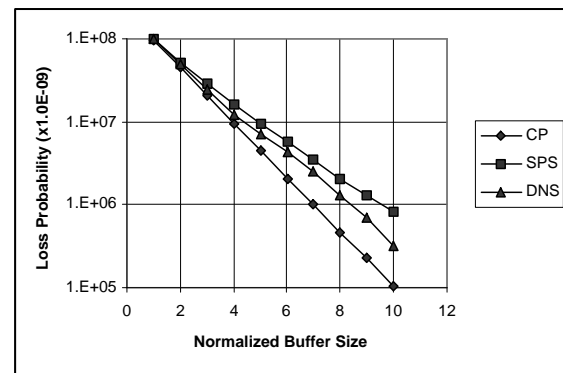


Fig. 4. Single source packet loss rate versus buffer size diagram for the triple source queuing system using complete partitioning (CP), static partial sharing (SPS), and dynamic neural sharing (DNS) in presence of FTDM scheduling algorithm.

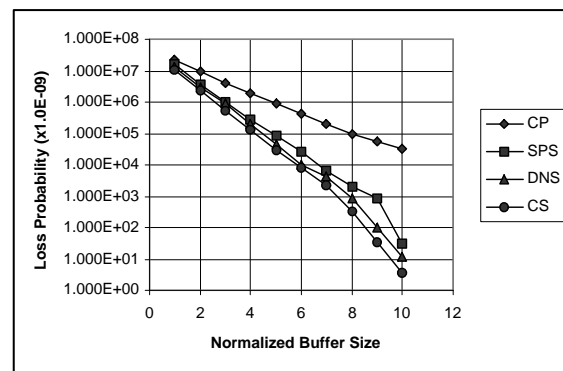


Fig. 5. Total packet loss rate versus buffer size diagram for the triple source queuing system using complete partitioning (CP), static partial sharing (SPS), dynamic neural sharing (DNS), and complete sharing (CS) in presence of STDM scheduling algorithm.

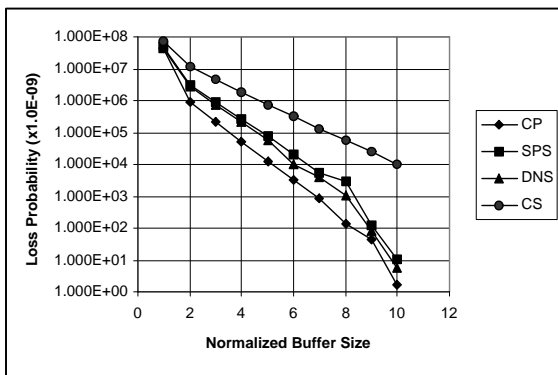


Fig. 6. Single source packet loss rate versus buffer size diagram for the triple source queuing system using complete partitioning (CP), static partial sharing (SPS), dynamic neural sharing (DNS), and complete sharing (CS) in presence of STDM scheduling algorithm.

per choice of buffer size. Applying a continuous learning algorithm, the fixed structure neural network has been able to follow the traffic pattern within the specified error range between 20 and 30 times covering an average of fifty samples per time. It is worth mentioning that the performance of different methods are very different as the result of applying different methods for traffic management of a heavily utilized system.

It is clearly observed from the figures that for both FTDM and STDM scheduling algorithm using neural sharing scheme, the total loss rate compared to complete partitioning scheme as well as per source loss rate compared to complete sharing and/or static partial sharing schemes are reduced. The results may be interpreted as the evidence that the neural sharing scheme has come up with a solution in between the two extreme cases addressing the trade off between fairness and efficiency. Comparing the results for static partial sharing and neural dynamic sharing show the higher efficiency of the latter method. This is a significant improvement compare to the other three schemes.

We close this section by mentioning some of the practical findings in the implementation of the algorithm. First, we note that the learning algorithm of the perceptron neural network used for the task of modeling is time consuming because of the rich dynamics of the traffic pattern that the neural network is trying to learn. Indeed, the neural network needs to access thousands of samples in each training period. In addition, all of the convergence results are strongly affected by the choice of initial conditions. In practice, the initial values of the neural network parameters play a crucial role in the convergence of the algorithm. As a practical result, setting the initial values of the weighting functions of the neural network at  $w_{ji}(0) = 0.01 \quad \forall i, j$  yields acceptable results. Additionally, one may set the weighting functions randomly in the order of 0.01 if facing biasing and saturation.

#### IV. CONCLUSION

In this paper, we studied packet loss reduction in a class of multiple source queuing systems as an application of neural network modeling of self-similar packet traffic. We modeled self-similar traffic patterns using a fixed structure perceptron neural

network.

We used a neural-based dynamic buffer management scheme called dynamic neural sharing to improve the loss performance of static partial sharing buffer management algorithm while considering fairness issue. Relying on the prediction power of neural networks, our neural-based algorithm was able to dynamically adjust the buffer allocation of individual sources in a multiple source system with a central shared or partitioned buffer.

We also compared the performance of different buffer management schemes, namely complete sharing, complete partitioning, static partial sharing, and dynamic neural sharing in presence of different server scheduling algorithms, fixed time division multiplexing and statistic time division multiplexing, and concluded that our dynamic neural sharing scheme was able to offer the best solution considering the trade off between fairness and loss issues.

#### REFERENCES

- [1] A. Adas, "Traffic Models in Broadband Networks", IEEE Communications Magazine, pp. 82-89, July 1997.
- [2] A. Alkhatib, M. Krunz, "Application of Chaos Theory to the Modeling of Compressed Video", In Proc. of the IEEE ICC 2000 Conference, Vol. 2, New Orleans, June 2000.
- [3] J. Beran, R. Sherman, M. S. Taqqu, W. Willinger, "Variable Bit Rate Video Traffic and Long Range Dependence", IEEE/ACM Trans. on Networking, Vol. 2, NO. 3, Apr. 1994.
- [4] D. E. Duffy, W. Willinger, "Statistical Analysis of CCSN/SS7 Traffic Data from Working CCS Subnetworks", IEEE JSAC, 1994.
- [5] A. Erramilli, R. P. Singh, P. Pruthi, "Chaotic Maps as Models of Packet Traffic.", ITC Vol. 14, pp. 329-338, 1994.
- [6] S. E. Fahlman, "An Empirical Study of Learning Speed in Back-Propagation Networks", Technical Report CMU-CS-88-162, Carnegie Mellon University, June 1988.
- [7] G. Gallasi, C. Rigolio, "ATM Bandwidth Assignment and Bandwidth Enforcement Policies", In Proc. of IEEE GLOBECOM '89, Dec. 1987.
- [8] M. Gerla, L. Kleinrock, "Flow Control: A Comparative Survey", IEEE Trans. on Commun., Vol. COM-28, No. 4, Apr. 1980.
- [9] G. Gomes, N. L. S. da Fonseca, N. Agoulmine, J. N. de Souza, "Neuro-computation of the Hurst Parameter", In Proc. of IEEE ITS, 2002.
- [10] K. M. Hellstern, P. Wirth, "Traffic Models for ISDN Data Users: Office Automation Application", In Proc. ITC-13, Denmark, 1991.
- [11] F. Kamoun, L. Kleinrock, "Analysis of Shared Storage in a Computer Network Node Environment under General Traffic Conditions", IEEE Trans. on Commun., Vol. COM-28, No. 7, Jul. 1980.
- [12] P. Kerami, L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique", Computer Networks, Vol.3, 1979.
- [13] W. E. Leland, W. Willinger, M. S. Taqqu, D. V. Willson, "Statistical Analysis and Stochastic Modeling of Self-Similar Datatrafic", ITC Vol. 14, pp. 319-328, 1994.
- [14] W. E. Leland, W. Willinger, M. S. Taqqu, D. V. Willson, "On the Self-Similar Nature of Ethernet Traffic", IEEE/ACM Trans. on Networking, Vol. 2, NO. 1, pp. 1-15, Feb. 1994.
- [15] A. Lin, J. A. Silvester, "Priority Queuing Strategies and Buffer Allocation Protocols in Traffic Control at an ATM Integrated Broadband Switching System", IEEE JSAC, Vol. 9, No. 9, Dec. 1991.
- [16] M. Minsky, S. A. Papert, "Perceptrons: An Introduction to Computational Geometry.", MIT Press, Cambridge, MA, expanded edition, 1988/1969.
- [17] A. Van Ooyen, B. Neuhuis, "Improving the Convergence of Back Propagation Algorithm", Neural Networks, Vol.5, No.3, 1992
- [18] J. M. Pitts, L. G. Cuthbert, M. Bocci, E. M. Scharf, "An Accelerated Simulation Technique for Modeling Burst Scale Queuing Behavior in ATM", ITC Vol. 14, pp. 777-786, 1994.
- [19] H. Yousefi'zadeh, "Neural Network Modeling of a Class of ON-OFF Source Models with Self-Similar Characteristics," In Proc. of the First Workshop on Fractals and Self-Similarity, ACM SIGKDD, July 2002.